IBM Reliable Scalable Cluster Technology for AIX 5L

RSCT Guide and Reference

SA22-7889-01

IBM Reliable Scalable Cluster Technology for AIX 5L

RSCT Guide and Reference

SA22-7889-01

Note
Before using this information and the product it supports, read the information in “Notices” on page 297.

Second Edition (October 2002)

This edition applies to versions 5.1 and 5.2 of AIX 5L (product numbers 5765-E61 and 5765-E62) 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: mhvrcfs@us.ibm.com

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

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

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

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

Contents

About This Book

This book describes various component subsystems of IBM's Reliable Scalable Cluster Technology (RSCT) that are included as part of the AIX 5L operating system. It 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.

Who should read this book

This book should be read by anyone who wants to:

- understand the core RSCT componentsshipped with AIX 5L.
- 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, "Creating and Administering an RSCT Peer Domain" on page 7 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 3, "Managing and Monitoring Resources Using RMC and Resource Managers" on page 29 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.
  - reference information for the various resource managers.
- Chapter 4, “Understanding and Administering Cluster Security Services” on page 129 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 5, “The Topology Services subsystem” on page 185 provides an overview of, and describes how you can troubleshoot problems related to, the Topology Services subsystem.
- Chapter 6, “The Group Services subsystem” on page 259 provides an overview of, and describes how you can troubleshoot problems related to, the Group Services subsystem.
- Chapter 7, “How to contact the IBM Support Center” on page 295 describes how to report problems related to RSCT.

Conventions and terminology used in this book

This book uses the following typographic conventions:

Convention	What it represents
bold	**Bold** words or characters represent system elements that you must use literally, such as: command names, file names, flag names, and path names.
`constant width`	Examples and information that the system displays appear in `constant-width` typeface.
italic	*Italicized* words or characters represent variable values that you must supply. *Italics* are also used for book titles, for the first use of a glossary term, and for general emphasis in text.
[item]	Used to indicate optional items.
<Key>	Used to indicate keys you press.

ISO 9000

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

32-Bit and 64-Bit Support for the UNIX98 Specification

Beginning with AIX® Version 4.3, the AIX operating system is designed to support The Open Group’s UNIX98 Specification for portability of UNIX-based operating systems. Many new interfaces, and some current ones, have been added or enhanced to meet this specification, making AIX Version 4.3 even more open and portable for applications.

At the same time, compatibility with previous releases of the operating system is preserved. This is accomplished by the creation of a new environment variable, which can be used to set the system environment on a per-system, per-user, or per-process basis.

To determine the proper way to develop a UNIX98-portable application, you may need to refer to The Open Group’s UNIX98 Specification, which can be obtained on

a CD-ROM by ordering *Go Solo 2: The Authorized Guide to Version 2 of the Single UNIX® Specification*, ISBN: 0-13-575689-8, a book that includes The Open Group's UNIX98 Specification on a CD-ROM.

Prerequisite and related information

See "Bibliography" on page 303 for a list of related publications.

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: mhvrcfs@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 for AIX 5L: 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 2, "Creating and Administering an RSCT Peer Domain" on page 7.

The following table lists the characteristics of the two domain types that can be present in your cluster. Keep in mind that an individual node can participate in both types of domains.

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 homogeneous.
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.

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 3, “Managing and Monitoring Resources Using RMC and Resource Managers” on page 29.

What are the RSCT Core 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. See "Audit Log Resource Manager" on page 88 for more information.
- the **Configuration resource manager** which provides the ability to create, administer, and monitor 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 "Configuration Resource Manager" on page 90 and Chapter 2, "Creating and Administering an RSCT Peer Domain" on page 7 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 43, "Advanced Resource Monitoring" on page 56 and "Event Response Resource Manager" on page 96.
- the **File System resource manager** manages file systems. For more information, refer to "File System Resource Manager" on page 103.
- the **Host resource manager** manages resources related to an individual machine. For more information, refer to "Host Resource Manager" on page 105.
- 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 on RMC and the core resource managers, refer to Chapter 3, "Managing and Monitoring Resources Using RMC and Resource Managers" on page 29.

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 40.

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 UNIX host based authentication is supported, but other security mechanisms may be supported in the future. In the case of UNIX 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 4, “Understanding and Administering Cluster Security Services” on page 129.

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 5, “The Topology Services subsystem” on page 185.

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 6, “The Group Services subsystem” on page 259.

Chapter 2. 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. An RSCT peer domain is a cluster of nodes configured for high availability; it could consist of all nodes in your cluster, or merely a subset of your overall cluster solution (which could also consist of nodes configured by CSM into a management domain). 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	1. The **preprpnode** 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 **startrpdomain** command to bring the peer domain online.	"Creating a Peer Domain" on page 9
Add nodes to an existing peer domain	1. The **preprpnode** command to prepare the security environment on the new node. 2. The **addrpnode** command to add the node to a peer domain. 3. The **startrpnode** command to bring the node online.	"Adding Nodes to an Existing Peer Domain" on page 14
Take a peer domain node offline	The **stoprpnode** command	"Taking a Peer Domain Node Offline" on page 17
Take a peer domain offline	The **stoprpdomain** command	"Taking a Peer Domain Offline" on page 17
Remove a node from a peer domain	The **rmrpnode** command	"Removing a Node From a Peer Domain" on page 18
Remove a peer domain	The **rmrpdomain** command	"Removing a Peer Domain" on page 19
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 20
Modify a communication group's characteristics (Topology Services' tunables)	the **chcomg** command to • 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 21

To:	Use:	For more information, refer to:
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 23
	the **mkcomg** command to create a communication group.	"Creating a Communication Group" on page 24
	the **rmcomg** command to remove a communication group.	"Removing a Communication Group" on page 26

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 to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*. If you encounter error messages while trying to perform the tasks outline in the chapter, refer to the manual *Reliable Scalable Cluster Technology for AIX 5L: Messages* for diagnosis and 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. These are available as part of the base AIX operating system.
  - rsct.core
  - rsct.basic
  - rsct.core.utils
  - rsct.core.sec
- The nodes of a particular RSCT peer domain must all be running AIX 5L. The nodes in a particular RSCT peer domain must also all have the same machine architecture. RSCT for AIX 5L will run on any IBM pSeries machine supported by AIX 5L.
- 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 all 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*
```

Migration

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 MANAGMENT_SCOPE=2
      ```

   b. Run the CompleteMigration action on the same node to complete the migration of the peer domain:

      ```
      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 **preprpnode** command.
- create a new peer domain definition by issuing the **mkrpdomain** command.
- bring the peer domain online using the **startrpdomain** command

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"), you first need to issue the **preprpnode** command to establish the initial trust between each node that will be in the peer domain, and the node from which you will issue the **mkrpdomain** command. Later, when you issue 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 **preprpnode** 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 136. 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 **preprpnode** 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:

```
preprpnode nodeA
```

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

```
preprpnode 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:

```
preprpnode -f node.list
```

When using the **preprpnode** command, you can identify the node by its IP address or by the long or short version of its DNS name. The **preprpnode** 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 4, "Understanding and Administering Cluster Security Services" on page 129.
- 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 40.

You can specify multiple nodes on the **preprpnode** 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 **preprpnode** command.

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

```
preprpnode -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 **preprpnode** 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 **preprpnode** command will still modify each node's RMC Access Control List (ACL) to enable access to peer domain resources between all nodes in the peer domain.

For more information on security issues related to the automatic transferal of public keys, refer to Chapter 4, "Understanding and Administering Cluster Security Services" on page 129.

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

Once you have run the **preprpnode** command on each peer domain node, 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 **preprpnode** 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 *ApplDomain*, you would, from *nodeA*, issue the command:

```
mkrpdomain ApplDomain nodeA nodeB nodeC
```

The above command creates the peer domain definition *ApplDomain* consisting of the nodes *nodeA*, *nodeB*, and *nodeC*.

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:

```
mkrpdomain -f node.list ApplDomain
```

The configuration resource manager will at this time create the communication group definitions needed to later enable liveness checks (Topology Services' "heartbeating") 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 issue the **startrpdomain** command (described next in "Step 3: Bring the Peer Domain Online" on page 12), the configuration resource manager will supply the communication group definition information to Topology Services. For more information on Topology Services, refer to Chapter 5, "The Topology Services subsystem" on page 185.

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 ApplDomain
```

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 ApplDomain 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*.

Once you have created your peer domain definition using the **mkrpdomain** command, you can bring the peer domain online using the **startrpdomain** command (described next in "Step 3: Bring the Peer Domain Online").

Step 3: Bring the Peer Domain Online

The **startrpdomain** 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 **startrpdomain** command the name of a peer domain you have already defined using the **mkrpdomain** command. For example, to bring the peer domain *ApplDomain* online, you would, from any of the nodes in the peer domain, issue the command:

```
startrpdomain ApplDomain
```

When bringing the peer domain online, the **startrpdomain** 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 (in other words, Topology Services' ″heartbeats″) 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 20). 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 21. To modify network interfaces in the communication group, refer to "Modifying a Communication Group's Network Interface" on page 23.

By default, the **startrpdomain** command will not attempt to bring the peer domain online until a quorum of nodes has been contacted. A quorum is defined as *n*/2+1 where *n* is the number of nodes defined in the peer domain. 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 (and not just a quorum of nodes), specify the **startrpdomain** 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:

```
startrpdomain -A ApplDomain
```

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

```
startrpdomain -t 240 ApplDomain
```

Once the domain is brought online, you can use the **lsrpnode** command to list the nodes in the domain and their status. You issue this command from any node in the peer domain.

```
lsrpnode
```

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

```
Name                OpState             RSCTVersion
nodeA               online              2.2.1.10
nodeB               online              2.2.1.10
nodeC               online              2.2.1.10
nodeD               offline             2.2.1.10
nodeE               offline             2.2.1.10
```

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:

```
export CT_MANAGEMENT_SCOPE=2
```

This tells RMC that the management scope is a peer domain. 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, they are used by 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.

You can later take the peer domain offline using the **stoprpdomain** command. You can also take an individual node offline using the **stoprpnode** command. These commands are described in "Taking Individual Nodes of a Peer Domain, or an Entire Peer Domain, Offline" on page 17.

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

Adding Nodes to an Existing Peer Domain

"Creating a Peer Domain" on page 9 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 **preprpnode** command
- add the node to the peer domain definition using the **addrpnode** command
- bring the node online in the peer domain using the **startrpnode** or **startrpdomain** command

Step 1: Prepare Security Environment on the Node

Before you can add a node to a peer domain using the **addrpnode** command (described next in "Step 2: Add Node To the Peer Domain" on page 15), you first need to issue the **preprpnode** command to establish the initial trust between the node to be added, and the node from which you will issue the **addrpnode** command. Later, when you issue the **addrpnode** 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 **addrpnode** command is called the *originator node*. To establish trust between the originator node and the node to be added to the peer domain, you must first run the **preprpnode** command on the node to be added.

For example, say you will be issuing the **addrpnode** command on *nodeA*. From the node you wish to add to the peer domain, issue the command:

```
preprpnode nodeA
```

You identify the node by its IP address or by the long or short version of its DNS name.

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

```
preprpnode 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:

```
preprpnode -f node.list
```

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

```
preprpnode -k nodeA
```

Using the **-k** flag disables the automatic transfer of public keys. You may also want to use the **-k** flag is 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 **preprpnode** 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 **preprpnode** command will still modify each node's RMC ACL file to enable access to peer domain resources between all 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 136.

For complete syntax information on the **lsrpnode** and **preprpnode** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: 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 9), 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 quorum of *n*/2+1 nodes (where *n* is the number of nodes defined in the cluster) to be active.

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

```
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 **startrpnode** command (described next in "Step 3: Bring Node Online in the Peer Domain" on page 16), 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 19. For more information on Topology Services, refer to Chapter 5, "The Topology Services subsystem" on page 185.

For complete syntax information on the **addrpnode** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: 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 **startrpnode** or **startrpdomain** command. These commands are described next in "Step 3: Bring Node Online in the Peer Domain" on page 16.

Step 3: Bring Node Online in the Peer Domain

The **startrpnode** command brings an offline node online in the current peer domain. To see which nodes are currently defined in the peer domain, but not online, use the **lsrpnode** command from any node in the peer domain.

```
lsrpnode
```

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

```
Name                 OpState              RSCTVersion
nodeA                online               2.2.1.10
nodeB                online               2.2.1.10
nodeC                online               2.2.1.10
nodeD                offline              2.2.1.10
nodeE                offline              2.2.1.10
```

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 **startrpnode** 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
ApplDomain    offline     2.2                  no               12347    12348
```

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

```
startrpnode 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 **startrpdomain** command to bring all of the offline nodes online in this peer domain. For example, to bring the peer domain *ApplDomain* online, you would, from any node, issue the command:

```
startrpdomain ApplDomain
```

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

For more information about the **startrpdomain** command, refer to the directions for creating a peer domain (the **startrpdomain** command is described in more detail in “Step 3: Bring the Peer Domain Online” on page 12 of those directions). For complete syntax information on the **startrpnode**, **startrpdomain**, **lsrpnode**, or **lsrpdomain** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: 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 **stoprpnode** command
- Take a peer domain offline using the **stoprpdomain** command

Taking a Peer Domain Node Offline

The **stoprpnode** 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 18). 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 **stoprpnode** 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 **lsrpnode** command for any node in the peer domain:

```
lsrpnode
```

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             2.2.1.10
nodeB               online              2.2.1.10
nodeC               online              2.2.1.10
nodeD               online              2.2.1.10
nodeE               offline             2.2.1.10
```

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

```
stoprpnode nodeA
```

You can also take multiple nodes offline. For example:

```
stoprpnode nodeA nodeB
```

An RSCT subsystem (such as Topology Services or Group Services) may reject the **stoprpnode** 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 **stoprpnode** command's **-f** flag. For example:

```
stoprpnode -f nodeA
```

To later bring the node back online, use the **startrpnode** command as described in "Step 3: Bring Node Online in the Peer Domain" on page 16. For complete syntax information on the **stoprpnode** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: 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 *ApplDomain* offline:

```
stoprpdomain ApplDomain
```

An RSCT subsystem (such as Topology Services or Group Services) may reject the **stoprpnode** 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 ApplDomain
```

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 12. For complete syntax information on the **stoprpdomain** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: 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 **rmrpnode** command
- remove a peer domain definition using the **rmrpdomain** 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 **stoprpnode** command to take it offline. For more information on the **stoprpnode** command, refer to "Taking a Peer Domain Node Offline" on page 17.

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

```
lsrpnode
```

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

```
Name                OpState             RSCTVersion
nodeA               offline             2.2.1.10
nodeB               online              2.2.1.10
nodeC               online              2.2.1.10
nodeD               online              2.2.1.10
nodeE               offline             2.2.1.10
```

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

```
rmrpnode nodeA
```

You can also remove multiple nodes from the peer domain:

```
rmrpnode nodeA nodeE
```

For complete syntax information on the **rmrpnode** and **lsrpnode** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: 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 **rmrpdomain** command from any online node in the peer domain. You pass the **rmrpdomain** command the name of the peer domain. For example, to remove the peer domain *ApplDomain*:

```
rmrpdomain ApplDomain
```

The **rmrpdomain** 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 **rmrpdomain** is run (for example, the network is down or the node is inoperative), the **rmrpdomain** 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 **rmrpdomain** command was run, you will need to run the **rmrpdomain** command from each node that did not have their peer domain definition removed. You should include the **-f** option to force the removal:

```
rmrpdomain -f ApplDomain
```

You should also use the **-f** flag if an RSCT subsystem (such as Topology Services or Group Services) rejects the **rmrpdomain** 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 **rmrpdomain** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*.

Understanding and Working With Communication Groups

Communication groups control how liveness checks (in other words, Topology Service's "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 23.

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 21.

Listing Communication Groups

The **lscomg** 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*:

```
lscomg
```

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                      NIMParameters
/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 **lscomg** command. For example, to list information about the communication group *ComGrp*, enter:

```
lscomg ComGrp
```

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

```
lscomg -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 **chcomg** command as described in “Modifying a Communication Group’s Characteristics”. For complete syntax information on the **lscomg** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: 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 **chcomg** 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 **lscomg** command as described in “Listing Communication Groups” on page 20.

You can also use the **chcomg** 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 23 for more information.

For complete syntax information on the **chcomg** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: 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 **lscomg** command as described in "Listing Communication Groups" on page 20. To modify a communication group's sensitivity setting, use the **chcomg** 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 **lscomg** command as described in "Listing Communication Groups" on page 20. 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 **lscomg** command as described in "Listing Communication Groups" on page 20. 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 **lscomg** command as described in "Listing Communication Groups" on page 20. 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 **lscomg** command as described in "Listing Communication Groups" on page 20. 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
```

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
```

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 unadvisable**. 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 **startrpdomain** command or the new node is brought online by the **startrpnode** 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).

Modifying a Communication Group's Network Interface

"Modifying a Communication Group's Characteristics" on page 21 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.

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 20 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 **lscomg** command (as described in "Listing Communication Groups" on page 20). To change any of the settings, you can use the **chcomg** command (as described in "Modifying a Communication Group's Characteristics" on page 21). For complete syntax information on the **mkcomg** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: 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 **lscomg** command as described in “Listing Communication Groups” on page 20. 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 23).

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*.

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 5, “The Topology Services subsystem” on page 185. For a complete discussion of Group Services, refer to Chapter 6, “The Group Services subsystem” on page 259. For more information on the IBM.RSCTParameters resource class, refer to “RSCT Parameters Resource Class” on page 95.

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**) .

Chapter 3. Managing and Monitoring Resources Using RMC and Resource Managers

The Resource Management 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 30 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 40 describes how to grant users the permissions they need to use RMC and the resource managers effectively.
- "Basic Resource Monitoring" on page 43 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 56 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 43 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 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 78 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.

- "Resource Manager Reference" on page 87 provides reference information for the resource managers provided with RSCT.

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 or 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 43 and "Advanced Resource Monitoring" on page 56).
- how RMC implements authorization (described in more detail later in "Managing User Access to Resources Using RMC ACL Files" on page 40).
- 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 simply 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 43 and "Advanced Resource Monitoring" on page 56, 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 a dynamic 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 that 90 percent full. The percentage of space used by a file system is one of its dynamic attribute values. It would 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 59) 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 43 and "Advanced Resource Monitoring" on page 56) where conditions used to trigger responses are logical expressions based on the value of a dynamic 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 2, “Creating and Administering an RSCT Peer Domain” on page 7) provides commands for creating creating a peer domain, adding nodes to the domain, taking the domain offline, and so on.

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 1. Resource Managers Provided with RSCT

Resource manager:	Description:
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 51 or the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*. For reference information on its resource classes and attributes, refer to “Audit Log Resource Manager” on page 88.
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 2, “Creating and Administering an RSCT Peer Domain” on page 7 or the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*. For reference information on its resource classes and attributes, refer to “Configuration Resource Manager” on page 90.
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 43 and “Advanced Resource Monitoring” on page 56. Complete syntax information on these commands is provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*. For reference information on this resource manager’s resource classes and attributes, refer to “Event Response Resource Manager” on page 96.
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 dynamic resource attributes using the event response resource manager. For reference information on the file system resource manager’s resource classes and attributes, refer to “File System Resource Manager” on page 103.
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 dynamic resource attributes using the event response resource manager. For reference information on the host resource manager, refer to “Host Resource Manager” on page 105.
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 *sensor*. A sensor is merely a command that the RMC subsystem runs at specified intervals 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 dynamic 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 67 or the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*. For reference information on its resource classes and attributes, refer to “Sensor Resource Manager” on page 127.

How Does RMC and the Resource Managers Enable You to Manage Resources?

As already described, RMC provides resource an 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 **lsrsrcdef** 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 **lsrsrcdef** lists a resource class definition.

For complete syntax and reference information on the generic RMC commands refer to the *Reliable Scalable Cluster Technology for AIX 5L: 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 a dynamic 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.

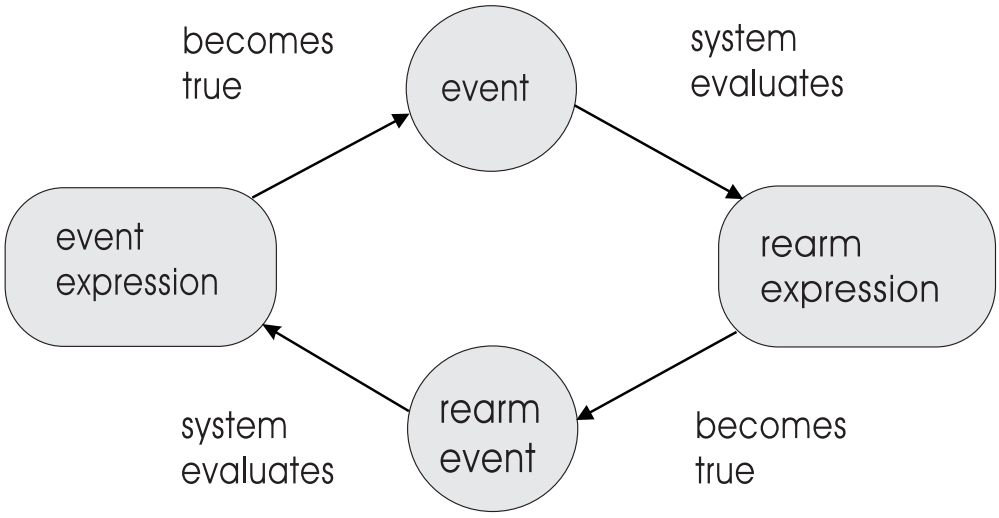
As already stated, each event expression refers to a particular dynamic attribute value, which will be polled by RMC at set intervals to determine if the expression tests true. RMC keeps track of the previously observed value of the dynamic attribute, so the event expression can compare the currently observed value with the previously observed value. If the event expression suffixes the dynamic attribute name with ″@P″, this represents the previously observed value of the dynamic 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.

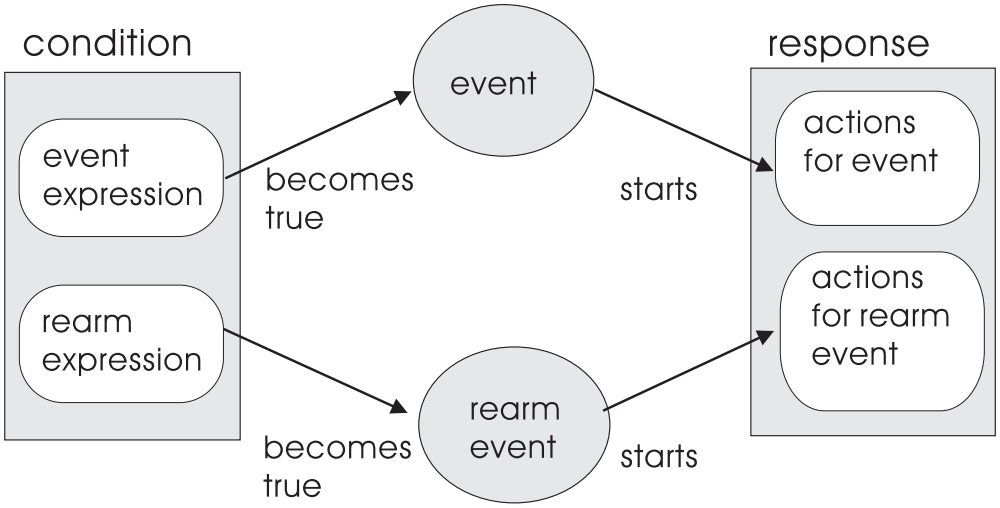
How Do I Create Conditions?: It is important to understand that, in most cases, you will not need to create conditions since we have provided a set of predefined conditions to monitor most of the dynamic attributes defined by the file system resource manager and host resource manager. You can list these predefined conditions using the **lscondition** command described in "Listing Conditions" on page 43. The predefined conditions for the RSCT resources managers are also listed by resource class in "Resource Manager Reference" on page 87. If the predefined conditions are not sufficient, you can create your own to monitor any dynamic attribute. To do this, you use the **mkcondition** command as described in "Creating a Condition" on page 59. 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 dynamic 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 to retrieve one or more user-defined values. For more information, refer to "Creating Event Sensor Commands for Monitoring" on page 67.

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 72), you can add a single action to the response. You can then use the **chresponse** command (as described in “Modifying a Response” on page 77) 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 2. Predefined Responses

Response Name	Action(s)	Description	Action in effect:
Broadcast event anytime	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.

Table 2. Predefined Responses (continued)

Response Name	Action(s)	Description	Action in effect:
Critical notification	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 whenever an event or a rearm event occurs.	All day, everyday.
	broadcast message	Uses the predefined action script **/usr/sbin/rsct/bin/wallevent** to broadcast the event or the rearm event to all logged-in users.	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.
Informational notification	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.
Log event anytime	log event	Uses the predefined action script **/usr/sbin/rsct/bin/logevent** to log an entry to **/tmp/systemlEvents** whenever an event or a rearm event occurs.	All day, everyday.
Send e-mail to 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.
Send e-mail to 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-12AM, Monday to Friday; 12AM-8AM, Monday to Friday; all day, Saturday and Sunday.
Warning notification	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 whenever an event or a rearm event occurs.	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 47 and “Starting Condition Monitoring” on page 48.

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 43). The predefined conditions are also listed by resource class in "Resource Manager Reference" on page 87.

You can also create a condition based on any dynamic attribute of a resource class. You can list the dynamic attributes using the **lsrsrc** command (described in "Creating a Condition From Scratch" on page 62). Like the predefined conditions, the dynamic attributes are also listed by resource class in "Resource Manager Reference" on page 87.

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 dynamic 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 a dynamic 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 40.

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 3. 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: **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. The *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* contains complete reference information for all of the 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 42.

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 or the keyword OTHER. The stanza name OTHER applies to all resource classes that are not otherwise specified in the file. 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, optionally, a set of permissions. The `user_identifier` portion of the stanza line can have any one of the forms shown in the following table:

This Form:	**Identifies:**
user_name@host_name	A particular user. The *host_name* is either a fully-qualified host domain name or the keyword LOCALHOST. The keyword LOCALHOST identifies the node running the RMC subsystem.
host_name	Any user running the RMC application on the named host. The *host_name* is either a fully-qualified host domain name or the keyword LOCALHOST. The keyword LOCALHOST identifies the node running the RMC subsystem.
*	Any user running an RMC application on any host.
UNAUTHENT	Specifies an unauthenticated user.

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

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.

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.
w	write permission. This allows uses 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.

Examples of ACL File Stanzas

1. The ACL file on the management server of a management domain should look similar to the following example. The UNAUTHENT keyword must be present for RMC to work properly.

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

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

2. The ACL file on the managed node of a management domain should look similar to the following example. The UNAUTHENT keyword must be present for RMC to work properly.

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

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

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 40 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, they are logged to **/var/ct/IW/log/mc/default**.

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 34, 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 56. For detailed syntax information on any the commands described in this section, refer to the *Reliable Scalable Cluster Technology for AIX 5L: 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:

- **lscondition** for listing information about conditions.
- **lsresponse** for listing information about responses.
- **lscondresp** for listing information about condition/response associations.

Listing Conditions

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

```
lscondition
```

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 59). 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 **lscondition** command. For example, to get detailed information about the `"FileSystem space used"` condition, enter the following at the command prompt:

```
lscondition "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 = "l"
```

Targeting Node(s):

The **lscondition** 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 **lscondition** 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 **lscondition** command man page and “How Do I Determine the Target Nodes For a Command?” on page 39.

For detailed syntax information on the **lscondition** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*.

Listing Responses

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

```
lsresponse
```

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 72).

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

```
lsresponse "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/notifyevent 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 39.

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

Listing Condition/Response Associations

As described in “Listing Conditions” on page 43 and “Listing Responses” on page 44, 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 56. 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 47), 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 **lscondresp** command. For example, to list the condition/response associations for the `"FileSysem space used"` condition, you would enter the following at the command prompt:

```
lscondresp "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 **lscondresp** command’s **-a** option. For example:

```
lscondresp -a "FileSystem space used"
```

Output would show only the active condition/response associations for the `"FileSysem space used"` condition.

```
Condition                 Response                      State        Location
"FileSystem space used"   "Broadcast event on-shift"    "Active"     nodeA
```

Targeting Node(s):

The **lscondresp** 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 **lscondresp** 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 **lscondresp** command man page and "How Do I Determine the Target Nodes For a Command?" on page 39.

For detailed syntax information on the **lscondresp** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 56. To list all the available conditions you can use in creating your condition/response association, use the **lscondition** command as described in "Listing Conditions" on page 43. To list all the available responses you can use in creating your condition/response association, use the **lsresponse** command as described in "Listing Responses" on page 44.

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 48).

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, 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 **mkcondresp** command man page and "How Do I Determine the Target Nodes For a Command?" on page 39.

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 46).

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").

For detailed syntax information on the **mkcondresp** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 47), 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 46. 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 39.

For detailed syntax information on the **startcondresp** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 **lscondition** command as described in “Listing Conditions” on page 43.

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, simple 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 **lscondresp** command as described in “Listing Condition/Response Associations” on page 46.) 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"
```

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 39.

For detailed syntax information on the **stopcondresp** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 46. 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"
```

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 39.

For detailed syntax information on the **rmcondresp** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 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 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 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 proc2 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 proc3 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 14:56:00      ERRM Info     Event from Processor idle time that occurre
d at 07/27/02 14:55:58 953165 caused /usr/sbin/rsct/bin/logevent /tmp/systemEven
ts 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 occurre
d at 07/27/02 14:55:58 953165 caused /usr/sbin/rsct/bin/logevent /tmp/systemEven
ts 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 occurre
d at 07/27/02 14:55:58 953165 caused /usr/sbin/rsct/bin/logevent /tmp/systemEven
ts 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 occurre
d at 07/27/02 14:55:58 953165 caused /usr/sbin/rsct/bin/logevent /tmp/systemEven
ts 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 sucessfully.
```

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**) the 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 78. 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 (and the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*)

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 `#mmddhhmmyyyy`, 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 `#-mmddhhmmyyyy`. 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 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 39.

For detailed syntax information on the **lsaudrec** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 **lsaudrec** command, the **rmaudrec** command has an **-s** option for specifying the selection string expression, which takes the same form as it does on the **lsaudrec** 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 39.

For detailed syntax information on the **lsaudrec** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 that the indicated limit. The default size limit of the audit log is 1 Megabyte, which will be an insufficient size for a

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*). 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 signi
ficant 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 deleted 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 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 or the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*

Advanced Resource Monitoring

As described in “Basic Resource Monitoring” on page 43, 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 47, 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 dynamic attribute you want to monitor for one or more resource of a particular resource class. 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 periodically 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 69.

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 47 and “Starting Condition Monitoring” on page 48. For detailed syntax information on any the commands described in this section, refer to the *Reliable Scalable Cluster Technology for AIX 5L: 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 43, 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         = "l"
```

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 4. 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 48 and “Stopping Condition Monitoring” on page 49.
`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 4. Explanation of **lscondition** command output (continued)*

This line of the lscondition command output:	Indicates:	Notes
`EventExpression = "PercentTotUsed > 90"`	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 47), and start monitoring (as described in "Starting Condition Monitoring" on page 48), 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). An event expression includes a dynamic attribute, a mathematical comparison symbol, and a constant. 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.	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"`	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. 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.	Specified by the **-E** flag of both the **mkcondition** and **chcondition** commands.
`RearmDescription = "The event will be rearmed when the percent of the sp ace 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 78.	Specified by the **-s** flag of both the **mkcondition** and **chcondition** commands.

*Table 4. 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 36.	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 36.	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 43. You can also refer to “Resource Manager Reference” on page 87 which lists the predefined conditions by resource manager and resource class. 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 47 and “Starting Condition Monitoring” on page 48.
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 60 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 a dynamic attribute containing the value you want to monitor. If none of them do, you can extend RMC by creating a *sensor* — a command to be run periodically by RMC to retrieve the value you want to monitor. Refer to “Creating a Condition From Scratch” on page 62.

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).

For more information, refer to the **mkcondition** command man page and "How Do I Determine the Target Nodes For a Command?" on page 39.

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 43, 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         = "l"
```

This **lscondition** output (described in detail in Table 4 on page 57) 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 68), 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         = "l"
```

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 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 39.

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 78. 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 4 on page 57 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 4 on page 57, 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 dynamic 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. Refer to "Host Resource Manager" on page 105 for a description of each of the resource classes managed by the Host resource manager. This reference also lists, by resource class, the dynamic attributes you can monitor.
- 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. Refer to "File System Resource Manager" on page 103 for a description of each of the resource classes managed by the File System resource manager. This reference also lists, by resource class, the dynamic attributes you can monitor.

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 dynamic attributes they enable you to monitor. You can also examine the available resource classes and dynamic 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 34 and the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* for more information on RMC commands.

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 periodically by RMC to retrieve the value you want to monitor. Refer to "Creating Event Sensor Commands for Monitoring" on page 67 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 refer to "Resource Manager Reference" on page 87, or enter the following command to list the available resource classes.

```
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. These are described in “Processor Resource Class” on page 118, but 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 60. 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 the the next step.

2. **Design an event expression that will test the attribute for the condition of interest.** Once you have identified the dynamic 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

dynamic 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 a dynamic attribute. If an event expression appends a dynamic attribute name with ″@P″, this refers to the previously observed value of the dynamic attribute. An event expression might use this capability to compare the currently observed value of the dynamic 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 78.

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. **Determinine 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 or its entry in the *Reliable Scalable Cluster Technology for AIX 5L: 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 dynamic 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.

For detailed syntax information on the **mkcondition** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*

Creating Event Sensor Commands for Monitoring: When none of the dynamic attributes of the available resource classes 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 subsystem runs at specified intervals 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. 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 not existing dynamic 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 you 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:

   ```
   mksensor 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 **mksensor** command. For

example, to specify that RMC should execute our **numlogins** script at five-minute (300-second) intervals, you would enter:

```
mksensor -i 300 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 **mksensor** 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, than the root user must issue the **mksensor** command.

4. **Create a condition to monitor a dynamic attribute of the sensor.** The **mksensor** command creates a sensor resource of the Sensor resource class. The sensor command script associated with this resource is executed at set intervals by RMC, and any value returned by the script are 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.

   ```
   mkcondition -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 **mksensor** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 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"
```

For detailed syntax information on the **chcondition** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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
```

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 39.

For detailed syntax information on the **rmcondition** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 44, 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/notifyevent 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 5. 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 **mkreponse** command's **-p** flag.

*Table 5. Explanation of **lsresponse** command output (continued)*

This line of the lsresponse command output:	Indicates:	Notes
`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.
`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 51.	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 51.	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 7 on page 76.)	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 2 on page 37.

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 2 on page 37. For instructions on listing the predefined responses available on your system, refer to "Listing Responses" on page 44. 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 47 and "Starting Condition Monitoring" on page 48.

One you understand the information contained in Table 5 on page 70, 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 three predefined action scripts that you can associate with the action. You can also create you 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/** are are described in the following table.

Table 6. Predefined Response Scripts

Script	Description
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.
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 74.

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 or its entry in the *Reliable Scalable Cluster Technology for AIX 5L: 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 or 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"
   ```

To add additional actions to a response, use the **chresponse** command, as described in “Modifying a Response” on page 77.

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).

For more information, refer to the **mkresponse** command man page and “How Do I Determine the Target Nodes For a Command?” on page 39.

For detailed syntax information on the **mkresponse** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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 you 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 6 on page 72). 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 7. Event Response Resource Manager Environment Variables

This environment variable:	Will contain:
ERRM_ATTR_NAME	The display name of the dynamic attribute used in the expression that caused this event to occur.
ERRM_ATTR_PNAME	The programmatic name of the dynamic 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 ct_data_type_t of the dynamic attribute that changed to cause this event. The following is a list of valid values for this environment variable: CT_INT32, CT_UINT32, CT_INT64,CT_UINT64, CT_FLOAT32, CT_FLOAT64, CT_CHAR_PTR, CT_BINARY_PTR, and CT_SD_PTR. The actual value of the dynamic attribute is stored in the **ERRM_VALUE** environment variable (except for dynamic attributes with a data type of CT_NONE).
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 dynamic attribute that changed, thus causing the event to occur.
ERRM_RSRC_CLASS_NAME	The display name of the resource class containing the dynamic 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 dynamic 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 CT_SD_PTR. For example: CT_CHAR_PTR, CT_UINT32_ARRAY, CT_UINT32_ARRAY, CT_UINT32_ARRAY.
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 *event* or *rearm event*.
ERRM_TYPEID	The value of **ERRM_TYPE**. This environment variable will have one of the following values: 0 (Event) or 1 (Rearm Event).

Table 7. Event Response Resource Manager Environment Variables (continued)

This environment variable:	Will contain:
ERRM_VALUE	The value of the dynamic attribute that caused the event to occur for all dynamic attributes except those with a data type of CT_NONE. The following data types are represented with this environment variable as a decimal string: CT_INT32, CT_UINT32, CT_INT64, CT_UINT64, CT_FLOAT32, and CT_FLOAT64. CT_CHAR_PTR is represented as a string for this environment variable. CT_BINARY_PTR is represented as a hexadecimal string separated by spaces. CT_SD_PTR is enclosed in square brackets and has individual entries within the SD that are separated by commas. Arrays within an SD are enclosed within braces {}. For example, [″My Resource Name″,{1,5,7},{0,9000,20000},{7000,11000,25000}] See the definition of **ERRM_SD_DATA_TYPES** for an explanation of the data types that these values represent.

Note:

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:

```
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"
```

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*.

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 72. 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 39.

For detailed syntax information on the **chresponse** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: 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
```

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 39.

For detailed syntax information on the **rmresponse** command, refer to its online man page or the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*.

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 34 and “What is a Rearm Event Expression?” on page 35.

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*.

Table 8. 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 68.
chrsrc	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.	*Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*
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 51.
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.	*Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*
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 59.
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 54.
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.	*Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*

SQL Restrictions

SQL syntax is supported for selection strings. The following table relates the RMC terminology to SQL terminology.

Table 9. 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 10. 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
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 0x0102030405060708090a0b0c0d0e0f1011121314"
```

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 81, that are separated by commas and enclosed in square brackets. For example, ['abc',1,{3,4,5}] specifies an SD that consists of three elements: (a) the string 'abc', (b) the integer value 1, and (c) the three-element array {3,4,5}.

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

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

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

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

For example, `a.b`

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

How Variable Names Are Handled

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

A variable name is restricted to include only 7-bit ASCII characters that are alphanumeric (a-z, A-Z, 0-9) or the underscore character (_). The name must begin with an alphabetic character.

When the expression is used by the RMC subsystem for an event or a rearm event, the name can have a suffix that is the '@' character followed by 'P', which refers to 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 11. 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

Table 11. Operators That Can Be Used in Expressions (continued)

Operator	Description	Left Data Types	Right Data Types	Example	Notes
|	Bitwise OR	Integers	Integers	″2|4″ results in 6	None
&	Bitwise AND	Integers	Integers	″3&2″ results in 2	None
~	Bitwise complement	None	Integers	~0x0000ffff results in 0xffff0000	None
^	Exclusive OR	Integers	Integers	0x0000aaaa^0x0000ffff results in 0x00005555	None
>>	Right shift	Integers	Integers	0x0fff>>4 results in 0x00ff	None
<<	Left shift	Integers	Integers	″0x0ffff<<4″ results in 0xffff0	None
== =	Equality	All but SDs	All but SDs	″2==2″ results in 1 ″2=2″ results in 1	Result is true (1) or false (0)
!= <>	Inequality	All but SDs	All but SDs	″2!=2″ results in 0 ″2<>2″ results in 0	Result is true (1) or false (0)
>	Greater than	Integer,float	Integer,float	″2>3″ results in 0	Result is true (1) or false (0)
>=	Greater than or equal	Integer,float	Integer,float	″4>=3″ results in 1	Result is true (1) or false (0)
<	Less than	Integer,float	Integer,float	″4<3″ results in 0	Result is true (1) or false (0)
<=	Less than or equal	Integer,float	Integer,float	″2<=3″ results in 1	Result is true (1) or false (0)
=~	Pattern match	Strings	Strings	″abc″=~″a.*″ results in 1	Right operand is interpreted as an extended regular expression
!~	Not pattern match	Strings	Strings	″abc″!~″a.*″ results in 0	Right operand is interpreted as an extended regular expression
=? LIKE like	SQL pattern match	Strings	Strings	″abc″=? ″a%″ results in 1	Right operand is interpreted as a SQL pattern
!? NOT LIKE not like	Not SQL pattern match	Strings	Strings	″abc″!? ″a%″ results in 0	Right operand is interpreted as a SQL pattern
|< IN in	Contains any	All but SDs	All but SDs	″{1..5}|<{2,10}″ results in 1	Result is true (1) if left operand contains any value from right operand
>< NOT IN not in	Contains none	All but SDs	All but SDs	″{1..5}><{2,10}″ results in 1	Result is true (1) if left operand contains no value from right operand
&<	Contains all	All but SDs	All but SDs	″{1..5}&<{2,10}″ results in 0	Result is true (1) if left operand contains all values from right operand

Table 11. Operators That Can Be Used in Expressions (continued)

Operator	Description	Left Data Types	Right Data Types	Example	Notes
|| OR or	Logical OR	Integers	Integers	″(1<2)||(2>4)″ results in 1	Result is true (1) or false (0)
&& AND and	Logical AND	Integers	Integers	″(1<2)&&(2>4)″ results in 0	Result is true (1) or false (0)
! NOT not	Logical NOT	None	Integers	″!(2==4)″ results in 1	Result is true (1) or false (0)

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

Table 12. Operator Precedence

Operators	Description
.	Structured data element separator
~ ! NOT not - +	Bitwise complement Logical not Unary minus Unary plus
* / %	Multiplication Division Modulo
+ -	Addition Subtraction
<< >>	Left shift Right shift

Table 12. Operator Precedence (continued)

Operators	Description
<	Less than
<=	Less than or equal
>	Greater than
>=	Greater than or equal
==	Equality
!=	Inequality
=?	SQL match
LIKE	
like	
!?	SQL not match
=~	Reg expr match
!~	Reg expr not match
?=	Reg expr match (compat)
|<	Contains any
IN	
in	
><	Contains none
NOT IN	
not in	
&<	Contains all
&	Bitwise AND
^	Bitwise exclusive OR
|	Bitwise inclusive OR
&&	Logical AND
||	Logical OR
,	List separator

Pattern Matching

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

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

Examples of Expressions

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

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

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

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

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

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

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

Resource Manager Reference

A resource manager is a process that maps resource and resource-class abstractions into calls and commands for one or more specific types of resources. A resource manager is a stand-alone daemon. The resource manager contains definitions of all resource classes that the resource manager supports. A resource class definition includes a description of all attributes, actions, and other characteristics of a resource class. These resource classes are accessible and their attributes can be manipulated by the user through the command line.

See the man pages for the commands or the *Reliable Scalable Cluster Technology for AIX: Technical Reference* to learn how to access the resource classes and manipulate their attributes through the command line interface.

The following resource managers are provided:

Audit Log resource manager (IBM.AuditRM)
: Provides a system-wide facility for recording information about the system's operation, which is particularly useful for tracking subsystems running in the background. (See "Audit Log Resource Manager" on page 88 for details.)

Configuration resource manager (IBM.ConfigRM)
: Provides the ability to monitor an RSCT peer domain. (See "Configuration Resource Manager" on page 90 for details)

Event Response resource manager (IBM.ERRM)
: Provides the ability to take actions in response to conditions occurring on the system. (See "Event Response Resource Manager" on page 96 for details.)

File System resource manager (IBM.FSRM)
: Monitors file systems. (See "File System Resource Manager" on page 103 for details.)

Host resource manager (IBM.HostRM)
: Monitors resources related to an individual machine. The types of values that are provided relate to load (processes, paging space, and memory

usage) and status of the operating system. It also monitors program activity from initiation until termination. (See “Host Resource Manager” on page 105 for details.)

Resource Manager Diagnostic Files

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

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

Minimal tracing can be activated with the command:

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

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

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

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

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

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

Audit Log Resource Manager

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

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

Audit Log Resource Class

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

The following attributes can be monitored for this resource class:

ResourceDefined
: Indicates that a subsystem definition has been added.

ResourceUndefined
: Indicates that a subsystem definition has been deleted.

ConfigChanged
: Indicates that a persistent resource class attribute has changed.

The following persistent resource attributes can be retrieved for instances of this resource class:

Name Identifies the name of the subsystem that will be adding entries to the audit log.

ResourceHandle
: An internally assigned handle that uniquely identifies the subsystem definition.

Variety
: Identifies which of the defined resource attributes and actions apply to the subsystem definition.

MessageCatalog
: Identifies the name of the message catalog for the subsystem that contains all audit log related information including format strings for records, descriptions of fields and records, and so on.

MessageSet
: Identifies the message set within the message catalog for this subsystem that contains all audit log related information including format strings for records, descriptions of fields and records, etc.

DescriptionId
: Identifies the message identifier in the message catalog for this subsystem that contains a description of the subsystem.

DescriptionText
: Contains text that describes the subsystem. This attribute is used by a client to retrieve the description of the subsystem. The text that is returned will be from the message catalog of the language that the requesting client is using. This text is obtained from the message catalog defined by the **MessageCatalog**, **MessageSet**, **DescriptionId** attributes and the client's current language.

RetentionPeriod
: Identifies how far back in time (in days) that records in the audit log for the subsystem will be retained. Any records which have a time field before the current time minus the retention period will be subject to automatic deletion from the audit log. If this value is zero, no records will be automatically removed based on their time field.

MaxSize
: Identifies the maximum size in Megabytes that the records for this

subsystem may occupy on disk. If the size exceeds this, records will be removed starting from the oldest until the total size of all records for the subsystem is less than the specified size.

SubsystemId
Identifies the subsystem identifier.

NodeIDs
Identifies the node that the audit log is on.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

Audit Log Template Resource Class

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

Configuration Resource Manager

The configuration resource manager (IBM.ConfigRM) is implemented as a resource manager within the RMC subsystem. It contains the following resource classes:

- The IBM.PeerDomain resource class which represents the RSCT peer domains to which a particular node is defined.
- The IBM.PeerNode resource class which represents fixed resources, one per node within the peer domain.
- The IBM.NetworkInterface resource class which represents the set of network interfaces that exist in the peer domain.
- The IBM.CommunicationGroup resource class which represents the set of communication resources upon which liveness checks can be performed.
- The IBM.RSCTParameters resource class which represents operational characteristics of the RSCT subsystems.

Peer Domain Resource Class

The program name for this resource class is IBM.PeerDomain. It represents the peer domains to which a particular node is defined. Each node has its own IBM.PeerDomain resource class. Each instance of this class represents an RSCT peer domain to which the node is defined. The number of instances in this resource class, therefore, indicates the number of peer domains to which the node is defined.

The resources of this class are somewhat different than other configuration resource manager resource classes since they span multiple peer domains while all other resources are contained in the context of a single peer domain.

This resource class has the following persistent class attribute:

OnlineDomain
Identifies the name of the domain that the node is currently online in. This is a NULL string if the node is not currently online in any domain.

The following persistent resource attributes can be retrieved for instances of the IBM.PeerDomain resource class.

Name The name of the peer domain.

ResourceHandle
An internally-assigned handle that uniquely identifies this resource.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

RSCTActiveVersion
Identifies the version of the RSCT software that is active in the peer domain. Some nodes may have a later version installed, but functionally they will operate at the minimum level existing on any defined nodes in the peer domain. This value is updated only after all nodes in the RSCT peer domain have the later version installed.

MixedVersions
Indicates whether there are different versions of the RSCT software installed on the nodes of the peer domain. If FALSE (0), then all nodes are at the same level. To determine the RSCT version installed on each node of the peer domain, refer to the IBM.PeerNode resource class. See "Peer Node Resource Class" on page 92 for more information about this class.

TSPort
Identifies the UDP port number that will be used by Topology Services for daemon to daemon communications within the peer domain.

GSPort
Identifies the UDP port number that will be used by Group Services for daemon to daemon communications within the peer domain.

RMCPort
Identifies the UDP port number that will be used by RMC for daemon to daemon communications within the peer domain.

ResourceClasses
The list of resource classes in the peer domain and their minimum level and version numbers. This list includes:

ClassName
The name of the resource class.

Id The ID of the resource class.

Version
The minimum level of the version of the resource class in the peer domain.

CSSKType
The Cluster Shared Secret Key (CSSK) type. This type is established by the system administrator when the peer domain is created. The CSSK key-type allows the system administrator to choose the appropriate balance of data protection and application performance. The longer the key, the stronger the encryption algorithm. The stronger the encryption algorithm, however, the slower the application performance. The key types are:

SEC_C_KEYTYPE_DES_MD5
The key digest is calculated using the MD5 hash and the encryption is performed using DES. The length of the signature is 16 bytes. Compared to the other key types, this key type provides a lesser degree of data protection, but greater performance with less data overhead.

SEC_C_KEYTYPE_3DES_MD5
The key digest is MD5 and the encryption is triple DES. The length of the signature is 16 bytes. Compared with the

SEC_C_KEYTYPE_DES_MD5 key type, this key type provides added data protection, slower performance, and the same data overhead.

SEC_C_KEYTYPE_AES256_SHA
The key digest is SHA and the encryption is AES 256 bit. The length of the signature is 24 bytes. Compared to the other key types, this key type provides the greatest data protection, but slower performance with greater data overhead.

CSSKRefreshInterval
Indicates the interval at which the configuration resource manager will refresh the CSSK in the peer domain.

AdminID
Indicates the user ID that is granted read and write authorization to all resource classes on all nodes.

The following dynamic resource attributes can be monitored for instances of the IBM.PeerDomain resource class.

OpState
Monitors the current operational state of the resource. Typical values for this state are Online and Offline.

ConfigChanged
Monitors whenever one or more persistent attribute values change.

CSSKLastUpdate
Indicates the time when the CSSK in the peer domain was last updated. Value is in seconds since UNIX epoch.

Peer Node Resource Class

The programmatic name for this resource class is IBM.PeerNode. It represents the nodes defined in the peer domain. A node is defined in this situation as an instance of an operating system, and is not necessarily tied to hardware boundaries.

The following persistent resource attributes can be retrieved for instances of the IBM.PeerNode resource class.

Name The name of the node.

ResourceHandle
An internally assigned handle that uniquely identifies the resource.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeList
This array contains one element that identifies the node on which the resource exists.

NodeID
A unique identifier for the node.

NodeNameList
Retrieves the same information as the **NodeID** attribute.

RSCTVersion
Identifies the version of the RSCT software that is installed on the node.

Some nodes may have a later version installed, but functionally they will operate at the minimum level existing on any defined nodes in the peer domain.

ClassVersions
An array indexed by resource class ID for the versions of the classes installed on the node.

PublicKey
The current public key associated with the node. This is used to provide security for remote operations to that node.

The following dynamic resource attributes can be monitored for instances of the IBM.PeerNode resource class.

OpState
Monitors the current operational state of the resource. Typical values for this state are Online and Offline.

ConfigChanged
Monitors whenever one or more persistent attribute values change.

Network Interface Resource Class

The programmatic name for this resource class is IBM.NetworkInterface. It represents the set of network interfaces that exist in the peer domain. Note that a network interface is not the same as a network device. A network device can host multiple network interfaces. Each resource instance in this class corresponds to an IP network interface. Each node may have one or more network interfaces, and one or more IP addresses may be assigned to a network interface.

The following persistent resource attributes can be retrieved for instances of the IBM.NetworkInterface resource class.

Name The name of the network interface.

ResourceHandle
An internally-assigned handle that uniquely identifies the network interface.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
A unique identifier for the node.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

DeviceName
Identifies the Network Device that hosts the network interface. If the operating system does not support this concept, this string will be NULL.

IPAddress
Identifies the base IP address (IPV4 or IPV6) for the network interface.

SubnetMask
Identifies the base subnet mask for the network interface.

Subnet
Identifies the base subnet for the network interface.

CommGroup
Identifies the name of the communication group to which this network interface is associated.

HeartbeatActive
Identifies whether the Topology Services “heartbeat” is active or not. 0 means it is inactive. 1 means it is active.

Aliases
Identifies all additional addresses that have been assigned to the interface. The following information is retrieved for each alias.

IPAddress
The IP address of the alias.

SubnetMask
The subnet mask for the alias.

Subnet
The base subnet for the alias.

DstAddress
Identifies the destination address for a point to point connection. This field is valid only if the Variety attribute has a value of 2 which indicates a point to point interface.

The following dynamic resource attributes can be monitored for instances of the IBM.NetworkInterface resource class.

OpState
Monitors the current operational state of the network interface. Typical values for this state are Online and Offline. If the Topology Services heartbeat is not active, this resource attribute value will be unknown.

ConfigChanged
Monitors whenever one or more persistent attribute values change.

Communication Group Resource Class

The programmatic name for this resource class is IBM.CommunicationGroup. It represents the set of communication resources among which liveness checks (Topology Services “heartbeating”) will be performed. A communication group resource identifies attributes that control the liveness checking between the set of network adapters and other devices in the group.

The following persistent resource attributes can be retrieved for instances of the IBM.CommunicationGroup resource class.

Name The name of the communication group.

ResourceHandle
An internally-assigned handle that uniquely identifies this communication group.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

Sensitivity
Identifies the number of missed heartbeats that constitute a failure.

Period
The number of seconds between heartbeats.

UseBroadcast
Indicates whether broadcast is used if it is supported by the underlying media.

UseSourceRouting
Indicates whether source routing is used if it is supported by the underlying media.

NIMPathName
The path name to the Network Interface Module (NIM) that supports the type of adapters in the communication group.

NIMParameters
The parameters that are passed to the NIM it is started.

Priority
Priority number indicating the importance of this communication group with respect to others. The lower the number, the higher the priority. The highest priority is 1.

The following dynamic resource attribute can be monitored for instances of the IBM.CommunicationGroup resource class.

ConfigChanged
Monitors whenever one or more persistent attribute values change.

RSCT Parameters Resource Class

The programmatic name for this resource class is IBM.RSCTParameters. It is used to represent operational characteristics of the RSCT subsystems.

The following persistent resource attributes can be retrieved for instances of the IBM.RSCTParameters resource class.

Variety
Identifies which of the defined class attributes and actions apply to this version of the resource class.

TSLogSize
Identifies the maximum number of lines that can be written in the log file used by the topology services daemon on each node.

TSFixedPriority
Identifies whether the topology services daemons should run with a fixed priority to avoid resource starvation, and, if so, the priority value it should run at. Valid values are:

-1 or 0
Do not use fixed priority.

>0 Use the value as the fixed priority.

TSPinnedRegions
Identifies which regions of the topology services daemon is pinned in memory. This value is a bit mask. Valid values are:

0 0x0000 Pin no region.

1 0x0001 Pin TEXT region.

2 0x0002 Pin DATA regions.

3 0x0003 Pin TEXT & DATA regions.

4 0x0004 Pin STACK regions.

5 0x0005 Pin TEXT & STACK regions.

6 0x0006 Pin DATA & STACK regions.

7 0x0007 Pin TEXT, DATA, & STACK regions.

GSLogSize
Identifies the maximum number of lines that can be written to the log file used by the group services daemons on each node.

GSMaxDirSize
Identifies the Group Services maximum directory in Kilobytes.

The following dynamic resource attributes can be monitored for instances of the IBM.RSCTParameters resource class.

ConfigChanged
Whenever a persistent attribute of the resource class changes, this dynamic attribute will be asserted.

Event Response Resource Manager

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

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

The Event Response resource manager (ERRM) is automatically started when the RMC subsystem is started.

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

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

Information is handled as follows:

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

ERRM contains the following three resource classes:

- The IBM.Condition resource class which contains the necessary information (event expression and rearm expression) for the ERRM to register with the RMC for event notifications.
- The IBM.EventResponse resource class which executes any number of configured commands when an event from an active IBM.Association resource occurs.
- The IBM.Association resource class which joins the IBM.Condition resource class together with the Event Response resource class.

Condition Resource Class

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

Configuration of ERRM begins with the definition of a set of Condition resources. A Condition resource is registered with the RMC subsystem when the Condition resource is used in the definition of an active Association resource, or its dynamic attribute EventOccurred is requested to be monitored.

Notes:

1. Registration with RMC is necessary for monitoring to run. Registration does not occur when a new Condition resource is defined, but rather when the resource is used in the definition of an active Association resource.
2. While monitoring a Condition on multiple nodes, if the RMC session with any one node is lost, the Condition's monitor status will be "monitored but in error."

The following dynamic resource attributes can be monitored for instances of this resource class:

ConfigChanged
: Monitors whenever one or more persistent attribute values change.

EventOccurred
: Indicates that an event has occurred for this condition. This is structured data that contains the following elements:

  Occurred
  : You can use the expression "EventOccurred.Occurred!=0" for monitoring "EventOccurred" Condition dynamic resource attribute.

  ErrNum
  : an error code returned from the RMC event notification. This element will contain a non-zero error number for an error event. It will be 0 for a non-error event.

  ErrMsg
  : the message returned by RMC to describe the error. It will be null for a non-error event.

  EventFlags
  : the same value returned from RMC mc_event_flags in the event notification. This element will contain valid information for a non-error event. It will be 0 for an error event.

EventTimeSec
The time the event occurred (in seconds). This element will contain valid information for a non-error event. It will be 0 for an error event.

EventTimeUsec
The time the event occurred (in microseconds). This element will contain valid information for a non-error event. It will be 0 for an error event.

RsrcHndl
The resource handle of the resource whose state change caused the generation of this event. This element will contain valid information for every type of event.

DynAttrDataType
RMC ct_data_type_t of the dynamic attribute that changed to cause the generation of this event. This element will contain valid information for a non-error event. It will be 0 for an error event.

RsrcName
The name of the resource whose dynamic attribute changed to cause this event. This element will contain valid information for every type of event.

NodeName
The node name returned from RMC. This is the node where the resource was being monitored. This element will contain valid information for any type of event.

DynAttrValue-*n*
The value of the dynamic attribute that caused the event to occur. If the data type of the dynamic attribute that caused the event to occur is not the CT_SD_PTR, there will be only one element (DynAttrValue-1) to represent the value of the dynamic attribute. If the data type of the dynamic attribute that caused the event to occur is the CT_SD_PTR, there will be multiple elements DynAttrValue-1, DynAttrValue-2, DynAttrValue-3... to represent the value of each element sequentially in the dynamic attribute's SD.

MonitorStatus
Indicates the monitor status of this condition. This is structured data that contains the following elements:

Status
A bit mask indicating the Condition's monitoring status.

Bit 0 The Condition is being monitored because it is associated with an active Association.

Bit 1 The Condition is being monitored because its dynamic resource attribute "EventOccurred" is requested to be monitored by a user.

Bit 2 The Condition is currently being monitored but an error has occurred. The bit will be set when the Resource Manager has a failure on one of the nodes that Condition is being monitored on, and RMC cannot monitor the Condition on that node.

Bit 3 The Condition is set to be monitored because it is

associated with an active Association. However, it cannot be monitored because of errors in its definition.

Bit 4 The Condition is set to be monitored because its dynamic resource attribute ″EventOccurred″ is requested to be monitored by a user. However, it cannot be monitored because of errors in its definition.

NodeNames
When the **MonitorStatus** attribute indicates that the Condition is currently being monitored but an error has occurred (bit 2 is set), this attribute stores a list of node names. This element will be null if the **MonitorStatus** bit 2 is not set.

ErrCodes
Stores a list of error code corresponding to **NodeNames**. Currently only error code = 1 is defined to indicate the Resource Manager on that specific node has a failure.

The following persistent resource attributes can be retrieved for instances of the IBM.Condition resource class:

ResourceHandle
An internally-assigned handle that uniquely identifies this Condition.

Name The name of the condition.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Unique identifiers for the nodes.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

ResourceClass
The name of the resource class of which the **DynamicAttribute** being monitored by this Condition is a member.

DynamicAttribute
The name of the dynamic attribute that will be submitted to RMC for monitoring.

EventExpression
The exact text to be submitted to RMC for monitoring which describes when an event should be generated.

EventDescription
Printable text assigned by the creator which describes the Condition that is being monitored.

RearmExpression
The exact text to be submitted to RMC to determine when monitoring should start again after this Condition generated an event.

RearmDescription
Printable text assigned by the creator which describes when monitoring should start again after this Condition generated an event.

SelectionString
The exact text to be submitted to RMC to limit which resources should be included in the monitoring.

ImmediateEvaluate
This is set if the **EventExpression** should be evaluated by RMC when the event registration is done.

Severity
A value assigned by the creator to describe the importance of this Condition compared to other Conditions. (0 is Informational, 1 is Warning, and 2 is Critical)

ManagementScope
The monitoring scope for this Condition.

MC_SESSION_OPTS_LOCAL_SCOPE
Connects to the RMC session with option for this scope.

MC_SESSION_OPTS_SR_SCOPE
Connects to the RMC session with option for this scope.

MC_SESSION_OPTS_DM_SCOPE
Connects to the RMC session with option for this scope.

NodeNames
A list of names that the Condition will be monitored on. The name can be a node name or a group name.

Event Response Resource Class

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

Predefined responses are available to use and to serve as templates for creating your own responses. For a description of predefined responses, see “What is a Response?” on page 36. Scripts for notification and logging of events and for broadcasting messages to logged-in user consoles are provided in the *Reliable Scalable Cluster Technology for AIX: Technical Reference*.

Note: Commands are run in parallel.

The following dynamic resource attribute can be monitored for instances of this resource class:

ConfigChanged
Monitors whenever one or more persistent attribute values change.

The following persistent resource attributes can be retrieved for instances of this resource class:

ResourceHandle
An internally-assigned handle that uniquely identifies this event response.

Name The name of the event response.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Unique identifiers for the nodes.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

Actions
The list of actions (which includes Command resources) associated with this **EventResponse** resource. Every element within the structured data is another individual Action. NULL means no action will be executed when an event occurs. Each listed action has the following elements:

ActionName
The name of the action.

WeekDay
This is a bit mask which indicates which days of the week the command for this Action should be executed. If the value is zero, the command will be executed on all days.

Bit 0 Sunday

Bit 1 Monday

Bit 2 Tuesday

Bit 3 Wednesday

Bit 4 Thursday

Bit 5 Friday

Bit 6 Saturday

StartTime
If the time when a Condition occurs is before the **StartTime**, the command for this Action will not be executed. This value represents the number of seconds past midnight.

EndTime
If the time when a Condition occurs is after the **EndTime**, the command for this Action will not be executed. If both the **StartTime** and **EndTime** are non-zero, this value represents the number of seconds past midnight and it must be greater than the **StartTime**.

Command
The command string that will be executed including the directory of the command, the command name and any command options.

EventType
This is a bit mask that determines which types of events will cause this command to be executed.

Bit 0 Arm event

Bit 1 ReArm event

StandardOutFlag
If this is set, the standard output from the command will be written to the Audit Log.

ReturnCode
The expected successful return code for the command.

CheckReturnCode

0 The ReturnCode is not to be checked after the command has been executed.

1 The ReturnCode is checked after the command has been executed.

EnvList
A list of environment variables (in the format *VariableName*=*VariableValue*) to be set before running the command.

UndefResFlag

0 Do not execute the command if the event is caused by a undefined resource.

1 Should still execute the command though the event is caused by a undefined resource.

Association Resource Class

The Association resource class joins the Condition resource class together with the Event Response resource class. It contains a flag that indicates whether the association between the condition and the event response is active. Event Responses and Conditions are separate entities, but for monitoring to take place, they need to be associated. An event cannot occur unless at least one Event Response is associated with a Condition. You can configure one or more actions for an Event Response, and one or more Event Responses for a Condition.

The following dynamic resource attribute can be monitored for instances of this resource class:

ConfigChanged
Monitors whenever one or more persistent attribute values change.

The following persistent resource attributes can be retrieved for instances of this resource class:

ResourceHandle
An internally-assigned handle that uniquely identifies this association.

Name The name of the association.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Unique identifiers for the nodes.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

ActiveFlag
Indicates whether or not the association is active. If active, the Condition will register with RMC and the associated **EventResponse** resource will execute the configured commands when the Condition occurs.

0 inactive

1 active

ConditionHandle
The resource handle of a Condition which will be used for RMC registration when **ActiveFlag** is set to be active

EventResponseHandle
The resource handle of an **EventResponse** which will be used to execute actions when the Condition event occurs.

File System Resource Manager

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

- List all file systems within the system.
- List only the file systems that match certain criteria.
- Obtain the status of a file system (mounted or unmounted).
- Obtain the values of the persistent attributes of the file system.
- Monitor the percentage of disk space used for the file system.
- Monitor the percentage of i-nodes used for the file system.
- Mount a resource (file system) using online() function.
- Unmount a resource (file system) using the offline() or reset() functions.

There is one File System resource manager (FSRM) on a node. It is started implicitly by the RMC subsystem.

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

Information is handled as follows:

- Files that contain internal trace output that is useful to a software service organization in resolving problems are written to **/var/ct/IW/log/mc/IBM.FSRM**.
- Persistent attributes and other information are stored in tables under a directory that corresponds to the resource class.
- Core files are written to the **/var/ct/IW/run/mc/IBM.FSRM** directory.

Filesystem Resource Class

The programmatic name for this resource class is IBM.Filesystem. The following dynamic resource attributes can be monitored for instances of this class.

These attributes of a file system resource can be monitored:

ConfigChanged
Monitors whenever one or more persistent attribute values change.

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

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

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

The following persistent resource attributes can be retrieved for instances of this resource class:

Name Identifies the name of the file system. This name is the same as the mount point defined in **/etc/filesystems**. Some examples are “/”, “/usr”, etc.

ResourceHandle
An internally assigned handle that uniquely identifies the file system.

Variety
Identifies which of the defined resource attributes and actions apply to the file system.

NodeIDs
Specifies the set of nodes upon which the operational interface of a resource is available. This attribute contains node Ids instead of node numbers as in **NodeList**. The **NodeIDs** attribute is implicitly mapped to attribute **NodeNameList** by RMC.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

ResourceType
Identifies the classification of resource. Possible values are Fixed, Floating, and Concurrent.

MountPoint
The directory over which a file system will be mounted as defined in **/etc/filesystems**.

MountDir
Identifies the actual mount point over which the file system is mounted. This may be the same value as the attribute **MountPoint**. For example, a record in file **/etc/filesystems** indicates that the device **/dev/sda7** will be automatically mounted at a mount point **/home**. Later, the administrator unmounts this file system **/home** and mounts it to a directory **/guest**. This actual mount point **/guest** is identified by the **MountDir** attribute. For the same resource, the **Name** and **MountPoint** attributes are still **/home**.

Dev Identifies the device name.

Vfs Virtual File System. Identifies the type of file system. Some examples are *jfs*, *jfs2*, *ext2*.

Permissions
Identifies the permission of the file system (**rw** or **ro**).

size Identifies the size of the file system in terms of 512-byte blocks.

Log Identifies the log logical volume name. This is only valid for journaled file system.

Mount This attribute is used by the **mount** command to determine if it should be mounted automatically.

Account
Used by the **dodisk** command to determine the file systems to be processed by the accounting system.

Type Used to group related mounts.

Frag Identifies the JFS fragment size in bytes.

Nbpi Identifies the number of bytes per I-Node (nbpi).

Compress
Identifies data compression.

Bf Identifies a large file enabled file system.

Ag Identifies the allocation group size in megabytes.

AgBlkSize
Identifies the JFS2 block size in bytes.

ManualMode
Identifies the manual mode.

Predefined Conditions for Monitoring File Systems

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

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

Host Resource Manager

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

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

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

Information is handled as follows:

- Files that contain internal trace output which is useful to a software service organization in resolving problems are written to **/var/ct/IW/log/mc/IBM.HostRM**.
- Persistent attributes and other information are stored in tables under a directory that corresponds to the resource class.
- Core files are written to the **/var/ct/IW/run/mc/IBM.HostRM** directory.

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

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

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

Host (IBM.Host)
This resource class externalizes the attributes of a machine that is running a single copy of an operating system. Primarily the attributes included are those that are advantageous in predicting or indicating when corrective action needs to be taken. See “Host Resource Class” on page 107 for more details.

Paging Device (IBM.PagingDevice)
This resource class externalizes the attributes of paging devices. See “Paging Device Resource Class” on page 117 for more details.

Physical Volume (IBM.PhysicalVolume)
This resource class externalizes many attributes of disks. See “Physical Volume Resource Class” on page 120 for more details.

Processor (IBM.Processor)
This resource class externalizes the attributes of individual processors, such as idle time. See “Processor Resource Class” on page 118 for more details.

Host Public (IBM.HostPublic)
This resource class gives information on the local host’s identifier token taken from the key files.

Program (IBM.Program)
This resource class allows a client to monitor attributes of a program that is running on a host. The program to monitor is identified by attributes such as program name, arguments, etc. The resource class does not monitor processes as such because processes are very transient and therefore inefficient to monitor individually. See “Program Resource Class” on page 125 for more details.

Each type of adapter that is supported has its own resource class as follows:

ATM Device (IBM.ATMDevice)
All ATM adapters installed in a node are externalized through this resource manager. See “ATM Device Resource Class” on page 122 for more details.

Ethernet Device (IBM.EthernetDevice)
All Ethernet adapters installed in a node are externalized through this resource manager. See “Ethernet Device Resource Class” on page 122 for more details.

FDDI Device (IBM.FDDIDevice)
All FDDI adapters installed in a node are externalized through this resource manager. See “FDDI Device Resource Class” on page 124 for more details.

Token-Ring Device (IBM.TokenRingDevice)
All Token-Ring adapters installed in a node are externalized through this resource manager. See “Token-Ring Device Resource Class” on page 124 for more details.

Host Resource Class

The programmatic name of this resource class is IBM.Host. The following persistent resource attributes can be retrieved for instances of the IBM.Host resource class.

Name Identifies the current name of the host as returned by the “hostname” command.

ResourceHandle
An internally assigned handle that uniquely identifies this host.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Specifies the set of nodes upon which the operational interface of a resource is available.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

NumProcessors
Indicates the number of processors installed in the system.

RealMemSize
Specifies the current size of physical memory in bytes.

OsName
This attribute reflects the name of the operating system running on the node.

OsVersion
This attribute reflects the version of the operating system or kernel running on the node.

DistributionName
This attribute reflects the name of the software distribution that is installed on the node. This is mainly applicable to the Linux implementation of RSCT.

DistributionVersion
This attribute reflects the version of the software distribution that is installed on a node.

MachineType
This attribute reflects generic type of machine the node is (for example, i386, s390, ppc, and so on). In some sense, it is more an indication of the instruction set that is running on the node.

The IBM.Host resource class allows the following resources of a host system to be monitored:

1. Processes in the run queue of the operating system scheduler (see “Monitoring the Operating System Scheduler”).
2. Global state of active paging spaces (see “Monitoring the Global State of Active Paging Space”).
3. Total processor utilization across all active processors in the system (see “Monitoring Processor Utilization” on page 110).
4. Real, virtual, and kernel memory utilization (see “Memory Management” on page 111).

Monitoring the Operating System Scheduler: The operating system scheduler maintains a run queue of all of the processes that are ready to be dispatched. Each second, the process table is scanned to determine which processes are ready to run. If one or more processes are ready, they are placed on the run queue, and a counter is incremented. The counter is used to compute the value of the **ProcRunQueue** variable as the average number of ready-to-run processes. The scheduler also scans the process table for processes that are inactive because they are waiting to be paged in. A swapped process may (or may not) have some or all of its pages moved to the swap (page) device. As with the **ProcRunQueue** variable, the system increments a counter for swapped processes, which is used to compute the value of the **ProcSwapQueue** variable as the average number of processes swapped out. A process must be paged in and marked non-swapped before it can be placed on the run queue. These attributes can be monitored:

ProcRunQueue
Average number of processes that are waiting for the processor.

ProcSwapQueue
Average number of processes that are waiting to be paged in.

Predefined Conditions for Monitoring the Operating System Scheduler: The following table shows the predefined conditions that are available for monitoring the operating system scheduler, and example expressions:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Processes in run queue	(ProcRunQueue - ProcRunQueue@P) >= (ProcRunQueue@P * 0.5)	An event is generated each time the average number of processes on the run queue has increased by 50% or more between observations.	ProcRunQueue < 50	The event is rearmed when the run queue length drops below 50.
Processes in swap queue	(ProcSwapQueue > 50) && (ProcSwapQueue@P > 50)	An event is generated each time two consecutive observations find 50 processes or more in the swap queue.	(ProcSwapQueue < 40) && (ProcSwapQueue@P < 40)	The event is rearmed when the number of processes in the swap queue drops below 40 for two consecutive observations.

Monitoring the Global State of Active Paging Space: A paging space is fixed disk storage for information that is resident in virtual memory but is not currently

being accessed. A paging space, or swap space, is a logical volume with the attribute type equal to paging. When the amount of free real memory in the system is low, programs or data that have not been used recently are moved from real memory to paging space to release real memory for other processes. The amount of paging space required depends upon the types of activities performed on the system. If paging space runs low, processes may be lost, and if paging space runs out, the system may panic. Paging-space shortage may cause memory performance degradation, and thrashing can occur (if VMM memory load control is turned off).

The system monitors the number of free paging-space blocks and detects when a paging-space shortage exists. When the number of free paging-space blocks falls below a threshold known as the paging-space warning level, the system informs all processes except kernel processes (kprocs) of this condition by sending the SIGDANGER signal. If the shortage continues and falls below a second threshold known as the paging-space terminate level, the system sends the SIGKILL signal to processes that are the major users of paging space and that do not have a signal handler for the SIGDANGER signal.

The warning-level and terminate-level thresholds can be obtained and altered by the command **vmtune** (*npswarn* and *npskill* parameters, respectively). Processes running in the early allocation environment avoid receiving the SIGKILL signal if a low paging space condition occurs. If the PSALLOC environment variable is set to early when a program starts, paging space is reserved at the time the process makes a memory request. If there is insufficient paging space, the early allocation algorithm used by the operating system causes the memory request to be unsuccessful. If the PSALLOC environment is not set, or is set to any value other than early, the operating system uses a late allocation algorithm for memory and paging-space allocation. Late allocation does not reserve paging space at the time the memory is requested but defers the reservation until the pages are touched.

Note: The VMM is a complex system, and paging-space requirements depend on a number of factors, including the paging-space allocation policy used, amount of real memory, and type of activities performed on the system. A thorough understanding of system paging requirements and operating system memory management is recommended before attempting to alter VMM operating parameters.

These attributes monitor the global state of all active paging spaces defined in the system (including NFS-mounted paging spaces):

TotalPgSpSize
: Holds the total size of all active paging-space devices in the system.

TotalPgSpFree
: Represents the size (in 4KB pages) of available paging space for all active paging space devices in the system.

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

PctTotalPgSpFree
: Represents the percentage of free paging space available for all paging space devices in the system.

Predefined Conditions for Monitoring Global State of Active Paging Space:
The following table shows the predefined conditions that are available for monitoring paging space, and example expressions:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Paging active space	TotalPgSpSize != TotalPgSpSize @P	An event is generated whenever the total amount of active paging space changes.	None	None
Paging free space	TotalPgSpFree<= 2560	An event is generated when the VMM is within 2MB (512 4KB pages) of reaching the paging space warning level.	TotalPgSpFree > 2560	The event is rearmed when the free paging space total becomes greater than the same threshold.
Paging percent space used	PctTotalPgSpUsed > 90	An event is generated when more than 90% of the total paging space is in use.	PctTotalPgSpUsed < 85	The event is rearmed when the percentage falls below 85%.
Paging percent space free	PctTotalPgSpFree< 10	An event is generated when the total amount of free paging space falls below 10%.	PctTotalPgSpFree > 15	The event is rearmed when the free paging space increases to 15%.

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

The idle and wait states of a processor are monitored, and the time spent running in protection mode is monitored. At each clock tick, an array of counters is incremented to reflect processor activity based on the state of the current running processes. The **PctTotalTimeKernel**, **PctTotalTimeUser**, **PctTotalTimeWait**, and **PctTotalTimeIdle** attributes provide the approximate average percentage of time all active processors are currently spending in each state. Therefore, the sum of these values is 100 at any given observation.

There are two protection modes that processes run in, kernel (or system) level and user level. Processes running in kernel mode run with kernel privileges and have access to kernel data. These processes include kernel processes (kprocs) and services (such as system calls and device drivers).

Processes running in user mode are normal applications with user level privileges and run in their own unique process space. When a user level process invokes a kernel service, for example, by making a system call, a mode switch occurs that causes the process to run in kernel mode while the service is running.

When the current running process makes a request that cannot be immediately satisfied, such as an I/O operation, the process is put into wait state. A processor is considered idle when the current running process is the *wait* process. The wait process is a kernel process (kproc) that is dispatched when no other processes are ready to run.

These attributes can be monitored:

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

PctTotalTimeKernel
: Represents the system-wide percentage of time that the processors are running in kernel mode.

PctTotalTimeUser
Represents the system-wide percentage of time that the processors are running in user mode

PctTotalTimeWait
Represents the system-wide percentage of time that the processors are in wait state.

Predefined Conditions for Monitoring Processor Utilization: The following table shows the predefined conditions that are available for monitoring system-wide processor idle time, and example expressions:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Processor idle time	PctTotalTimeIdle>= 70	An event is generated when the average time all processors are idle at least 70% of the time.	PctTotalTimeIdle < 10	The event is rearmed when the idle time decreases below 10%.
Processor kernel time	PctTotalTimeKernel>= 70	An event is generated when the average time all processors are running in kernel mode is at least 70% of the time.	PctTotalTimeKernel < 10	The event is rearmed when the kernel time decreases below 10%.
Processor user time	PctTotalTimeUser>= 70	An event is generated when the average time all processors are running in user mode is at least 70% of the time.	PctTotalTimeUser< 10	The event is rearmed when the user time decreases below 10%.
Processor wait time	PctTotalTimeWait >= 50	An event is generated when the average time all processors are waiting on I/O is at least 50% of the time.	PctTotalTimeWait < 10	The event is rearmed when the wait time decreases below 10%.

Memory Management: The VMM (Virtual Memory Manager) manages the allocation of real memory page frames, resolves references to virtual memory pages that are not currently in real memory (or do not yet exist), and manages the reading and writing of pages to disk storage.

The VMM maintains a list of free page frames that it uses to accommodate page faults. A page fault occurs when a page that is not in real memory is referenced. In most environments, the VMM must occasionally add to the free list by reassigning some page frames owned by running processes. The virtual-memory pages whose page frames are to be reassigned are selected by the VMM's page-replacement algorithm, which takes into consideration the segment type, statistics regarding rate of reoccurring page faults, and user-tunable thresholds. The number of frames reassigned to the free list is also determined by VMM thresholds.

Memory regions defined in either system or user space may be pinned. Pinning a memory region prohibits the pager from stealing pages from the pages backing the pinned memory region. After a memory region is pinned, accessing that region does not result in a page fault until the region is subsequently unpinned. While a portion of the kernel remains pinned, many regions are pageable and are only pinned while being accessed.

Thresholds used by the VMM include the minimum and maximum number of pages to be maintained on the free list (*minfree* and *maxfree*). These thresholds are used to determine when the VMM should start or stop stealing pages to replenish the

free list. There is also a maximum percentage of real memory that may be pinned. The values of these thresholds may be queried or altered using the system command **vmtune**.

Virtual memory is partitioned into fixed-size units called pages. Each page may be in real memory (RAM) or stored on disk until needed. Real memory is partitioned into units that are equal in size to virtual pages and are referred to as page frames. To accommodate a large virtual memory space with a limited real memory space, the system uses real memory for work space and maps inactive data and programs to disk.

Pages of a virtual address space are considered to be persistent or working. Persistent pages have permanent storage locations on disk. Data files or programs are mapped to persistent pages. Since persistent pages have a permanent storage location, the VMM can write a changed page back to its permanent location or simply free the page frame if it was not altered and re-read the page on a subsequent request.

Working pages are transitory and exist only during their use by a process. Examples are process stack and data regions, kernel and kernel-extension text regions, and shared-library text and data regions. Working pages also require disk storage locations when they cannot be kept in real memory. Disk paging space is used for this purpose.

The operating system provides routines used by the kernel and by services running at system level for allocating memory in kernel space. Counters are maintained in the kernel to track requests and use of kernel memory, based on the type of data structure or service. These attributes can be used to monitor the number and size and the state of requests for buffers allocated in kernel memory. The types of kernel memory available are:

- Mbuf (network data buffer)
- Socket (kernel socket structure)
- Protcb (protocol control block)
- OtherIP (other buffers used by IP)
- Mblk (stream header and data)
- Streams (other streams-related memory)
- Other (other kernel memory).

The following attributes are available for monitoring real and virtual memory and kernel memory. The <x> in the names below refers to the type of kernel memory allocation as shown in the preceding list (28 possible monitors).

PctRealMemFree
: Represents the percentage of real page frames that are currently available on the VMM free list.

PctRealMemPinned
: Represents the percentage of real page frames that are currently pinned and cannot be paged out.

RealMemFramesFree
: Represents the number of real page frames that are currently available on the VMM free list.

VMPgInRate
: Represents the rate (in pages per second) that the VMM is reading both persistent and working pages from disk storage.

VMPgOutRate
Represents the rate (in pages per second) that the VMM is writing both persistent and working pages to disk storage.

VMPgFaultRate
Represents the average rate of page faults that occur per second.

VMPgSpInRate
Represents the rate (in pages per second) that the VMM is reading working pages from paging-space disk storage.

VMPgSpOutRate
Represents the rate (in pages per second) that the VMM is writing working pages to paging-space disk storage.

KMemReq<x>Rate
Represents the rate of requests per second for a kernel memory buffer of type <x>.

KMemFail<x>Rate
Represents the rate of requests per second for a kernel memory buffer of type <x> that were unsuccessful.

KMemNum<x>
Represents the number of kernel memory buffers of type <x> that are currently in use.

KMemSize<x>
Represents the amount, in bytes, of kernel memory buffers of type <x> that are currently in use.

VMActivePageCount
Represents the total number of virtual memory pages that are being accessed by all running processes. It does not include pages used by the kernel or file systems.

PctRealMemActive
Represents the percentage of real memory pages that are needed to accommodate the set of active virtual memory pages for all running processes. This value does not include those pages in use by the kernel or file systems.

LoadAverage An array containing three entries which contain the number of jobs in the run queue averaged over 1, 5, and 15 minutes.

NumUsers Represents the number of users that are currently logged on to the system.

UpTime Represents the number of seconds since the system was last booted.

ActiveMgtScopes
Represents the set of management scopes that are active on the node. A management scope is a concept implemented by the RMC subsystem that controls the set of nodes to which RMC operations will potentially have an effect. One or more scopes may be active at the same time on a node. Each active scope is represented by a bit in the value of this attribute. The values corresponding to each scope are:

1 Local

2 Peer Domain

Predefined Conditions for Memory Management: The following table shows the predefined conditions that are available for monitoring memory management, and example expressions:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Real memory free	PctRealMemFree < 5	An event is generated when the percentage of real page frames that are free falls below 5%.	PctRealMemFree> 10	The event is rearmed when the percentage of free frames exceeds 10%.
Real memory pinned	PctRealMemPinned > 75	An event is generated when the percentage of real page frames that are pinned exceeds 75%.	PctRealMemPinned < 70	The event is rearmed when the percentage falls below 70%.
Real memory free frames	PctMemFramesFree < 120	An event is generated when the number of free real page frames falls below 120.	PctMemFramesFree> 150	The event is rearmed when the number free exceeds 150.
Page in rate	VMPgInRate > 500	An event is generated when the rate of pages read by the VMM for both persistent and working pages exceeds 500 per second.	VMPgInRate < 400	The event is rearmed when the rate drops below 400.
Page out rate	VMPgOutRate > 500	An event is generated when the rate of pages written by the VMM for both persistent and working pages exceeds 500 per second.	VMPgOutRate < 400	The event is rearmed when the rate drops below 400.
Page fault rate	VMPgFaultRate > 500	An event is generated when there are more than 500 page faults per second.	VMPgFaultRate < 400	The event is rearmed when the rate drops to less than 400 pages per second.
Page space in rate	VMPgSpInRate > 500	An event is generated when more than 500 pages per second are read by the VMM from paging space devices (working pages only).	VMPgSpInRate< 400	The event is rearmed when the rate drops to less than 400 pages per second.
Page space out rate	VMPgSpOutRate> 500	An event is generated when more than 500 pages per second are written by the VMM to paging space devices (working pages only).	VMPgSpOutRate < 400	The event is rearmed when the rate drops to less than 400 pages per second.
Kernel Mbuf rate	KMemReqMbufRate> 5000	An event is generated when the number of requests for a kernel buffer of type <Mbuf> (network data buffer) exceeds 5000 per second.	KMemReqMbufRate< 4000	The event is rearmed when the rate falls below 4000 per second.
Kernel socket buffer rate	KMemReqSockRate > 5000	An event is generated when the number of requests for a kernel buffer of type <Socket> (kernel socket structure) exceeds 5000 per second.	KMemReqSockRate < 4000	The event is rearmed when the rate falls below 4000 per second.

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Kernel protocol CB rate	KMemReqProtcbRate> 5000	An event is generated when the number of requests for a kernel buffer of type <Protcb> (Protocol Control Block) exceeds 5000 per second.	KMemReqProtcbRate < 4000	The event is rearmed when the rate falls below 4000 per second.
Kernel other IP CB rate	KMemReqOtherIPRate > 5000	An event is generated when the number of requests for a kernel buffer of type <OtherIP> (other buffers used by IP) exceeds 5000 per second.	KMemReqOtherIPRate < 4000	The event is rearmed when the rate falls below 4000 per second.
Kernel Mblk rate	KMemReqMblkRate > 5000	An event is generated when the number of requests for a kernel buffer of type <Mblk> (stream header and data) exceeds 5000 per second.	KMemReqMblkRate< 4000	The event is rearmed when the rate falls below 4000 per second.
Kernel streams buffer rate	KMemReqStreamsRate > 5000	An event is generated when the number of requests for a kernel buffer of type <Streams> (other streams related memory) exceeds 5000 per second.	KMemReqStreamsRate < 4000	The event is rearmed when the rate falls below 4000 per second.
Kernel other memory rate	KMemReqOtherRate > 5000	An event is generated when the number of requests for a kernel buffer of type <Other> (other kernel memory) exceeds 5000 per second.	KMemReqOtherRate < 4000	The event is rearmed when the rate falls below 4000 per second.
Kernel Mbuf failed rate	KMemFailMbufRate > 10	An event is generated when the number of failures of requests for a kernel buffer of type <Mbuf> (network data buffer) exceeds 10 per second.	KMemFailMbufRate < 5	The event is rearmed when the rate falls below 5 per second.
Kernel socket buffer failed rate	KMemFailSockRate> 10	An event is generated when the number of failures of requests for a kernel buffer of type <Socket> (kernel socket structure) exceeds 10 per second.	KMemFailSockRate< 5	The event is rearmed when the rate falls below 5 per second.
Kernel protocol CB failed rate	KMemFailProtcbRate > 10	An event is generated when the number of failures of requests for a kernel buffer of type <Protcb> (Protocol Control Block) exceeds 10 per second.	KMemFailProtcbRate < 5	The event is rearmed when the rate falls below 5 per second.
Kernel other IP CB failed rate	KMemFailOtherIPRate> 10	An event is generated when the number of failures of requests for a kernel buffer of type <OtherIP> (other buffers used by IP) exceeds 10 per second.	KMemFailOtherIPRate < 5	The event is rearmed when the rate falls below 5 per second.

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Kernel Mblk failed rate	KMemFailMblkRate> 10	An event is generated when the number of failures of requests for a kernel buffer of type <Mblk> (stream header and data) exceeds 10 per second.	KMemFailMblkRate< 5	The event is rearmed when the rate falls below 5 per second.
Kernel streams buffer failed rate	KMemFailStreamsRate> 10	An event is generated when the number of failures of requests for a kernel buffer of type <Streams> (other stream related memory) exceeds 10 per second.	KMemFailStreamsRate < 5	The event is rearmed when the rate falls below 5 per second.
Kernel other memory failed rate	KMemFailOtherRate> 10	An event is generated when the number of failures of requests for a kernel buffer of type <Other> (other kernel memory) exceeds 10 per second.	KMemFailOtherRate < 5	The event is rearmed when the rate falls below 5 per second.
Kernel Mbufs	KMemNumMbuf > 10000	An event is generated when the allocated number of kernel buffers of type <Mbuf> (network data buffer) exceeds 10000.	KMemNumMbuf < 9000	The event is rearmed when the number falls below 9000.
Kernel socket buffers	KMemNumSock > 10000	An event is generated when the allocated number of kernel buffers of type <Socket> (kernel socket structure) exceeds 10000.	KMemNumSock< 9000	The event is rearmed when the number falls below 9000.
Kernel protocol CBs	KMemNumProtcb> 10000	An event is generated when the allocated number of kernel buffers of type <Protcb> (Protocol Control Block) exceeds 10000.	KMemNumProtcb< 9000	The event is rearmed when the number falls below 9000.
Kernel other IP CBs	KMemNumOtherIP> 10000	An event is generated when the allocated number of kernel buffers of type <OtherIP> (other buffers used by IP) exceeds 10000.	KMemNumOtherIP< 9000	The event is rearmed when the number falls below 9000.
Kernel Mblk buffers	KMemNumMblk> 10000	An event is generated when the allocated number of kernel buffers of type <Mblk> (stream header and data) exceeds 10000.	KMemNumMblk < 9000	The event is rearmed when the number falls below 9000.
Kernel stream buffers	KMemNumStreams> 10000	An event is generated when the allocated number of kernel buffers of type <Streams> (other streams related memory) exceeds 10000.	KMemNumStreams< 9000	The event is rearmed when the number falls below 9000.

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Kernel other memory	KMemNumOther > 10000	An event is generated when the allocated number of kernel buffers of type <Other> (other kernel memory) exceeds 10000.	KMemNumOther < 9000	The event is rearmed when the number falls below 9000.
Kernel Mbufs size	KMemSizeMbuf> 0x4000000	An event is generated when the total space occupied by kernel buffers of type <Mbuf> (network data buffer) exceeds 64MB.	KMemSizeMbuf < 0x2000000	The event is rearmed when the allocated amount drops below 32MB.
Kernel socket buffers size	KMemSizeSock> 0x4000000	An event is generated when the total space occupied by kernel buffers of type <Socket> (kernel socket structure) exceeds 64MB.	KMemSizeSock < 0x2000000	The event is rearmed when the allocated amount drops below 32MB.
Kernel protocol CBs size	KMemSizeProtcb > 0x4000000	An event is generated when the total space occupied by kernel buffers of type <Protcb> (Protocol Control Block) exceeds 64MB.	KMemSizeProtcb< 0x2000000	The event is rearmed when the allocated amount drops below 32MB.
Kernel other IP CBs size	KMemSizeOtherIP> 0x4000000	An event is generated when the total space occupied by kernel buffers of type <OtherIP> (other buffers used by IP) exceeds 64MB.	KMemSizeOtherIP< 0x2000000	The event is rearmed when the allocated amount drops below 32MB.
Kernel Mblks size	KMemSizeMblk > 0x4000000	An event is generated when the total space occupied by kernel buffers of type <Mblk> (stream header and data) exceeds 64MB.	KMemSizeMblk < 0x2000000	The event is rearmed when the allocated amount drops below 32MB.
Kernel streams buffers size	KMemSizeStreams > 0x4000000	An event is generated when the total space occupied by kernel buffers of type <Streams> (other streams related memory) exceeds 64MB.	KMemSizeStreams < 0x2000000	The event rearmed when the allocated amount drops below 32MB.
Kernel other memory size	KMemSizeOther > 0x4000000	An event is generated when the total space occupied by kernel buffers of type <Other> (other kernel memory) exceeds 64MB.	KMemSizeOther < 0x2000000	The event is rearmed when the allocated amount drops below 32MB.

Paging Device Resource Class

The program name of this resource class is IBM.PagingDevice. It can be used to monitor devices that are used by the operating system for paging. Each host may have one or more paging devices. On the operating system, the paging device is a logical volume.

The following persistent resource attributes can be retrieved for instances of the IBM.PagingDevice resource class.

Name Identifies the name of the paging space device.

ResourceHandle
An internally assigned handle that uniquely identifies a paging space device.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Specifies the set of nodes upon which the operational interface of a resource is available.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

Size Identifies the size of the paging device in terms of 4K pages.

Monitoring Amount of Free Paging Space for Device: These attributes can be monitored:

OpState Monitors whether the current operational state of the page device is online or offline.

PctFree Represents the percentage of free paging space available for a specific paging space device.

Predefined Conditions for Monitoring Paging Space for a Specific Device: The following table shows the predefined conditions and examples of expressions that are available for monitoring paging space for a specific device:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Paging device state	OpState != 1	An event is generated when the paging space device goes offline.	OpState == 1	The event is rearmed when the device comes back online.
Paging device percent free	PctFree < 20	An event is generated when less that 20% of the paging device is free.	PctFree > 25	The event is rearmed when the amount of free paging space on the device exceeds 25%.

Processor Resource Class

The programmatic name of this resource class is IBM.Processor. Because the system tracks the amount of time each processor spends idle, in wait state, and running in kernel and user modes, this resource class can be used to monitor these processor activities. At each clock tick, an array of counters is incremented to reflect the processor activity based on the state of the current running process. The processor user, kernel, wait, and idle resource attributes provide the approximate percentage of time that a specific processor is currently spending in each state. Therefore, the sum of these attributes is 100 at any given observation.

There are two protection modes that processes run in, kernel (or system) level and user level. Processes running in kernel mode run with kernel privileges and have access to kernel data. These processes include kernel processes (kprocs), and services (such as system calls and device drivers).

Processes running in user mode are normal applications with user level privileges and run in their own unique process space. When a user level process invokes a kernel service, for example, by making a system call, a mode switch occurs that causes the process to run in kernel mode while the service is running.

When the current running process makes a request that cannot be immediately satisfied, such as an I/O operation, the process is put into wait state.

The following persistent resource attributes can be retrieved for instances of the IBM.Processor resource class.

Name Identifies the name of the processor as known by the kernel.

ResourceHandle
An internally assigned handle that uniquely identifies the processor.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Specifies the set of nodes upon which the operational interface of a resource is available.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

ProcessorType
Identifies the type of processor.

Monitoring Utilization of a Single Processor: The following attributes can be monitored:

OpState Monitors whether the current operational state of the processor is online or offline.

PctTimeIdle Represents the percentage of time the processor is in the idle state.

PctTimeKernel
Represents the percentage of time the processor is running in kernel mode.

PctTimeUser Represents the percentage of time the processor is running in user mode.

PctTimeWait Represents the percentage of time the processor is running in wait state.

Predefined Conditions for Monitoring a Processor: This resource class represents the characteristics of the processors within a host. There is one instance of this resource for each processor installed in a host regardless of whether it is active or not. The following table shows the predefined conditions and examples of expressions that are available for monitoring a processor:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Processor state	OpState !=1	An event is generated when the processor goes offline.	OpState == 1	The event is rearmed when the processor returns online.
Processor idle time	(PctTimeIdle >= 80) && (PctTimeIdle @P >= 80)	An event is generated each time the processor is idle at least 80% of the time for two consecutive observations.	(PctTimeIdle < 50) (PctTimeIdle @P < 50)	The event is rearmed when the idle time for the processor is below 50% for two consecutive observations.
Processor wait time	(PctTimeWait >= 50) && (PctTimeWait @P >= 50)	An event is generated when the average time the processor is in wait state is at least 50% for two consecutive observations.	(PctTimeWait < 30) && (PctTimeWait @P < 30)	The event is rearmed when the processor is in wait state at most 30% of the time for two consecutive observations.

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Processor kernel time	(PctTimeKernel >= 70) && (PctTimeKernel @P >= 70)	An event is generated when the average time the processor is in kernel mode for two consecutive observations is 80%.	(PctTimeKernel < 20) && (PctTimeKernel @P < 20)	The event is rearmed when the kernel mode time for the processor is below 20% for two consecutive observations.
Processor user time	(PctTimeUser>=80) && (PctTimeUser@P > 80)	An event is generated when the average time the processor is in user mode for two consecutive observations is 80%.	(PctTimeUser < 50) && (PctTimeUser @P < 50)	The event is rearmed when the user mode time for the processor is below 50% for two consecutive observations.

Physical Volume Resource Class

The programmatic name of this resource class is IBM.Physical Volume. After a disk is added to the system, it must first be designated as a physical volume before it can be added to a volume group and used to contain a file system or paging space. A physical volume has certain configuration and identification information written on it. When a disk becomes a physical volume, it is divided into 512–byte physical blocks. Physical volumes have a unique name (typically **hdisk*x*** where ***x*** is a unique number on the system), which is permanently associated with the disk until it is undefined.

The following persistent resource attributes can be retrieved for instances of the IBM.PhysicalVolume resource class.

Name Identifies the name of the disk.

ResourceHandle
An internally assigned handle that uniquely identifies the physical volume.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Specifies the set of nodes upon which the operational interface of a resource is available.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

PVId Provides the unique identifier that is written on the disk.

Monitoring Physical Disks: These attributes, which reflect the basic performance of a physical disk, can be monitored:

PctBusy Average percentage of time the disk is busy from one observation of the value to the next.

RdBlkRate Average rate at which blocks are read from disk. The rate is calculated as the difference in total blocks read from the disk between two observations, divided by the time between observations.

WrBlkRate Average rate at which blocks are written to disk. The rate is calculated as the difference in total blocks written to the disk between two consecutive observations, divided by the time between observations.

XferRate Average rate of transfers per second that were issued to the physical disk. A transfer is an I/O request to the physical disk.

Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of indeterminate size. The rate is calculated as the difference in total transfers between two consecutive observations, divided by the time between observations.

Predefined Conditions for Monitoring Physical Disks: Each instance of this resource class represents a physical volume that has been defined to the system. All resources are monitored. The following table shows the predefined condition and examples of expressions that are available for monitoring physical disks:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Disk percent busy	(PctBusy >= 90) && (PctBusy@P >=90)	An event is generated when the disk has been busy at least 90% of the time for two consecutive observations.	PctBusy <80	The event is rearmed when the value decreases below 80%.
Disk read rate	RdBlkRate < 50	An event is generated when the rate per second of 512–byte blocks read from the disk is less than 50.	RdBlkRate > 100	The event is rearmed when the rate exceeds 100.
Disk write rate	WrBlkRate < 50	An event is generated when the rate per second of 512–byte blocks written to disk is less than 50.	WrBlkRate > 100	The event is rearmed when the rate exceeds 100.
Disk transfer rate	(XferRate > XferRate@P) && ((XferRate - XferRate@P) > (XferRate@P * 0.5))	An event is generated each time the rate of transfer to disk has increased 50%.	None	None

Adapters

The following adapters are supported, each by its own resource class:

ATM Device (IBM.ATMDevice)

All ATM adapters installed in a node are externalized through this resource manager. See “ATM Device Resource Class” on page 122 for more details.

Ethernet Device (IBM.EthernetDevice)

All Ethernet adapters installed in a node are externalized through this resource manager. See “Ethernet Device Resource Class” on page 122 for more details.

FDDI Device (IBM.FDDIDevice)

All FDDI adapters installed in a node are externalized through this resource manager. See “FDDI Device Resource Class” on page 124 for more details.

Token-Ring Device (IBM.TokenRingDevice)

All Token-Ring adapters installed in a node are externalized through this resource manager. See “Token-Ring Device Resource Class” on page 124 for more details.

See “Ethernet Device Resource Class” on page 122 for details on what can be monitored for an adapter. The other adapters have the same types of attributes. Only the adapter name is different.

ATM Device Resource Class

The programmatic name of this resource class is IBM.ATMDevice. The details of this class are identical to those of the IBM.EthernetDevice class except that the display name of the resource class is “ATM Device.” See the description of “Ethernet Device Resource Class” for details that also apply to this device.

Ethernet Device Resource Class

The programmatic name of this resource class is IBM.EthernetDevice. This resource class allows attributes of all Ethernet adapters that are installed in a system to be monitored. The network interfaces that may be defined on the adapters are not represented.

A network adapter card is the hardware that is physically attached to the network cabling. It is responsible for receiving and transmitting data at the physical level. The network adapter card is controlled by the network adapter device driver. A machine must have one network adapter card (or connection) for each network (not network type) to which it connects. For instance, if a host attaches to two Token-Ring networks, it must have two network adapter cards. When a new network adapter is physically installed in the system, the operating system assigns it a logical name. Some examples are: tok0 for a Token-Ring adapter, ent0 for an Ethernet adapter, or atm0 for an ATM adapter. The trailing number assigned, creates a unique logical number. For example, a second Token-ring adapter would have the logical name, tok1. The **lsdev** command can be used to display information about network adapters.

Messages received by a LAN adapter, referred to as frames, are encapsulated within destination, header, and trailer information added by the various network protocol layers. A counter, maintained for each adapter, tracks the number of frame-receive errors at the adapter device level that caused unsuccessful reception due to hardware or network errors. This counter is the raw value for **RecErrorRate**.

When frames are received by an adapter, they are transferred from the adapter into a device-managed receive queue. The number of packets accepted but dropped by the device driver level for any reason (for example, queue buffer shortage) is tracked by a counter, which provides the raw value of the **RecDropRate** attribute.

Messages and data sent by an application to a LAN adapter for transmission are broken up into packets and appended with address, header, and trailer information by the various network protocol layers. At the adapter device driver level, packets are placed in buffers on a transmit queue. The packets are appended with a network interface header, then transmitted as frames by the adapter device.

Counters are maintained for each adapter to track the number of transmission errors at the device level (due to hardware or network errors), number of transmission queue overflows at the device driver level (due to buffer shortage), and the number of packets dropped (packets not passed to the device by the driver for any reason). These counters provide the raw values for **XmitErrorRate**, **XmitOverflowRate**, and **XmitDropRate**, respectively.

The following persistent resource attributes can be retrieved for instances of the IBM.EthernetDevice resource class.

Name Identifies the name of the device.

ResourceHandle
An internally assigned handle that uniquely identifies the device.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Specifies the set of nodes upon which the operational interface of a resource is available.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

Monitoring Device Performance: The following attributes can be monitored:

RecErrorRate Represents the number of receive errors per second that occurred at the adapter level.

RecDropRate Represents the number of receive packets per second that were dropped by the adapter device driver.

XmitDropRate
Represents the number of outbound packets per second that were dropped by the adapter device driver.

XmitErrorRate
Represents the number of transmit errors per second that were detected at the adapter level.

XmitOverflowRate
Represents the number of transmit queue overflows per second that were detected by the adapter.

RecByteRate Reflects the number of bytes received per second.

RecPacketRate
Reflects the number of packets received per second.

XmitByteRate Reflects the number of bytes transmitted per second.

XmitPacketRate
Reflects the number of packets transmitted per second.

RecErrors Reflects the number of receive errors that have occurred at the adapter level.

RecDrops Reflects the number of receive packets that were dropped by the adapter device driver.

XmitDrops Reflects the number of outbound packets that were dropped by the adapter device driver.

XmitErrors Reflects the number of transmit errors that have been detected at the adapter level.

XmitOverflows
Reflects the number of transmit queue overflows that were detected by the adapter.

RecBytes Reflects the number of bytes received.

RecPackets Reflects the number of packets received.

XmitBytes Reflects the number of bytes transmitted.

XmitPackets Reflects the number of packets transmitted.

Predefined Conditions for Monitoring Device Performance: This resource class externalizes the characteristics of all Ethernet adapters that are installed in a system. It is important to note that this class does not represent the network interfaces that may be defined on the adapters. This class represents the actual adapters (ent0, etc.).

The characteristics are limited to a small set in the first release that are compatible with what is available through Event Management's aixos resource monitor.

The following table shows the predefined conditions and examples of expressions that are available for monitoring device performance. All resources are monitored.

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Ethernet receive error rate	RecErrorRate > 1	An event is generated when the number of receive errors exceeds 1 per second.	(RecErrorRate == 0) && (RecErrorRate@P == 0)	The event is rearmed when the receive error rate is 0 for two consecutive observations.
Ethernet receive drop rate	RecDropRate > 10	An event is generated when the number of receive packets dropped exceeds 10 per second.	RecDropRate < 5	The event is rearmed when the number of dropped packets goes below 5 per second.
Ethernet transmit drop rate	XmitDropRate > 10	An event is generated when the number of outbound packets dropped exceeds 10 per second.	XmitDropRate< 5	The event is rearmed when the number of dropped packets goes below 5 per second.
Ethernet transmit error rate	XmitErrorRate > 1	An event is generated when the number of transmit errors exceeds 1 per second.	(XmitErrorRate == 0) && (XmitErrorRate@P == 0)	The event is rearmed when the transmit error rate is 0 for two consecutive observations.
Ethernet transmit overflow rate	XmitOverflowRate > 10	An event is generated when the number of transmit queue overflows exceeds 10 per second.	XmitOverflowRate < 2	The event is rearmed when the number of overflows goes below 2 per second.

FDDI Device Resource Class

The programmatic name of this resource class is IBM.FDDIDevice. The details of this class are identical to those of the IBM.EthernetDevice class except that the display name of the resource class is "FDDI Device." See the description of "Ethernet Device Resource Class" on page 122 for details that also apply to this device.

Token-Ring Device Resource Class

The programmatic name of this class is IBM.TokenRingDevice. The details of this class are identical to those of the IBM.EthernetDevice class except that the display name of the resource class is "Token-Ring Device." See the description of "Ethernet Device Resource Class" on page 122 for details that also apply to this device.

Host Public Resource Class

The programmatic name of this resource class is IBM.HostPublic. It gives information on the local host's identifier token taken from the key files. The following dynamic attributes can be monitored for instances of the IBM.HostPublic resource class.

ResourceDefined
An event is generated each time a new resource is created or discovered.

ResourceUndefined
An event is generated each time a resource is deleted.

ConfigChanged
An event is generated each time a persistent class attribute or class ACL changes.

The following persistent resource attributes can be retrieved for instances of the IBM.HostPublic resource class.

ResourceHandle
An internally assigned handle that uniquely identifies this host.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Specifies the set of nodes upon which the operational interface of a resource is available.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

PublicKey
Specifies the text form of the local host's identifier token taken from the key files.

PublicKeyBinary
Specifies the binary form of the local host's identifier token taken from the key files.

Program Resource Class

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

Program Definition: The following persistent resource attributes can be retrieved for instances of the IBM.Program resource class.

Name Identifies a user defined name for the program definition.

ResourceHandle
An internally assigned handle that uniquely identifies the program resource.

Variety
Identifies which of the defined resource attributes and actions apply to the resource.

NodeIDs
Specifies the set of nodes upon which the operational interface of a resource is available.

NodeNameList
Retrieves the same information as the **NodeIDs** attribute.

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

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

Origin Specifies how the program definition was created. This persistent attribute indicates whether the program resource instance was defined explicitly through the DefineResource operation or implicitly through the specification of a select string as specified below. (0=*Implicitly Defined* and 1=*Explicitly Defined*).

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

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

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

This attribute can be monitored: **Processes**

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

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

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

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

ChangeList Contains a list of IDs for the processes that were added to or removed from the **CurrentList** since the last state change. Whether the list represents additions or deletions can be determined by comparing **CurPidCount** and **PrevPidCount**. If **CurPidCount** is

greater, this list contains additions; otherwise, it contains deletions. Additions and deletions are not combined in the same state change.

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

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

root  7786 8040 biod

root  8040 5624 biod

root  8300 8040 biod

root  8558 8040 biod

root  8816 8040 biod

root  9074 8040 biod
```

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

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

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

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

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

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description	Monitored Resources	Notes
sendmail daemon state	Processes .CurPidCount <=0	An event is generated whenever the **sendmail** daemon is not running.	Processes .CurPidCount> 1	The event is rearmed when the **sendmail** daemon is running.	**sendmail**	n/a
inetd daemon state	Processes .CurPidCount <=0	An event is generated whenever the **inetd** daemon is not running.	Processes .CurPidCount> 1	The event is rearmed when the **inetd** daemon is running.	**inetd**	n/a

Sensor Resource Manager

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

logged on to the system. Then an ERRM Condition and Response can be defined to run an action when the number of users logged on exceeds a certain threshold.

Sensor Resource Class

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

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

Predefined Condition for Sensor Resource Class

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

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

Chapter 4. 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 2, “Creating and Administering an RSCT Peer Domain” on page 7. For information on creating a management domain, refer to *IBM Cluster Systems Management for AIX 5L: 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 40, 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 132.

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 133 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 40.

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 UNIX host based authentication is supported, but other security mechanisms may be supported in the future. In the case of UNIX 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).

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 UNIX host-based security 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 UNIX Host Based Authentication

UNIX 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 **ctcasd** daemon provides and authenticates UNIX identity-based credentials for cluster security services; it is started by the RMC Service whenever the RMC service starts. The first time the **ctcasd** daemon starts on a particular node, it will create the private key for the node, and from this private key will derive the public key for the node. The node will use its private key to encrypt data. The node's public key will be provided to other nodes and will be used by them to decipher the data. Similarly, the public key can be used by the other nodes to encrypt data that will be 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. It is 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.

The first time the **ctcasd** daemon is started on a node, it will, in addition to creating the node's private/public key pair, also create an initial *trusted host list* file for the node. A trusted host list file associates host names with public key values; any host listed within this file will be capable of authenticating to the local node. When creating the initial trusted host list file for a node, the **ctcasd** daemon will create an entry for the local host and the host names associated with all AF_INET configured adapters that the daemon can detect. 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 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 change node. Specifically, you will need to modify the trusted host list file to include the new host name, associating it with the existing public key.

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	*/var/ct/cfg/ct_has.gkf* This file is readable and accessible only to the root user.
public key	*/var/ct/cfg/ct_has.pkf* This file is readable to all users on the local system. Write permission is not granted to any system user.
trusted host list	*/var/ct/cfg/ct_has.thl* 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 134. 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 is 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 UNIX 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 136. 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 138 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 40, 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 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 130, 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 139.

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 UNIX Host Based Authentication Mechanism" on page 134, it describes tasks that are specific to UNIX host based authentication. Finally, in "Configuring the Global and Local Authorization Identity Mappings" on page 139, 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 (UNIX host based security) exists, you should not need to modify this file unless you have moved the UNIX host based security MPM file to a new location and need to update that location in the configuration file. If you wish to disable UNIX host based security (even though this effectively eliminates *any* security), contact the IBM Support Center. See Chapter 7, "How to contact the IBM Support Center" on page 295.

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, UNIX host-based security 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 UNIX Host Based Authentication Mechanism

This section describes the administrative tasks you may need or want to perform that are related to the UNIX host based authentication mechanism. The following table outlines the administrative tasks covered.

This task	Describes how to:	Perform this task if:
"Configuring the ctcasd Daemon on a Node"	Modify a configuration file read by the Cluster Security Services daemon (**ctcasd**) upon startup.	You want to modify the operational parameters of the **ctcasd** 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 Identify Spoofing When Transferring Public Keys" on page 136	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 138	Modify a node's private and public keys.	A node's private key needs to be modified.

Configuring the ctcasd Daemon on a Node

When using UNIX host-based authentication as a security method, cluster security services uses the **ctcasd** daemon to provide and authenticate UNIX identity based credentials.

The **ctcasd** daemon obtains its operational parameters from a configuration file (**ctcasd.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/ctcasd.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 **ctcasd** configuration on a node to, for example, improve the performance of the daemon by altering the thread limits or enabling execution tracing, you should:

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

```
cp /usr/sbin/rsct/cfg/ctcasd.cfg /var/ct/cfg/ctcasd.cfg
```

2. Using an ASCII text editor, modify the new **ctcasd.cfg** file in **/var/ct/cfg**. The contents of the file will look similar to the following:

```
TRACE=
TRACEFILE=
TRACELEVELS=
TRACESIZE=
RQUEUESIZE=
MAXTHREADS=
MINTHREADS=
HBA_USING_SSH_KEYS= false
HBA_PRVKEYFILE=
HBA_PUBKEYFILE=
HBA_THLFILE=
HBA_KEYGEN_METHOD= rsa1024
SERVICES=hba CAS
```

The keywords listed in this file will set the configurable parameters for the **ctcasd** daemon on this node. The following table describes the configurable parameters.

Keyword	Description
TRACE	Reserved for future use.
TRACEFILE	Reserved for future use.
TRACELEVELS	Reserved for future use.
TRACESIZE	Reserved for future use.
RQUEUESIZE	Indicates the maximum length permitted for the daemon's internal run queue. If this value is not set, a default value of 64 is used.
MAXTHREADS	The limit to the number of working threads that the daemon may create and use at any given time (the "high water mark"). If this value is not set, a default value of 10 is used.
MINTHREADS	The number of idle threads that the daemon will retain if the daemon is awaiting further work (the "low water mark"). If this value is not, set, a default value of 4 is used.
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 Shells are not supported in the initial release.
HBA_PRVKEYFILE	Provides the full path name of the file that contains the local node's private key. If this value is not set, the default location of **/var/ct/cfg/ct_has.qkf** is used.
HBA_PUBKEYFILE	Provides the full path name of the file that contains the local node's public key. If this value is not set, the default location of **/var/ct/cfg/ct_has.pkf** is used.
HBA_THLFILE	Provides the full path name of the file that contains the local node's trusted host list. If this value is not set, the default location of **/var/ct/cfg/ct_has.thl** is used.
HBA_KEYGEN_METHOD	Indicates the method to be used by **ctcasd** to generate the private and public keys of the local node if the files containing these keys do not exist. Acceptable values are those that can be provided as arguments to the **ctskeygen -m** command. If no value is provided for this attribute, the default value of **rsa1024** is used.

Keyword	Description
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 cluster security software service provider.

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
   ```

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 for AIX 5L: Administration Guide*.
- In an RSCT peer domain, the **preprpnode** 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 **preprpnode** command, refer to "Step 1: Prepare Initial Security Environment on Each Node That Will Participate in the Peer Domain" on page 9.

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 **preprpnode** command's automatic key transferal. Refer to "Manually Transferring Public Keys" on page 137 for more information.
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 138 for more information.

Manually Transferring Public Keys: In an RSCT peer domain, public keys are normally transferred automatically by the **preprpnode** command. To guard against address and identity spoofing, you can disable the automatic key transfer, and instead manually copy the public keys.

Note: In a management domain, there is currently no way to disable the automatic key transfer carried out by the **updatenode** and **installnode** commands. For this reason, manual transfer of public keys in a management domain is not useful. If you were to manually transfer the keys, they would simply be overwritten when you issue the **updatenode** and **installnode** commands. In a management domain, you should instead merely verify the accuracy of the keys automatically transferred by CSM. Refer to "Verifying the Accuracy of Keys That Have Been Automatically Transferred" on page 138 for more information.

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. 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
   2499eb5b641e52dbceda2dccb8e8163f08070b5e3fc7e355319a84407ccbfc98252072ee1c0
   381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3c34e23bb976eb55a386619b70c5
   dc9507796c9e2e8eb05cd33cebf7b2b27cf630103
   (generation method: rsa1024)
   ```
3. Edit the **/tmp/hostname_pk.sh** file, converting it to a shell script that issues the `ctsthl` command to insert this public key into a trusted host list file. Determine if the cluster has been set up to use fully qualified host names for UNIX Host Based Authentication, or whether short host names are being used; and use the appropriate host name format as an 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 hostname -p
   120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8e15a5dda5
   2499eb5b641e52dbceda2dccb8e8163f08070b5e3fc7e355319a84407ccbfc98252072ee1c0
   381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3c34e23bb976eb55a386619b70c5
   dc9507796c9e2e8eb05cd33cebf7b2b27cf630103
   ```
4. 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.
5. Log on to the remote node.
6. 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
   ```

7. 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:
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8e15a5dda5
2499eb5b641e52dbceda2dccb8e8163f08070b5e3fc7e355319a84407ccbfc98252072ee1c0
381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3c34e23bb976eb55a386619b70c5
dc9507796c9e2e8eb05cd33cebf7b2b27cf630103
-------------------
```

When you are setting up the RSCT peer domain and issuing the **preprpnode** command, be sure to use its **-k** option to disable automatic transfer of public keys. For more information on the **preprpnode** command, refer to Chapter 2, “Creating and Administering an RSCT Peer Domain” on page 7.

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
   2499eb5b641e52dbceda2dccb8e8163f08070b5e3fc7e355319a84407ccbfc98252072ee1c0
   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. 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:
   120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8e15a5dda5
   2499eb5b641e52dbceda2dccb8e8163f08070b5e3fc7e355319a84407ccbfc98252072ee1c0
   381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3c34e23bb976eb55a386619b70c5
   dc9507796c9e2e8eb05cd33cebf7b2b27cf630103
   -------------------
   ```

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 132, 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 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 133). Currently, only one security mechanism (UNIX host based security) 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 (UNIX host based security). For more information on the format of identity mapping entries for UNIX host based security, refer to "Configuring the UNIX Host Based Authorization Mechanism Mappings" on page 141.

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 is sometimes unclear 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 UNIX 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*

Configuring the UNIX Host Based Authorization Mechanism Mappings

To indicate that an entry in the **ctsec_map.global** or **ctsec_map.local** file refers to the UNIX host based security 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 UNIX host-based security network identifier **jbrady@epsilon3.ibm.com** to the local user identifier **jbrady**.

To create a negative mapping (a mapping that explicitly state that no association is allowed for a particular security network identifier), used the reserved character **!**. For example, the following entry denies any local user identity association for the UNIX host-based security network identifier **jbrady@epsilon3.ibm.com**.

```
unix:!jbrady@epsilon3.ibm.com
```

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 13. 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 UNIX host-based security network identifier from the host **epsilon3.ibm.com** with the local user identifier **jbrady**.

Table 13. Using the wildcard character to match multiple user names in the security network identifier (continued)

For example, this entry:	Does this:
`unix:!*@epsilon3.ibm.com`	Explicitly states that no association is allowed for any UNIX host-based security network identifier from the host **epsilon3.ibm.com**.
`unix:j*@epsilon3.ibm.com=jbrady`	Associates any UNIX host-based security network identifier starting with the letter ″j″ from the host **epsilon3.ibm.com** 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 names portion of the security network identifier

Table 14. 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 UNIX host-based security network identifier 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 UNIX host-based security network identifier that contains the user name **jbrady** (regardless of the host).
`unix:zathras@*.ibm.com=zathras`	Associates any UNIX host-based security 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**.

You can only use the * wildcard character once within the host name 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 15. 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 UNIX host-based security 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**.

Table 15. Using the wildcard character to associate each security identifier with the local user identifier that matches the user name (continued)

For example, this entry:	Does this:
`unix:*@*=*`	Associates any UNIX host-based security network identifier 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**.

In addition to the wildcard character, there are two MPM-defined reserved words you can use when configuring the UNIX host-based authorization 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**.

Diagnosing Cluster Security Services problems

Requisite function

This is a list of the software directly used by the cluster security services component of RSCT. Problems within the requisite software may manifest themselves as error symptoms in the cluster security services. If you perform all the diagnostic procedures and error responses listed in this chapter, and still have problems with the cluster security services component of RSCT, you should consider these components as possible sources of the error. They are ordered with the most likely candidate first, least likely candidate last.

- TCP/IP
- UDP/IP
- UNIX Domain Sockets
- **/var** file system space, specifically the **/var/ct/cfg** directory
- **/usr/sbin/rsct** directory availability
- First Failure Data Capture Library (libct_ffdc)

- Cluster Utilities Library (libct_ct)
- System Resource Controller (SRC)

Error Information

The UNIX Host Based Authentication service daemon **ctcasd** records failure information to the AIX Error Log. The First Failure Data Capture (FFDC) facility is utilized to make these recordings, and an unique FFDC Failure Identifier is associated with each record. For compatibility to other UNIX based operating systems, records of any **ctcasd** failures are also made to the System Log by the FFDC utilities, provided that the System Log is active.

The **ctcasd** daemon uses the resource value of *ctcasd* in all of its AIX Error Log entries. To quickly obtain a brief summary of all entries recorded by the **ctcasd** daemon on the local system, issue the following command:

```
errpt -Nctcasd
```

A detailed report of all entries can be obtained using the command:

```
errpt -a -Nctcasd
```

By default, the AIX Error Log is stored in the file **/var/adm/ras/errlog**. One entry is recorded to this log per instance of the condition. Conditions are logged to the AIX Error Log file on the node where the incident occurred.

The AIX Error Log file size is limited, and it operates as a circular file. When the log file reaches its maximum length, the oldest entries within the log are discarded in order to record newer entries. AIX installs a **cron** job that removes any hardware related failure records within the log file after 90 days, and any software related failure records or operator information records after 30 days. The error log file size can be viewed and modified through SMIT using the **smit error** command, or through the following commands:

/usr/lib/errdemon -l
: Displays the error log file size

/usr/lib/errdemon -s
: Sets the error log file size.

Both the **smit** and the **errdemon** commands require *root* user authority.

Consult the documentation for the **errupdate** command for an explanation of the types associated with the AIX Error Log templates and the general format of AIX Error Log entries.

When the Cluster Security Services are installed, a number of AIX Error Log templates are installed for use by the **ctcasd** daemon. Templates of type INFO are operator informational messages only, and do not necessarily indicate a failure condition. Templates of type PERM record failure incidents that require operator intervention to resolve; the failure condition will remain unless action is taken to rectify the condition. The templates used by the **ctcasd** daemon to report failures indicate the nature of the failure, the most likely causes of the failure, and any actions that the system administrator can take in response tot he failure to either correct it or to assist the administrator in resolving the failure through the IBM Customer Support Center.

The following AIX Error Log templates are registered upon installation:

Table 16. Error Log templates for cluster security services

Label	Type	Description
CASD_UP_IN	INFO	**Explanation:** The **ctcasd** daemon has been started on the node. Authentication is now possible, using the UNIX Host Based Authentication mechanism. This is a normal operational message. **Details:** The **ctcasd** daemon is started automatically by the RSCT RMC daemon.
CASD_DN_IN	INFO	**Explanation:** The **ctcasd** daemon has been shut down on the node. Authentication attempts using the UNIX Host Based Authentication mechanism will no longer be successful until the daemon is restarted. This is a normal operational message. **Details:** The **ctcasd** daemon is shut down automatically by the RSCT RMC daemon when the RMC daemon is shut down. If this record is not accompanied by a record for the RMC daemon shut down soon after, the **ctcasd** daemon may have been forcibly shut down.
ARG_INT_ER	PERM	**Explanation:** The **ctcasd** daemon received invalid arguments or startup options. The daemon on this node has shut itself down. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** The error log entry will indicate the invalid option or argument that as detected by the daemon. Because the daemon is started in a consistent manner for all nodes, other nodes within the cluster may also experience this failure. Examine the error logs on the other cluster nodes to see if any other cluster nodes exhibit the same failure condition. This error log entry will display the argument or startup option that caused the failure. Make note of this information and contact the Cluster Security software service provider.
CASD_INT_ER	PERM	**Explanation:** An unexpected internal failure condition was detected by the **ctcasd** daemon. The daemon has shut itself down. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** Note the information recorded in this entry and contact the Cluster Security software service provider.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
KEYF_CFG_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to locate the local node's public or private key file. The daemon has shut itself down. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). Upon startup, the daemon was unable to locate one of these files. Concluding that this is a configuration failure, the daemon shut itself down. The identity of the missing file is recorded in the Detail Data section of this error log entry.
KEYF_QCREA_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to create a private key for the local node, or was unable to store the private key to a file. The daemon has shut itself down. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to create or store the private key for this host in the intended file. The intended file is named in the Detail Data section of this error log record. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
KEYF_PCREA_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to create a public key for the local node, or was unable to store the public key to a file. The daemon has shut itself down. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to create or store the public key for this host in the intended file. The intended file is named in the Detail Data section of this error log record. The daemon has shut itself down.
KEYF_QLCK_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to lock the private key file on the local node for exclusive use. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to obtain exclusive use of the private key file. The file is named in the Detail Data section of this error log record. Another process making use of this file may be hung, or may not have released its exclusive use lock on this file. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
KEYF_PLCK_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to lock the public key file on the local node for exclusive use. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to obtain exclusive use of the public key file. The file is named in the Detail Data section of this error log record. Another process making use of this file may be hung, or may not have released its exclusive use lock on this file. The daemon has shut itself down.
KEYF_ACC_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to access the files containing either the local system's public or private key. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to access at least one of these files. The files may not exist, or may have permissions set that do not permit processes running with *root* authority to access them. The name of the specific file causing the failure is named in the Detail Data section of this record. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
KEYF_STAT_ER	PERM	**Explanation:** The **ctcasd** daemon failed while issuing the C library stat() call on either the local system's public or private key files. The presence of these files cannot be confirmed by the daemon. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to determine if at least one of these files are missing form the local system. The file causing this failure is named in the Detail Data section of this record, along with the **errno** value set by the C library stat() routine. Examining the documentation for the stat() routine and determining what could cause the generation of the specific **errno** value may assist in determining the root cause of the failure. The daemon has shut itself down.
KEYF_QSPC_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to create a file to store the local node's private key because sufficient file system space was not available. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon detected that neither the public nor the private key file existed on this system. Assuming this to be the initial execution of the daemon, **ctcasd** attempted to create these files. The private key could not be stored because there is not sufficient space in the file system where the public key file — either **/var/ct/cfg/ct_has.qkf** or whatever override value was used in the **ctcasd.cfg** file — was to be stored. The name of the intended target file is provided in the Detail Data section of this record. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
KEYF_PSPC_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to create a file to store the local node's public key because sufficient file system space was not available. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon detected that neither the public nor the private key file existed on this system. Assuming this to be the initial execution of the daemon, **ctcasd** attempted to create these files. The public key could not be stored because there is not sufficient space in the file system where the public key file — either **/var/ct/cfg/ct_has.pkf** or whatever override value was used in the **ctcasd.cfg** file — was to be stored. The name of the intended target file is provided in the Detail Data section of this record. The daemon has shut itself down.
KEYF_QDIR_ER	PERM	**Explanation:** The **ctcasd** daemon could not access the directory where the private key file for the local system is stored. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to access the directory where the private key file is supposed to reside on the local system. The directory may be missing, or permissions may have been altered on one or more elements of the directory path to prevent access from root authority processes. The Detail Data section of this record contains the path name of the directory used by the daemon when the failure was detected. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
KEYF_PDIR_ER	PERM	**Explanation:** The **ctcasd** daemon could not access the directory where the public key file for the local system is stored. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** For UNIX Host Based Authentication to succeed, each host must possess both a private key, and a public key derived from that private key. When the **ctcasd** daemon executed for the first time after installation, these keys are created and stored by default in the following locations: • **/var/ct/cfg/ct_has.qkf** (private key) • **/var/ct/cfg/ct_has.pkf** (public key) These defaults can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to access the directory where the public key file is supposed to reside on the local system. The directory may be missing, or permissions may have been altered on one or more elements of the directory path to prevent access from root authority processes. The Detail Data section of this record contains the path name of the directory used by the daemon when the failure was detected. The daemon has shut itself down.
THL_CREAT_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to create the initial UNIX Host Based Authentication Trusted Host List for the local system. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** To authenticate remote clients using UNIX Host Based Authentication, the local host must posses a Trusted Host List file, which associates known trusted host names to the node's associated public key value. When the **ctcasd** daemon is executed for the first time after installation, an initial Trusted Host List file is created and initially populated with the local node's name and public key. This file is stored by default in **/var/ct/cfg/ct_has.thl**. The default path name can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to create the initial Trusted Host List file. The intended name of the Trusted Host List file is provided in the Detail Data section of this record. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
THL_ACC_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to access the Authentication Trusted Host List for the local system. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** To authenticate remote clients using UNIX Host Based Authentication, the local host must posses a Trusted Host List file, which associates known trusted host names to the node's associated public key value. When the **ctcasd** daemon is executed for the first time after installation, an initial Trusted Host List file is created and initially populated with the local node's name and public key. This file is stored by default in **/var/ct/cfg/ct_has.thl**. The default path name can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to access the initial Trusted Host List file. The file may not exist, or may have permissions altered to prevent access to the file. The intended name of the Trusted Host List file is provided in the Detail Data section of this record. The daemon has shut itself down.
THL_SPC_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to create a file to store the local node's Trusted Host List because sufficient file system space was not available. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** To authenticate remote clients using UNIX Host Based Authentication, the local host must posses a Trusted Host List file, which associates known trusted host names to the node's associated public key value. When the **ctcasd** daemon is executed for the first time after installation, an initial Trusted Host List file is created and initially populated with the local node's name and public key. This file is stored by default in **/var/ct/cfg/ct_has.thl**. The default path name can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon detected that the Trusted Host List file did not exist on this system. Assuming this to be the initial execution of the daemon, **ctcasd** attempted to create this files. The file data could not be stored because there is not sufficient space in the file system where the Trusted Host List file was to be stored. The name of the intended file is provided in the Detail Data section of this record. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
THL_DIR_ER	PERM	**Explanation:** The **ctcasd** daemon could not access the directory where the UNIX Host Based Authentication Trusted Host List file for the local system is stored. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** To authenticate remote clients using UNIX Host Based Authentication, the local host must posses a Trusted Host List file, which associates known trusted host names to the node's associated public key value. When the **ctcasd** daemon is executed for the first time after installation, an initial Trusted Host List file is created and initially populated with the local node's name and public key. This file is stored by default in **/var/ct/cfg/ct_has.thl**. The default path name can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon was unable to access the directory where the Trusted Host List file is supposed to reside on the local system. The directory may be missing, or permissions may have been altered on one or more elements of the directory path to prevent access from *root* authority processes. The Detail Data section of this record contains the path name of the directory used by the daemon when the failure was detected. The daemon has shut itself down.
HID_MEM_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to allocate dynamic memory while creating the UNIX Host Based Authentication host identifier token for the local system. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** To authenticate remote clients using UNIX Host Based Authentication, the local host must posses a Trusted Host List file, which associates known trusted host names to the node's associated public key value. When the **ctcasd** daemon is executed for the first time after installation, an initial Trusted Host List file is created and initially populated with the local node's name and public key. This file is stored by default in **/var/ct/cfg/ct_has.thl**. The default path name can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon detected that the Trusted Host List file did not exist on this system. Assuming this to be the initial execution of the daemon, **ctcasd** attempted to create this files. While creating the host identifier token to be stored in this file for the local system, **ctcasd** was not able to allocate dynamic memory to store the token. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
I18N_MEM_ERR	PERM	**Explanation:** The **ctcasd** daemon was unable to convert UNIX Host Based Authentication host identifier token data either to or from a locale independent format. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** To authenticate remote clients using UNIX Host Based Authentication, the local host must posses a Trusted Host List file, which associates known trusted host names to the node's associated public key value. When the **ctcasd** daemon is executed for the first time after installation, an initial Trusted Host List file is created and initially populated with the local node's name and public key. This file is stored by default in **/var/ct/cfg/ct_has.thl**. The default path name can be overridden by the files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override). The daemon detected that the Trusted Host List file did not exist on this system. Assuming this to be the initial execution of the daemon, **ctcasd** attempted to create this files. While creating the host identifier token to be stored in this file for the local system, **ctcasd** was not able to convert this information either to or from a locale independent format. The daemon has shut itself down.
CTS_MEM_ERR	PERM	**Explanation:** The **ctcasd** daemon was unable to dynamically allocate memory. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** The daemon dynamically allocates memory to construct UNIX Host Based Authentication credentials and to authenticate these credentials. During one of these attempts, the daemon was unable to obtain dynamic memory. The internal routine that attempted to allocate this memory, and the amount of memory requested, are listed in the Detail Data section of this record. The daemon has shut itself down.
CTS_ENV_ERR	PERM	**Explanation:** The **ctcasd** daemon detected that it was being invoked in an incorrect environment or configuration. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** The **ctcasd** daemon attempts to change to a specific working directory, submit itself to System Resource Controller (SRC) control, and create a UNIX Domain Socket to interface with the cluster security services library. During the startup of the daemon, one of these efforts failed. The Detail Data section will list the intended working directory for the process and the socket file name that the daemon was to create. The daemon has shut itself down.
CTS_DCFG_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to read its configuration information from a file. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** The files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override) provide configuration information to the **ctcasd** daemon. Upon startup, the daemon reads this information to determine its operational parameters. When the daemon started, the file could not be read. The name of the configuration file is listed in the Detail Data section of this record. The daemon has shut itself down.

Table 16. Error Log templates for cluster security services (continued)

Label	Type	Description
CTS_THRDI_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to create and initialize process threads. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** The files **/usr/sbin/rsct/cfg/ctcasd.cfg** (default) or **/var/ct/cfg/ctcasd.cfg** (override) provide configuration information to the **ctcasd** daemon, including instructions as to how many threads to create. The daemon encountered a failure while creating and initializing at least one thread. The number of available threads on the system may need to be increased, or the number of active processes and threads on the system may need to be decreased. Consult the error log entry for specific responses to take.
CTS_QUE_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to create an internal process thread queue for organizing and dispatching working threads. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** This error log entry will provide internal diagnostic information on the cause of the failure. Make note of this information and contact the Cluster Security software service provider.
CTS_THRD_ER	PERM	**Explanation:** The **ctcasd** daemon detected an unexpected failure in the execution of one of its process threads. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** This error log entry will provide internal diagnostic information on the cause of the failure. Make note of this information and contact the Cluster Security software service provider.
CTS_USVR_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to set up the service to handle requests via its UNIX Domain Socket. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** The **ctcasd** daemon interfaces with the cluster security services library through a UNIX Domain Socket. This socket may have been removed, or permissions on the file or directory may have been altered. The name of the socket file is provided in the Detail Data section of this record. The daemon is unable to set up a service thread for this socket as a result. The daemon has shut itself down.
CTS_ISVR_ER	PERM	**Explanation:** The **ctcasd** daemon was unable to set up the service to handle requests via an Internet Domain Socket. Authentication attempts using the UNIX Host Based Authentication mechanism will not be successful on this node. **Details:** The **ctcasd** daemon interfaces with certain cluster security services library requests through an Internet Domain Socket. The daemon was unable to set up a service thread to handle these requests because of a failure condition detected with the Internet Domain Socket. The daemon is unable to set up a service thread for this socket as a result. The daemon has shut itself down

Trace information

ATTENTION - READ THIS FIRST

Do *not* activate this trace facility until you have read this section completely, and understand this material. If you are not certain how to properly use this facility, or if you are not under the guidance of IBM Service, do *not* activate this facility.

Activating this facility may result in degraded performance of your system. Activating this facility may also result in longer response times, higher processor loads, and the consumption of system disk resources. Activating this facility may also obscure or modify the symptoms of timing-related problems.

The cluster security services libraries exploit the Cluster Trace facility. By default, these libraries do not generate trace information. Trace information can be obtained by activating one or more of the available Cluster Trace tracing levels and specifying a trace output file. Any trace output generated is specific to events and processing that occurs on the local system; security events on remote nodes within the cluster are not reflected within this trace output. To trace authentication and authorization related processing within the cluster, it may be necessary to activate tracing on multiple nodes within the cluster, and for IBM Customer Support to consolidate these traces and detect patterns within the trace files.

Tracing must not be activated within the cluster security services libraries without instruction or guidance from the IBM Customer Support Center.

Trace is activated by setting two environment variables for a process using the cluster security services libraries:

CT_TR_TRACE_LEVELS

This environment variable is used to control what tracing points and levels of detail are activated. The format of this environment variable is *component:category=level*.

For example, to activate the trace points within the cluster security services library **libct_sec** to trace the entry and exit of routines:

```
export CT_TR_TRACE_LEVELS="_SEA:API=1"
```

To enable multiple tracing levels, separate the trace level specifications by a comma:

```
export CT_TR_TRACE_LEVELS="_SEA:API=1,_SEU:API=1"
```

CT_TR_FILENAME

This environment variable names the output file where trace information is to be stored. To avoid confusion, specify a fully qualified path name for this variable.

Trace output files are recorded in binary format. The **rpptr** command reads trace output files and converts them to text readable forms.

The following trace categories and levels are supported:

Table 17. Trace categories supported by cluster security services

Library	Component	Category	Level	Description
libct_sec	_SEA	Errors	1	Records incidents of failure detected by the cluster security services **libct_sec** library.
libct_sec	_SEA	API	1	Records the entry and exit points of **libct_sec** library and subroutine calls. This level is used to trace which routines are invoked to handle an application request. No data is displayed.
libct_sec	_SEA	API	8	Records the entry and exit points of internal cluster security services library and subroutine calls. Entry points display the parameter values provided by the calling routine. Exit points display the return code value being passed to the caller.
libct_sec	_SEA	BUF	1	Records when internal buffers of a certain type are created.
libct_sec	_SEA	Buffer	8	Records when internal buffers of certain types are created. Lists the buffer address and length.
libct_sec	_SEA	SVCTKN	4	Traces status changes in a cluster security services security services token — required by any exploiter of the cluster security services library — through the **libct_sec** library.
libct_sec	_SEA	CTXTKN	4	Traces status changes in a cluster security services security context token — which defined a secured context between a service requestor and a service provider — through the **libct_sec** library.
libct_sec	_SEU	Errors	1	Records incidents of failure detected by the UNIX Host Based Authentication Mechanism Pluggable Module.
libct_sec	_SEU	API	1	Records entry and exit points within the UNIX Host Based Mechanism Pluggable Module that were invoked in response to an application request. No data is displayed.
libct_sec	_SEU	API	8	Records entry and exit points within the UNIX Host Based Mechanism Pluggable Module that were invoked in response to an application request. Entry points display the parameter values provided by the calling routine. Exit points display the return code value being passed to the caller.
libct_sec	_SEU	BUF	1	Records when internal buffers of a certain type are created and modified by the UNIX Host Based Authentication mechanism Pluggable Module.
libct_sec	_SEU	Buffer	8	Records when internal buffers of certain types are created and modified by the UNIX Host Based Authentication mechanism Pluggable Module. Lists the buffer address and length.

Table 17. Trace categories supported by cluster security services (continued)

Library	Component	Category	Level	Description
libct_sec	_SEU	SVCTKN	4	Traces status changes in a cluster security services security services token — required by any exploiter of the cluster security services library — by the UNIX Host Based Authentication Mechanism Pluggable Module.
libct_sec	_SEU	CTXTKN	4	Traces status changes in a cluster security services security context token — which defined a secured context between a service requestor and a service provider — by the UNIX Host Based Authentication Mechanism Pluggable Module.
libct_mss	_SEM	Errors	1	Records incidents of failure detected by the cluster security services **libct_mss** library.
libct_mss	_SEM	API	1	Records the entry and exit points of **libct_mss** library and subroutine calls. This level is used to trace which routines are invoked to handle an application request. No data is displayed.
libct_mss	_SEM	API	8	Records the entry and exit points of **libct_mss** library and subroutine calls. Entry points display the parameter values provided by the calling routine. Exit points display the return code value being passed to the caller.
libct_mss	_SEM	Perf	1	Records data used to monitor the overall performance of the **libct_mss** functions. Performance assessments should only be made by IBM Customer Support Center personnel.
libct_idm	_SEI	Error	1	Records incidents of failure detected by the cluster security services **libct_idm** library.
libct_idm	_SEI	API	1	Records the entry and exit points of **libct_idm** library and subroutine calls. This level is used to trace which routines are invoked to handle an application request. No data is displayed.
libct_idm	_SEI	API	8	Records the entry and exit points of **libct_idm** library and subroutine calls. Entry points display the parameter values provided by the calling routine. Exit points display the return code value being passed to the caller.
libct_idm	_SEI	Mapping	1	Records the identity mapping rule utilized by cluster security services to map a network security identity to a local user identity.
libct_idm	_SEI	Mapping	2	Records the local identity that was mapped to a security network identity by the **libct_idm** library.
libct_idm	_SEI	Mapping	8	Records both the identity mapping rule utilized by cluster security services to map a network security identity to a local user identity, and the local identity obtained from applying this rule.

Table 17. Trace categories supported by cluster security services (continued)

Library	Component	Category	Level	Description
libct_idm	_SEI	Milestone	1	Generates a record to indicate that a specific internal checkpoint has been reached. This record contains only the name of the checkpoint.
libct_idm	_SEI	Milestone	8	Generates a record to indicate that a specific internal checkpoint has been reached. This record contains the name of the checkpoint and some diagnostic data that IBM Customer Support may need in tracing internal failures.
libct_idm	_SEI	Diag	1	Records diagnostic information about the identity mapping definition file input and output processing. This information is meaningful only to IBM Customer Support.

Information To Collect Prior To Contacting IBM Service

Collect any AIX Error Log entries from nodes experiencing security related failures.

Authentiation Issues

Determine which security mechanisms are involved in the authentication failure. For the UNIX Host Based Authentication mechanism, obtain any AIX Error Log entries from the failing systems for the **ctcasd** daemon.

These error log entries can be obtained issuing the following command:

```
errpt -Nctcasd -a > ctcasderr.out
```

If any trace output has been generated for these systems, locate the trace output files and have them ready for IBM Service. Normal execution of the security software does not result in the generation of trace output; this output tis obtained by activating the tracing controls within the security library. The preceding section on Trace Information must be consulted prior to attempting to generate any trace output from the security library. Convert any trace output to text format by issuing the **rpptr** command on the system where the trace output file resides:

```
/usr/sbin/rsct/bin/rpptr tracefile > ctsectrace.out
```

Diagnostic Procedures

Diagnostic procedures are divided into those oriented towards the two primary security functions: authentication and authorization.

Authentication Troubleshooting Procedures

Mechanism Independent Authentication Troubleshooting Procedures: When troubleshooting the RSCT Security subsystem, these procedures can be used regardless of the specific security mechanisms employed throughout the cluster. These diagnostic procedures should be performed first, before attempting to troubleshoot specific security mechanisms.

These diagnostic procedures should be performed by the **root** user.

Procedure 1: Verifying the Location of the Cluster Security Services Configuration File:

Purpose:

To ensure that the cluster security services libraries can locate configuration information for the node.

Instructions:

The cluster security services library employs a configuration file that informs the library which security mechanisms are currently available on the local system. By default, this information resides in the file **/usr/sbin/rsct/cfg/ctsec.cfg**. Should a system administrator care to modify or extend this configuration information, the file must be copied to the override location of **/var/ct/cfg/ctsec.cfg** before any modifications are made. If a configuration file exists as **/var/ct/cfg/ctsec.cfg** on the local node, the cluster security services library will ignore the default configuration file and use this one. Under normal circumstances, when all nodes within the cluster employ the same software levels of RSCT, all nodes should use either the default or the override file; there should not be a set of nodes using the default configuration while others use an override. Verify that at least one of these files is present on the local system, and that any such files are not zero-length files:

```
ls -l /usr/sbin/rsct/cfg/ctsec.cfg /var/ct/cfg/ctsec.cfg
```

Verifying The Diagnostic:

Normal configurations will yield a result similar to:

```
ls: 0653-341 The file /var/ct/cfg/ctsec.cfg does not exist
-r--r--r--   1 bin   bin  630  Apr 09 14:29
                     /usr/sbin/rsct/cfg/ctsec.cfg
```

At least one of the files should be detected, and any detected file should show read-only permissions and a size greater than zero bytes.

Failure Actions:

Restore the default cluster security services configuration file **/usr/sbin/rsct/cfg/ctsec.cfg** from either a system backup or from the RSCT installation media. Monitor the system to ensure that the file is not removed by another user or process.

Next Diagnostic Procedure:

Proceed to Procedure 2

Procedure 2: Verifying the Contents of the Cluster Security Services Configuration File:

Purpose:

To ensure that the configuration information for the node is valid.

Instructions:

Examine the configuration file that will be used by cluster security services. If an override file is in place (as described in Procedure 1), examine that file with a text editor; otherwise, examine the default file with a text editor. The format of the cluster security services configuration file is:

```
#Prior  Mnemonic   Code        Path                Flags
#-------------------------------------------------------
 1      unix       0x00001     /usr/lib/unix.mpm    i
```

Each line within the file constitutes and entry for a security mechanism. Any blank lines or lines beginning with a # character are ignored. Each entry not commented should posses an unique mnemonic for the security mechanism, code for the mechanism, and priority.

Verifying the Diagnostic:

Examine the contents of the file to ensure that none share a priority value, a mnemonic name, or a code number. For any entries that are not commented, verify that a binary file exists on the system in the location specified in the `Path` column.

Failure Actions:

If the file being examined is the override configuration file, consider moving it so that the default cluster security services configuration file will be used until problems with this file are corrected.

If any priority or code numbers are shared, modify the file to makes these values unique for each entry. It is best to examine other **ctsec.cfg** files elsewhere within the cluster and to choose values for the priority and code that agree with those used by the other cluster members. Do **not** alter the value for the mechanism mnemonic unless instructed to do so by the IBM Customer Support Center.

Next Diagnostic Procedure:

Proceed to Procedure 3.

Procedure 3: Verifying that Mechanism Pluggable Modules are Installed:

Purpose:

To ensure that the cluster security services library **libct_sec** can locate the mechanism pluggable modules (MPMs) required to use the security mechanisms configured in the **ctsec.cfg** file.

Instructions:

The **ctsec.cfg** configuration file provides the location of the MPM that is loaded by the cluster security services library to interface with that security mechanism. This location is specified in the Path column of each entry:

```
#Prior   Mnemonic   Code      Path                Flags
#------------------------------------------------------------
 1        unix       0x00001   /usr/lib/unix.mpm    i
```

MPMs shipped by RSCT reside in the **/usr/sbin/rsct/lib** directory and have an extension of ***.mpm**. RSCT places symbolic links to these modules in the **/usr/lib** directory so that the cluster security services library can find them as part of the default library path search. Verify that any MPM files listed in the configuration exist and are binary files. For example:

```
file /usr/lib/unix.mpm
```

If the file proves to be a symbolic link, check the type of file referenced by that link. For example:

```
file /usr/sbin/rsct/lib/unix.mpm
```

Verifying the Diagnostic:

For PowerPC based platforms, the loadable module should appear as:

```
 /usr/sbin/rsct/bin/unix.mpm: executable (RISC System 6000) or object
module
```

For Intel based platforms, the mechanism loadable module should appear as:

```
/usr/sbin/rsct/bin/unix.mpm: ELF 32-bit LSB shared object, Intel
80386, version 1
```

Failure Actions:

If the file being examined is the override configuration file, consider moving

it so that the default cluster security services configuration file will be used until problems with this file are corrected.

If mechanism pluggable modules exist in the **/usr/sbin/rsct/lib** directory but not the **/usr/lib** directory, make symbolic links to these files in the **/usr/lib** directory, or alter the default library search path setting (LIBPATH on AIX systems, LD_LIBRARY_PATH on Linux systems) to include the **/usr/sbin/rsct/lib** directory.

If MPMs are not found in either location, restore them from a system backup or from the RSCT installation media.

Next Diagnostic Procedure:

Proceed to Procedure 4.

Procedure 4: Verifying Consistent Cluster Security Services Configuration Throughout the Cluster:

Purpose:

To ensure that all cluster security services libraries within the cluster are using consistent configurations.

Instructions:

Unless the cluster consists of nodes at differing RSCT software levels, all nodes within the cluster should employ either the default cluster security services library configuration file, or they should use the override location for this file. Nodes would only use a mix of these files when the cluster contains back-level RSCT nodes that have been modified to operate within a cluster containing more recent RSCT nodes.

The exact content of this file will depend on the RSCT Cluster setup.

- In a management domain, each node must share at least one security mechanism in common with the Management Server. Verify this by examining the active cluster security services configuration files on the Management Server and any nodes that the Management Server controls.
- In an RSCT peer domain, each node must share all security mechanisms, since each node can be considered a fail-over replacement for each other node within the peer domain. Verify this by examining the active cluster security services configuration files on each node within the peer domain.

Verifying the Diagnostic:

Examine the cluster security services configuration files on all nodes within the cluster using a text editor. Verify that these files are consistent, using the criteria stated in the preceding “Instructions” subsection.

Failure Actions:

If modifications must be made to the configurations on specific nodes to make them consistent with the configurations on the remaining cluster nodes, **make modifications to the override configuration file instead of the default configuration file**. Edit the configuration files to be consistent. However, do **not** add entries to these files **unless** the system contains the mechanism pluggable module for any security mechanism that is to be added **and** that node is configured to make use of that security mechanism.

Next Diagnostic Procedure:

Determine which security mechanism would be used by an application, and proceed to the diagnostic procedures specific to that security mechanism.

UNIX Host Based Authentication Troubleshooting Procedures: UNIX Host Based Authentication relies upon the ability to resolve the IP address of a host to a host name. The local system's UNIX Host Based Authentication Trusted Host List is then searched to find an entry matching this host name and the public key associated with this host. Authentication failures can result if the UNIX Host Based Authentication Mechanism Pluggable Module and **ctcasd** daemon are unable to resolve IP addresses, or if the addresses are resolved in inconsistent ways.

Procedure 1: Verifying the location of the ctcasd daemon configuration file:

Purpose:

To ensure that the **ctcasd** daemon can locate its configuration information. The default location of this file is **/usr/sbin/rsct/cfg/ctcasd.cfg**, but this file can be overridden by a file in the location **/var/ct/cfg/ctcasd.cfg**.

Instructions:

Verify that at least one of these files is present on the local system, and that any such files are not zero-length files:

```
ls -l /usr/sbin/rsct/cfg/ctcasd.cfg /var/ct/cfg/ctcasd.cfg
```

Verifying the Diagnostic:

Normal configurations will yield a result similar to:

```
 ls: 0653-341 The file /var/ct/cfg/ctcasd.cfg does not exist
 -r--r--r--   1 bin   bin  824  Apr 09 14:29  /usr/sbin/rsct/cfg/ctcasd.cfg
```

Failure Actions:

Restore the default **ctcasd** configuration file **/usr/sbin/rsct/cfg/ctcasd.cfg** from either a system backup or from the RSCT installation media. Monitor the system to ensure that the file is not removed by another user or process.

Next Diagnostic Procedure:

Proceed to Procedure 2

Procedure 2: Verifying the contents of the ctcasd daemon configuration file:

Purpose:

To determine if the **ctcasd** daemon can read and use its configuration information. This configuration information exists in a text file. The default location of this file is **/usr/sbin/rsct/cfg/ctcasd.cfg**, but this file can be overridden by a file in the location **/var/ct/cfg/ctcasd.cfg**. The format of this file is:

```
TRACE= false
TRACEFILE=
TRACELEVELS=
TRACESIZE=
RQUEUESIZE=
MAXTHREADS=
MINTHREADS=
HBA_USING_SSH_KEYS= false
HBA_PRVKEYFILE=
HBA_PUBKEYFILE=
HBA_THLFILE=
HBA_KEYGEN_METHOD= rsa1024
SERVICES=hba CAS
```

Details

For more details on the **ctcasd.cfg** file, refer to "Configuring the ctcasd Daemon on a Node" on page 134.

Instructions:

Using a text editor, examine the file.

Verifying the Diagnostic:

Ensure that any settings are those intended. If the default locations for the private and public keyfiles are to be used, the HBA_PRVKEYFILE and HBAPPUBKEYFILE fields should not be set. If the default Trusted Host List File is to be used, the HBA_THLFILE field should not be set. **Make note of these settings, because later diagnostics will need this information**.

Failure Actions:

If the file being examined is the override configuration file, consider moving it so that the default **ctcasd** configuration file will be used until problems with this file are corrected.

Repair the settings within the configuration file to match those intended for the system.

Next Diagnostic Test:

Proceed to Procedure 3.

Procedure 3: Verifying Trusted Host List Availability:

Purpose:

To ensure that the local system is capable of authenticating UNIX Host Based credentials provided by remote client systems. UNIX Host Based Authentication uses private keys to generate credentials and the corresponding public key to authenticate these credentials. To authenticate credentials from remote client systems, the remote system's public key must be recorded in the local system's Trusted Host List file. The location of this file is provided in the **ctcasd** configuration file, which was verified in Procedure 2. The default location of this file is **/var/ct/cfg/ct_has.thl**.

Instructions:

Verify that the Trusted Host List file is present on the local system, and that the file is not zero-length. If the default Trusted Host List file location is used:

```
ls -l /var/ct/cfg/ct_has.thl
```

Verifying the Diagnostic:

Output should be similar to the following:

```
-r--r--r--  1 root  system   1256 Apr 23 16:47 /var/ct/cfg/ct_has.thl
```

Failure Actions:

The **ctcasd** daemon will attempt to create an initial trusted host list file when it starts. Proceed to Procedure 4 to continue validating the **ctcasd** configuration.

Next Diagnostic Test:

Proceed to Procedure 4.

Procedure 4: Verifying Private Key File Availability:

Purpose:

To ensure that the local system is capable of generating UNIX Host Based credentials. UNIX Host Based Authentication uses private keys to generate credentials and the corresponding public key to authenticate these credentials. The **ctcasd** configuration information — verified in Procedure 2 — dictates which files are used to store this information.

Instructions:

Verify that the private key file can be accessed and is a non-zero length file. If the default private key file is intended to be used:

```
ls -l /var/ct/cfg/ct_has.qkf
```

Verify the Diagnostic:

Output should be similar to the following:

```
-r--------  1 root  system    271 Apr 23 16:47 /var/ct/cfg/ct_has.qkf
```

The file must be owned by the **root** user and should have read-only permission for the **root** user only. The file size should not be incredibly large. Anything over 271 bytes should be considered suspicious.

Failure Actions:

If the file is missing, the **ctcasd** daemon will attempt to create a private key file when it starts. Proceed to Procedure 5 to continue validating the **ctcasd** configuration.

If the file exists but the ownership and permissions are not correct, consider the private key to be compromised. Proceed to "Error Symptoms, Responses, and Recoveries" on page 175 and perform the action associated with a compromised private key.

Next Diagnostic Test:

Proceed to Procedure 5.

Procedure 5: Verifying Public Key File Availability:

Purpose:

To ensure that the local system is capable of generating UNIX Host Based credentials. UNIX Host Based Authentication uses private keys to generate credentials and the corresponding public key to authenticate these credentials. The **ctcasd** configuration information — verified in Procedure 2 — dictates which files are used to store this information.

Instructions:

Issue the **ctskeygen -d** command to determine if the public key file exists, and to display the value of this public key if it does exist. If an alternate public key file location was specified in the **ctcasd** configuration file, use this file name as the argument to the **-f** option. For example:

```
ctskeygen -d -f /var/ct/cfg/hostpub.pkf
```

If a public key file exists, the public key should be displayed. If not, an error message will be displayed.

Verifying the Diagnostic:

Correct results depend on the results from Procedure 4.

- If no private key file exists on the node, a public key file should not exist, and the above command should generate an error. If a public key file does exist, then a configuration error exists.
- If a private key file exists, a public key file should also exist, and the above command should display a key value. **Make note of the key value or store it to a file**, since it will be needed in later diagnostic procedures. The output will be similar to:

  ```
  [c174n07][/]> ctskeygen -d -f /var/ct/cfg/ct_has.pkf
  120400b76496afa9653add28204392ada847be57d5928ee5acc666db9e885e4
  a91be65866f187f85cd906d80ae7348bb40428244740be206ad1afe0348e3f7
  c113aadc61f998fa3a25e09cd38fcbcf8e8ce5b32678af853e6c48a9702eed8
  fb05045fe3d011bdb4663e912c8d756ca39894097826d7800643dfbf3af7ba1
  bdc41b39790103
  (generation method: rsa1024)
  ```

Failure Actions:

Either both the private or public key files should exist, or neither should exist. If one file exists, the configuration error needs to be cleared and new

keys generated for the node. For more information, refer to “Error Symptoms, Responses, and Recoveries” on page 175. After new keys have been generated for the node, consult the cluster security services administration documentation to complete the population of the local node’s Trusted Host List.

Next Diagnostic Procedure:
Proceed to Procedure 6.

Procedure 6: Verify that the local host’s public key is recorded in the local node’s Trusted Host List file.:

Purpose:
Verify that the local node can validate UNIX Host Based credentials created by applications executing on this node. All nodes must record their own public key in their own Trusted Host List. The **ctcasd** configuration information — verified in Procedure 2 — dictates what file is used to store the Trusted Host List file.

Instructions:
Examine the **/etc/hosts** file on the local node to obtain the list of host names used for the local system. After obtaining this list, issue the **ctsthl -l** command to obtain the list of known public keys on this host. Save this information to a file to make the verification easier:

```
ctsthl -l > /tmp/thl.txt
```

Verifying the Diagnostic:
Ensure that each name that this host uses is listed in the Trusted Host List contents. If a name is missing, any application using that host name will be unable to authenticate to clients or services on this node.

Compare the key value listed for each host name. These keys should all be the same.

Compare each key value listed to the value obtained in Procedure 5. These values should be the same.

Failure Actions:
Add entries to the trusted host list for any local host names that may be missing, using the **ctsthl -a** command. For any entries that use incorrect key values, remove these entries with the **ctsthl -d** command and add back new entries using the correct public key value using the **ctsthl -a** command.

Next Diagnostic Procedure:
Proceed to Procedure 8

Procedure 8: Verify that the ctcasd daemon is operational:

Purpose:
To verify that the local system can create and validate UNIX Host Based credentials.

Instructions:
The **ctcasd** daemon is controlled by the System Resource Controller (SRC) and operates as a standalone daemon. Verify that the **ctcasd** daemon is active using the SRC query:

```
lssrc -s ctcas
```

Verifying the Diagnostic:
If the daemon is active, the command will respond:

```
Subsystem   Group     PID     Status
ctcas       rsct      89347   active
```

If the daemon is not active, the command will respond:

```
Subsystem   Group     PID     Status
ctcas       rsct              inoperative
```

If the daemon has not been properly installed, an error message will be displayed.

Failure Actions:

If **ctcasd** is not active, attempt to activate it using the SRC command:

```
startsrc -s ctcas
```

Wait about five seconds, and then reissue the query instruction listed in the "Instructions" subsection above. If the daemon is not reported as active, examine the error information logs on the system to determine a possible cause of failure. See the section **Error Information** earlier in this chapter for assistance in finding this information.

Next Diagnostic Test:

Proceed to Procedure 9.

Procedure 9: Determining Host Name Setup:

Purpose:

To determine if the UNIX Host Based Authentication mechanism is capable of producing credentials on this system. This mechanism requires that a host name be assigned to each node within the cluster, and that all nodes within the cluster use a consistent format for host names. Host name resolution must be consistent, or authentication attempts can fail.

Instructions:

Locate the host's address in the **/etc/hosts** file and issue the **host** command to determine the host name that would be returned for that address. For example, if the local host has an address of 9.119.10.30:

```
host 9.119.10.30
```

Verifying the Diagnostic:

If the node has a host name assigned and is configured to use a fully qualified host name, the command will yield results similar to the following:

```
epsilon3.ialliance.org is 9.119.10.30
```

If the node has a host name assigned and is configured to use a short host name, the command will yield results similar to the following:

```
epsilon3 is 9.119.10.30
```

Make a note of the name provided and whether the name is fully qualified. This will be needed for Procedure 13.

Failure Actions:

If the node does not have a host name assigned, assign the node a host name. It is recommended that the node be assigned a fully qualified host name, unless the other cluster nodes are configured to use short host names.

Next Diagnostic Test:

Proceed to Procedure 10.

Procedure 10: Verifying Domain Name Server Setup:

Purpose:

To ensure that the security library can resolve host IP addresses and names to the correct host name equivalent. The UNIX Host Based Authentication mechanism maps public keys to host names. Host name resolution must be consistent, or authentication attempts can fail.

Instructions:

Examine the **/etc/resolv.conf** file to determine if any name servers have been set up for the node. If a name server has been established, and entry with the label `nameserver` will appear at least once within this file.

Verifying the Diagnostic:

Using a text file viewer, examine the **/etc/resolv.conf** file and search for `nameserver` entries. It is not necessary for a node to have established a name server for host name resolution, but make note of any host names or addresses if a name server is specified.

Failure Actions:

It is not necessary for a node to have established a name server for host name resolution. However, it is likely that if any one host within a cluster configuration makes use of a domain name server, the rest of the nodes should also be making use of the domain name server. Record this fact for later use in Procedures 11 and 14.

Next Diagnostic Test:

Proceed to Procedure 11.

Procedure 11: Verifying the Host Name Resolution Order:

Purpose:

To ensure that the security library can resolve host IP addresses and names to the correct host name equivalent. The UNIX Host Based Authentication mechanism maps public keys to host names. Host name resolution must be consistent, or authentication attempts can fail.

Instructions:

Check if the local host specifies the name resolution order through the configuration files **/etc/irc.conf** or **/etc/netsvc.conf**. Neither of these files should exist if a name server entry was not found on the local host in Procedure 10. If neither or these files exist, the host is using the default name resolution order. Otherwise, note the order of name resolution as specified in these files. Make note of this information as it will be used in Procedure 14.

Verifying the Diagnostic:

If a name server entry was not found while performing Procedure 10, ensure that neither the **/etc/netsvc.conf** nor the **/etc/irc.conf** file exists on the node.

Failure Actions:

If a name server is not specified but either the **/etc/netsvc.conf** or the **/etc/irc.conf** files exist, the node may have an incorrect network configuration. Troubleshoot the node's network configuration to make sure it is correct.

Next Diagnostic Test:

Proceed to Procedure 12

Procedure 12: Verify the UNIX Host Based Authentication configuration on the remaining nodes:

Purpose:

To verify that services or clients on remote nodes can authenticate with applications running on the local node.

Instructions:

Repeat Procedures 1 through 9 on the remaining nodes within the cluster.

Next Diagnostic Test:

Proceed to Procedure 13.

Procedure 13: Verifying Consistent Host Naming:

Purpose:

To ensure that all nodes within the cluster are capable of authenticating requests from each other. UNIX Host Based Authentication makes use of host names during the authentication process. Host name resolution needs to yield a host name that is the same at that retrieved on the local host. If host naming formats are inconsistent, or name resolutions yield different names on different hosts, authentication attempts can fail.

Instructions:

Compare the results of Procedure 9 on all nodes within the cluster. All nodes should either make use of a fully qualified host name format or a short host name format.

Failure Actions:

Proceed to Procedure 14 to ensure that all nodes are consistent in domain name server usage. If some nodes are unable to reach the domain name server, then it is possible that host name resolution will differ between hosts.

If all hosts are not consistent in their use of domain name servers, reconfigure the cluster to make consistent use of name servers.

If domain name servers are not in use within the cluster, reconfigure the cluster nodes to use either fully qualified host names or short host names. A mixture of methods will result in authentication failures using the UNIX Host Based authentication mechanism. This reconfiguration may require modifications to the Trusted Host List files throughout the cluster, so that these lists associate the correct host name (either fully qualified or short) to the correct public key for that system.

Next Diagnostic Test:

Proceed to Procedure 14.

Procedure 14: Testing for Public Key Distribution:

Purpose:

To verify that the cluster nodes have distributed public keys. UNIX Host Based Authentication utilizes private and public key pairs. If the cluster nodes have not exchanged public keys between themselves and recorded these keys in their Trusted Host Lists, authentication will not succeed.

How widely the public keys need to be distributed depends upon the cluster configuration in use:

- In a management domain, the public key of the Management Server must be distributed to all nodes within the Management Domain, and the public key for each Managed Node must be distributed to the Management Server.
- In an RSCT peer domain, the public keys for all nodes within the Peer Domain must be distributed to all other nodes within the Peer Domain.

Instructions:

This is a time consuming procedure that involves all nodes within the cluster configuration. On each node, obtain the public key for the host using the **ctskeygen -d** command. Record this information someplace where it can be easily obtained when executing instructions on other nodes. Also obtain the list of host names used by each node within the cluster by examining the **/etc/hosts** file on each node as was done in Procedure 7.

Once these lists are obtained, determine which nodes require knowledge of the public key using the criteria specified in the "Purpose" section above. Prepare to run commands on that node, either by logging in to the remote system or by opening a remote shell connection to that host.

Issue the **ctsthl -l** command on the remote host and save the output to a file.

Verifying the Diagnostic:

Ensure that each name that this host uses is listed in the Trusted Host List contents. If a name is missing, any application using that host name will be unable to authenticate to clients or services on the remote node.

Compare the key value listed for each host name used by the local host. These keys should all be the same.

Compare each key value listed to the value obtained in Procedure 5. These values should be the same.

Failure Actions:

Add entries to the trusted host list for any local host names that may be missing, using the **ctsthl -a** command. For any entries that use incorrect key values, remove these entries with the **ctsthl -d** command and add back new entries using the correct public key value using the **ctsthl -a** command. Refer to "Manually Transferring Public Keys" on page 137 for information on distributing the local node's public key to remote node.

Next Diagnostic Test:

Proceed to Procedure 15.

Procedure 15: Verify Host Name Resolution on the Remaining Cluster Nodes:

Purpose:

To ensure that the security library can resolve host IP addresses and names to the correct host name equivalent. The UNIX Host Based Authentication mechanism maps public keys to host names. Host name resolution must be consistent, or authentication attempts can fail.

Instructions:

Repeat Procedures 10 and 11 for each node in the cluster. All nodes should utilize the same name resolution scheme: either resolving host names locally using the **/etc/hosts** file contents alone, or by using a domain name server.

Verifying the Diagnostic:

Ensure that all nodes are consistent in both using (or not using) a name

server, and that the same hosts are used by all nodes as a name server (if one is in use). Ensure that all nodes make use of the same order for name resolution.

Failure Actions:

Modify the configurations on the cluster nodes to use the same name resolution order and the same name servers. The operating system may need to be restarted on any nodes where such configuration changes are made in order for these changes to take effect.

Next Diagnostic Test:

Proceed to Procedure 16.

Procedure 16: Verify Access to Domain Name Servers:

Purpose:

To ensure that the security library can resolve host IP addresses and names to the correct host name equivalent through a name server. The inability to contact a domain name server can inject significant performance degradation to the UNIX Host Based authentication process, and can cause the UNIX Host Based authentication procedure to fail.

Instructions:

If the cluster nodes are not making use of name servers, skip this procedure. Verify that each node in the cluster can access the name servers discovered in Procedure 10 and configured in Procedure 15 by issuing a ping command from each node to the name servers. For example:

```
ping -c1 9.199.1.1
ping -c1 129.90.77.1
```

Verifying the Diagnostic:

If the name server can be reached, you will get results similar to the following:

```
PING 9.114.1.1: (9.199.1.1): 56 data bytes
64 bytes from 9.199.1.1: icmp_seq=0 ttl=253 time=1 ms

----9.199.1.1 PING Statistics----
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1/1/1 ms
```

If the name server cannot be reached, an error message will be displayed:

```
PING 9.114.1.1: (9.199.1.1): 56 data bytes

----9.199.1.1 PING Statistics----
1 packets transmitted, 0 packets received, 100% packet loss
```

Failure Actions:

Verify that the correct name or address is being used for the domain name server. Troubleshoot the network connectivity between any failing node and the name server. Consider changing to a backup or alternate name server.

Authorization Troubleshooting Procedures

Identity Mapping Troubleshooting Procedures: The cluster security services identity mapping facility permits administrators to associate an operating system user identity on the local system to a security network identity. Future versions of the cluster security services library will permit group based authorization making use of such mapped identities.

Procedure 1: Verifying Default Global Mapping File:

Purpose:

To verify that the cluster security services library can locate the correct identity mapping definition files for the local system. Two input files are supported: a global mapping file intended to contain identity maps for network identities that are intended to be consistent throughout the cluster; and a local mapping file that defines identity maps intended to be used on the local node alone. The local definition file resides in the file **/var/ct/cfg/ctsec_map.local**. A default global definition file is shipped with RSCT in the file **/usr/sbin/rsct/cfg/ctsec_map.global**. If system administrators wish to extend the contents of this file, the file should be copied to its override position of **/var/ct/cfg/ctsec_map.global** and modifications made to that version of the file.

Instructions:

Test for the presence of the default global identity map file:

```
file /usr/sbin/rsct/cfg/ctsec_map.global
```

Verifying the Diagnostic:

Output will be similar to:

```
/usr/sbin/rsct/cfg/ctsec_map.global: commands text
```

Failure Actions:

Restore the default global map definition file from either a system backup or from the RSCT installation media.

Next Diagnostic Test:

Proceed to Procedure 2.

Procedure 2: Verifying Override Global Mapping File:

Purpose:

To verify that the cluster security services library can locate the correct identity mapping definition files for the local system. Two input files are supported: a global mapping file intended to contain identity maps for network identities that are intended to be consistent throughout the cluster; and a local mapping file that defines identity maps intended to be used on the local node alone. The local definition file resides in the file **/var/ct/cfg/ctsec_map.local**. A default global definition file is shipped with RSCT in the file **/usr/sbin/rsct/cfg/ctsec_map.global**. If system administrators wish to extend the contents of this file, the file should be copied to its override position of **/var/ct/cfg/ctsec_map.global** and modifications made to that version of the file.

Instructions:

Test for the presence of the override global identity map file:

```
file /var/ct/cfg/ctsec_map.global
```

Verifying the Diagnostic:

The absence of an override global identity map file does not necessarily constitute a failure condition. If the file is present, output will be similar to:

```
/var/ct/cfg/ctsec_map.global: commands text
```

Next Diagnostic Test:

Proceed to Procedure 3.

Procedure 3: Verifying Local Mapping File:

Purpose:

To verify that the cluster security services library can locate the correct identity mapping definition files for the local system. Two input files are

supported: a global mapping file intended to contain identity maps for network identities that are intended to be consistent throughout the cluster; and a local mapping file that defines identity maps intended to be used on the local node alone. The local definition file resides in the file **/var/ct/cfg/ctsec_map.local**. A default global definition file is shipped with RSCT in the file **/usr/sbin/rsct/cfg/ctsec_map.global**. If system administrators wish to extend the contents of this file, the file should be copied to its override position of **/var/ct/cfg/ctsec_map.global** and modifications made to that version of the file.

Instructions:

Test for the presence of the local identity map file:

```
file /var/ct/cfg/ctsec_map.local
```

Verifying the Diagnostic:

The absence of an override global identity map file does not necessarily constitute a failure condition. If the file is present, output will be similar to:

```
/var/ct/cfg/ctsec_map.global: commands text
```

Next Diagnostic Test:

Proceed to Procedure 4.

Procedure 4: Checking the Mapping for a Network Identity on a Node:

Purpose:

To verify that the cluster security services library will find the correct local user map for a network identity.

Instructions:

Select a network identity from a specific security mechanism supported by cluster security services. Examine the cluster security services configuration file — **/usr/sbin/rsct/cfg/ctsec.cfg** or **/var/ct/cfg/ctsec.cfg** — to determine the correct mnemonic to be used for that security mechanism. Provide both the network identity and the security mnemonic as arguments to the **ctsidmck** command. For example, to test the mapping for the the UNIX Host Based network identity *zathras@epsilon3.org*:

```
ctsidmck -dm -munix zathras@epsilon3.org
```

This command will display any map that was obtained, as well as display the mapping file entry that resulted in the map.

Verifying the Diagnostic:

Verify that the resulting map — if any — was the intended mapping for the network identifier.

Failure Actions:

If a mapping was intended and not found, extend the identity mapping definition files to include a mapping entry to form this mapping. Add the definition either to the local definition file (if the map is intended for this node only) or the override version of the global mapping file (if the map is intended to eventually be used on all nodes within the cluster). Do **not** make modifications to the default global identity mapping definition file **/usr/sbin/rsct/cfg/ctsec_map.global**. After making the necessary modifications, reissue Procedure 4 to ensure that the correct modifications were made.

If a mapping was intended and an incorrect mapping was displayed, proceed to Procedure 6.

If a mapping was not intended an a map was found, proceed top Procedure 5.

Next Diagnostic Test:
None.

Procedure 5: Modifying Incorrect Mapping Definitions:

Purpose:
To ensure that a local operating system user identity map is not granted to a network identity that should not receive such a map.

Instructions:
Find the mapping definition file that specifies the rule in error that was displayed in Procedure 4. For example, if Procedure 4 indicated that the rule ″*@epsilon3.org=draal″ mapped ″zathras@epsilon3.org″ to ″draal″, issue the following command to locate the file that specifies this rule:

```
grep -l "*@epsilon3.org=draal" \
/usr/sbin/rsct/cfg/ctsec_map.global \
/var/ct/cfg/ctsec_map.global \
/var/ct/cfg/ctsec_map.local
```

This command will display the name of the file that contains the rule. Modify this file using a text editor to correct the mapping rule to yield the correct result.

Verifying the Diagnostic:
Return to Procedure 4 and reissue the test.

Next Diagnostic Test:
None.

Procedure 6: Adding Mapping Definitions:

Purpose:
To ensure that a local operating system user identity map is granted to a network identity that should receive it.

Instructions:
Determine whether the identity mapping is unique to the local node, or will apply to all nodes within the cluster configuration.

- If the mapping is intended to be used only on this node, ensure that the local mapping definition file **/var/ct/cfg/ctsec_map.local** exists. If not, issue the following commands to bring it into being:

  ```
  touch /var/ct/cfg/ctsec_map.local
  chmod 644 /var/ct/cfg/ctsec_map.local
  ```

- If the mapping is intended to be used on all nodes within the cluster configuration, ensure that the override global mapping file **/var/ct/cfg/ctsec_map.global** exists. If not, issue the following command to bring it into being:

  ```
  cp /usr/sbin/rsct/cfg/ctsec_map.global \
      /var/ct/cfg/ctsec_map/global
  ```

  Using a text editor, modify the correct file to include a mapping rule to yield the desired map. Remember, order is important within these files. New additions should be added at the **end** of the file to allow current mappings to continue to work as before.

Verifying the Diagnostic:
Return to Procedure 4 and reissue the test.

Next Diagnostic Test:

None.

Error Symptoms, Responses, and Recoveries

Error Condition:	Action:
Private or public key file missing on a node	Action 1
Private and public key mismatch on a node	Action 1
ctcasd daemon abnormally terminates	Action 2
Cannot add entries to Trusted Host List File	Action 3
Trusted Host List File size too large	Action 3
Authentication Failures	Action 4 and Action 5
Host Name Resolution and Short Host Name Support	Action 5
Private key becomes compromised	Action 6

Action 1

Description:

Used to correct UNIX Host Based Authentication mechanism configuration errors where one of the necessary key files is missing, or to recover from a mismatch between the node's private and public keys. New private and public keys are generated for this node in this step.

Repair Action:

Follow these steps:

1. Log onto the local system as **root**.
2. Shut down all trusted services on the local node.
3. On each node within the cluster configuration (including the local node), remove the public key for this node from the Trusted Host List files on these nodes using the **ctsthl -d** command. Be sure to remove all entries for every name that can be used by this node.
4. Remove the trusted host.
5. On the local node, determine the parameters for private and public keys on the node. Examine the UNIX Host Based Authentication configuration file — **/var/ct/cfg/ctcasd.cfg** or **/usr/sbin/rsct/cfg/ctcasd.cfg** — and find the values for the following entries:

   ```
   HBA_PRVKEYFILE
   HBA_PUBKEYFILE
   HBA_KEYGEN_METHOD
   ```

   If no explicit values are provided for these entries, the defaults used by the **ctcasd** daemon are:

   ```
   HBA_PRVKEYFILE=/var/ct/cfg/ct_has.qkf
   HBA_PUBKEYFILE=/var/ct/cfg/ct_has.pkf
   HBA_KEYGEN_METHOD=rsa512
   ```
6. Issue the **ctskeygen -n -d** command to create new private and public keys for the local node and store them in the appropriate files. The command will display the new public key value to standard output, so redirect standard output to a file. The new key value will be needed in later steps. If the default **ctcasd** settings are used by the configuration file, issue the command:

```
ctskeygen -n -mrsa512 -p/var/ct/cfg/ct_has.pkf \
   -q/var/ct/cfg/ct_has.qkf -l > /tmp/pubk.out
```

7. Issue the **ctsthl -a** command to add the node's new public key to the local node's Trusted Host List. Use the public key value as recorded to the temporary file created in Step 6. Reissue the command for each host name used by the local host until an entry exists for all the names used by this host.
8. Manually distribute the new public key to the cluster nodes. For information on how to do this, refer to "Manually Transferring Public Keys" on page 137.
9. Restart the trusted services on the local node.
10. Remove the temporary file created in Step 6.
11. Log off from the node.

Repair Test:

Perform the troubleshooting procedures for the UNIX Host Based Authentication mechanism listed earlier in this section to validate the repair.

Recovery Actions:

Read this paragraph in its entirety. A recovery action exists that can help avoid triggering failures related to private and public key mismatches. This recovery action will **disable** the UNIX Host Based Authentication mechanism on the local node. Applications on the local node will not be able to authenticate with other applications using the UNIX Host Based Authentication mechanism. If no other mechanism is available, then all applications on the local node will be unauthenticated if this recovery action is taken. Do not use this recovery action if this solution is not acceptable.

1. Log on to the node as **root**.
2. Shut down all trusted services on the node.
3. If an override for the cluster security services configuration file does not exist in the file **/var/ct/cfg/ctsec.cfg**, create this file using the following command:

   ```
   cp /usr/sbin/rsct/cfg/ctsec.cfg /var/ct/cfg/ctsec.cfg
   ```

4. Using a text editor, insert a comment character # at the start of the entry for the UNIX Host Based Authentication mechanism:

   ```
   #Prior Mnemonic Code    Path               Flags
   #-------------------------------------------------------
   # 1    unix     0x00001 /usr/lib/unix.mpm  i
   ```

5. Restart the trusted services on this node
6. Log off the node.

Recovery Removal:

To remove the above recovery action:

1. Log on to the node as root.
2. Shut down all trusted services on the node.
3. Using a text editor, edit the override cluster security services configuration file **/var/ct/cfg/ctsec.cfg**. Delete the comment character # from the start of the entry for the UNIX Host Based Authentication mechanism:

   ```
   #Prior Mnemonic Code    Path               Flags
   #-------------------------------------------------------
   1     unix     0x00001 /usr/lib/unix.mpm  i
   ```

4. Compare the override configuration file to the default configuration file using the **diff** command:

```
diff /var/ct/cfg/ctsec.cfg /usr/sbin/rsct/cfg/ctsec.cfg
```

5. If the files are not different, remove the override file **/var/ct/cfg/ctsec.cfg** from this system; it is no longer required.
6. Restart the trusted services on this node.
7. Log off the node.

Action 2

Description:

Used to identify, rectify, or report failures in the **ctcasd** daemon.

Repair Actions:

Examine the AIX Error Log for any entries made by the **ctcasd** daemon. Consult the earlier section on Error Information for assistance in locating these entries. Perform any recommended actions indicated in the AIX Error Log entry for the failure condition.

Repair Test:

Restart the **ctcasd** daemon. If the daemon will not restart or stay operational, examine the AIX Error Log for any new failure records recorded by the daemon. Contact the IBM Customer Support Center for assistance if the problem cannot be rectified on site.

Recovery Actions:

Read this paragraph in its entirety. A recovery action exists that can help avoid triggering failures related to private and public key mismatches. This recovery action will **disable** the UNIX Host Based Authentication mechanism on the local node. Applications on the local node will not be able to authenticate with other applications using the UNIX Host Based Authentication mechanism. If no other mechanism is available, then all applications on the local node will be *unauthenticated* if this recovery action is taken. Do not use this recovery action if this solution is not acceptable.

1. Log on to the node as **root**.
2. Shut down all trusted services on the node.
3. If an override for the cluster security services configuration file does not exist in the file **/var/ct/cfg/ctsec.cfg**, create this file using the following command:

   ```
   cp /usr/sbin/rsct/cfg/ctsec.cfg /var/ct/cfg/ctsec.cfg
   ```

4. Using a text editor, insert a comment character # at the start of the entry for the UNIX Host Based Authentication mechanism:

   ```
   #Prior Mnemonic Code    Path              Flags
   #-------------------------------------------------------
   # 1    unix     0x00001 /usr/lib/unix.mpm  i
   ```

5. Restart the trusted services on this node.
6. Log off the node.

Recovery Removal:

To remove the above recovery action:

1. Log on to the node as **root**.
2. Shut down all trusted services on the node.
3. Using a text editor, edit the override cluster security services configuration file **/var/ct/cfg/ctsec.cfg**. Delete the comment character # from the start of the entry for the UNIX Host Based Authentication mechanism:

```
#Prior Mnemonic Code    Path               Flags
#-------------------------------------------------------
 1     unix     0x00001 /usr/lib/unix.mpm  i
```

4. Compare the override configuration file to the default configuration file using the **diff** command:

   ```
   diff /var/ct/cfg/ctsec.cfg /usr/sbin/rsct/cfg/ctsec.cfg
   ```

   If the files are not different, remove the override file **/var/ct/cfg/ctsec.cfg** from this system; it is no longer required.
5. Restart the trusted services on this node.
6. Log off the node.

Action 3

Description:

Used to remove unused entries from the UNIX Host Based Authentication mechanism's Trusted Host List File.

Repair Actions:

Perform the following steps:

1. Select a time when system activity is low, and RMC clients will not be attempting to authenticate to the RMC subsystem.
2. Log onto the system as **root**.
3. Examine the UNIX Host Based Authentication mechanism configuration file — **/usr/sbin/rsct/cfg/ctcasd.cfg** or **/var/ct/cfg/ctcasd.cfg** — to determine what file is being used as the Trusted Host List file. This value is given in the following entry:

   ```
   HBA_PRVKEYFILE
   ```

   If no value is given for this entry, the default file location of **/var/ct/cfg/ct_has.htl** is in used.
4. Copy the trusted host list file to a backup. For example:

   ```
   cp /var/ct/cfg/ct_has.thl /var/ct/cfg/ct_has.thl.orig
   ```
5. Display the current contents of the trusted host list file, redirecting the output to a file. This file will be used to verify the actions of a shell script used in the subsequent steps. For example:

   ```
   /usr/sbin/rsct/bin/ctsthl -l -f /var/ct/cfg/ct_has.thl >\
       /tmp/thlorig.out
   ```

   The contents of this file will be similar to the following example:

   ```
   --------------------
   Host name: avenger.pok.ibm.com
   Identifier Generation Method: rsa1024
   Identifier Value:
   120400a25e168a7eafcbe44fde48799cc3a88cc177019100
   09587ea7d9af5db90f29415db7892c7ec018640eaae9c6bd
   a64098efaf6d4680ea3bb83bac663cf340b5419623be80ce
   977e153576d9a707bcb8e8969ed338fd2c1df4855b233ee6
   533199d40a7267dcfb01e923c5693c4230a5f8c60c7b8e67
   9eb313d926beed115464cb0103
   --------------------
   Host name: ppsclnt16.pok.ibm.com
   Identifier Generation Method: rsa1024
   Identifier Value:
   120400a25e168a7eafcbe44fde48799cc3a88cc177019100
   09587ea7d9af5db90f29415db7892c7ec018640eaae9c6bd
   a64098efaf6d4680ea3bb83bac663cf340b5419623be80ce
   977e153576d9a707bcb8e8969ed338fd2c1df4855b233ee6
   ```

```
533199d40a7267dcfb01e923c5693c4230a5f8c60c7b8e67
9eb313d926beed115464cb0103
--------------------
Host name: sh2n04.pok.ibm.com
Identifier Generation Method: rsa1024
Identifier Value:
120400a25e168a7eafcbe44fde48799cc3a88cc177019100
09587ea7d9af5db90f29415db7892c7ec018640eaae9c6bd
a64098efaf6d4680ea3bb83bac663cf340b5419623be80ce
977e153576d9a707bcb8e8969ed338fd2c1df4855b233ee6
533199d40a7267dcfb01e923c5693c4230a5f8c60c7b8e67
9eb313d926beed115464cb0103
--------------------
```

6. Copy this file to a new file. This new file will be used as the shell script to clean up the trusted host list file. For example:

   ```
   cp /tmp/thlorig.out /tmp/cleanthl
   ```

7. Select a name for a new trusted host list file. This is going to be the “compressed” or “cleaned up” trusted host list file. It will not become the “active” trusted host list file for a few steps yet. To ensure that the later step is as seamless as possible, select a file within the same directory as the existing trusted host list file. Create the file and set the file permissions to 444, so that the remaining steps will work properly. For example:

   ```
   touch /var/ct/cfg/ct_has.thl.new
   chmod 444 /var/ct/cfg/ct_has.thl.new
   ```

8. Edit the file created in Step 6, converting it to a shell script. For each entry, create a new **ctsthl** command to add an entry to a brand new trusted host list file. Specify the new trusted host list file selected in Step 7 as the argument to the **-f** option. Use the “`Host Name:`” listed in each entry as the argument to the **-n** option, the “`Identifer Generation Method:`” listed as the argument to the **-m** option, and the string after the “`Identifier Value:`” as the argument to the **-p** option. Ensure that all new **ctsthl** commands are part of a single script command line. Continuing the example from Step 6, the new contents of the **/tmp/cleanthl** will create a new trusted host list file **/var/ct/cfg/ct_has.thl.new**; the new **/tmp/cleanthl** file contents would be:

   ```
       /usr/sbinrsct/bin/ctsthl -f/var/ct/cfg/ct_has.thl.new -a \
       -n avenger.pok.ibm.com \
       -m rsa1024 \
       -p \
   120400a25e168a7eafcbe44fde48799cc3a88cc177019100
   09587ea7d9af5db90f29415db7892c7ec018640eaae9c6bd
   a64098efaf6d4680ea3bb83bac663cf340b5419623be80ce
   977e153576d9a707bcb8e8969ed338fd2c1df4855b233ee6
   533199d40a7267dcfb01e923c5693c4230a5f8c60c7b8e67
   9eb313d926beed115464cb0103
        /usr/sbinrsct/bin/ctsthl -f/var/ct/cfg/ct_has.thl.new -a \
        -n ppsclnt16.pok.ibm.com \
        -m rsa1024 \
        -p \
   120400a25e168a7eafcbe44fde48799cc3a88cc177019100
   09587ea7d9af5db90f29415db7892c7ec018640eaae9c6bd
   a64098efaf6d4680ea3bb83bac663cf340b5419623be80ce
   977e153576d9a707bcb8e8969ed338fd2c1df4855b233ee6
   533199d40a7267dcfb01e923c5693c4230a5f8c60c7b8e67
   9eb313d926beed115464cb0103
        /usr/sbinrsct/bin/ctsthl -f/var/ct/cfg/ct_has.thl.new -a \
        -n sh2n04.pok.ibm.com \
        -m rsa1024 \
        -p \
   ```

```
120400a25e168a7eafcbe44fde48799cc3a88cc177019100
09587ea7d9af5db90f29415db7892c7ec018640eaae9c6bd
a64098efaf6d4680ea3bb83bac663cf340b5419623be80ce
977e153576d9a707bcb8e8969ed338fd2c1df4855b233ee6
533199d40a7267dcfb01e923c5693c4230a5f8c60c7b8e67
9eb313d926beed115464cb0103
```

9. Execute this shell script to create a new trusted host list file. Note that the new trusted host list file will not be used yet, since it is known by a new name. For example:

   ```
   sh /tmp/cleanthl
   ```

10. Verify that Step 9 executed correctly by listing the contents of the new trusted host list file, capturing the output in a file, and comparing those results to the original output captured in Step 5. For example:

    ```
    /usr/sbin/rsct/bin/ctsthl -l -f \
      /var/ct/cfg/ct_has.thl.new > /tmp/thlnew.out
    diff /tmp/thlnew.out /tmp/thlorig.out
    ```

    There should be no differences detected.

11. Overlay the new trusted host list file over the old. For example:

    ```
    mv /var/ct/cfg/ct_has.thl.new /var/ct/cfg/ct_has.thl
    ```

12. Clean up any temporary files that were made to accomplish this (in our example, the temporary files are `/tmp/thlnew.out`, `/tmp/thlorig.out`, and `/tmp/cleanthl`).
13. Log off the system and resume normal operations.

Repair Tests:

Repair is tested using Step 10 in the above sequence.

Recovery Actions:

Read this paragraph in its entirety. A recovery action exists that can help avoid triggering failures related to private and public key mismatches. This recovery action will **disable** the UNIX Host Based Authentication mechanism on the local node. Applications on the local node will not be able to authenticate with other applications using the UNIX Host Based Authentication mechanism. If no other mechanism is available, then all applications on the local node will be *unauthenticated* if this recovery action is taken. Do not use this recovery action if this solution is not acceptable.

1. Log on to the node as **root**.
2. Shut down all trusted services on the node.
3. If an override for the cluster security services configuration file does not exist in the file **/var/ct/cfg/ctsec.cfg**, create this file using the following command:

   ```
   cp /usr/sbin/rsct/cfg/ctsec.cfg /var/ct/cfg/ctsec.cfg
   ```

4. Using a text editor, insert a comment character # at the start of the entry for the UNIX Host Based Authentication mechanism:

   ```
   #Prior Mnemonic Code    Path               Flags
   #-------------------------------------------------------
   # 1    unix     0x00001 /usr/lib/unix.mpm  i
   ```

5. Restart the trusted services on this node.
6. Log off the node.

Recovery Removal:

To remove the above recovery action:

1. Log on to the node as **root**.
2. Shut down all trusted services on the node.

3. Using a text editor, edit the override cluster security services configuration file **/var/ct/cfg/ctsec.cfg**. Delete the comment character # from the start of the entry for the UNIX Host Based Authentication mechanism:

```
#Prior Mnemonic Code    Path               Flags
#-------------------------------------------------------
 1     unix     0x00001 /usr/lib/unix.mpm  i
```

4. Compare the override configuration file to the default configuration file using the **diff** command:

```
diff /var/ct/cfg/ctsec.cfg /usr/sbin/rsct/cfg/ctsec.cfg
```

   If the files are not different, remove the override file **/var/ct/cfg/ctsec.cfg** from this system; it is no longer required.

5. Restart the trusted services on this node.
6. Log off the node.

Action 4

Description:

Used to identify the cause of authentication related failures.

Repair Actions:

Authentication failures can be specific to the underlying security mechanism, or they can be the result of configuration problems with the cluster security services library. Perform the troubleshooting procedures outlined in “Authentication Troubleshooting Procedures” on page 159. Perform any recommended actions indicated by these procedures. If conditions persist, contact IBM Customer Support for additional assistance.

Action 5

Description:

Setting consistent host name resolution.

Repair Actions:

Before performing this action, understand the desired cluster configuration in regards to:

- Domain name servers - Does the cluster make use of domain name servers? If so, decide on the name resolution order between the domain name server and the local **/etc/hosts** file. The default setting can vary between AIX and Linux operating systems. It is recommended that the search order be explicitly stated in either the **/etc/netsvc.conf** or the **/etc/irc.conf** files. If the search order will use the **/etc/hosts** file before contacting the domain name server, then updates to the **/etc/hosts** file on each node will be required as follows:
  - Management Domains: The host name and address of the Management Server will need to be added to the **/etc/hosts** file for each node within the Management Domain. The name and address of each managed node will need to be added to the **/etc/hosts** file on the Management Server.
  - Peer Domains: The host names and addresses of each node within the cluster will need to be added to the **/etc/hosts** file on each node within the cluster.
- Host name format - Does the cluster span multiple domains? If so, fully qualified host names should be in use. If the cluster is contained within a

single domain, then short host names can be used, although it is recommended that fully qualified host names be used to support future growth.

Perform the following tasks on each node within the cluster:

1. Log on to the node as **root**.
2. Shut down all trusted services. It is preferred that the node be changed to single user mode.
3. If the cluster uses domain name servers, modify the **/etc/netsvc.conf** or the **/etc/irc.conf** files to specify the desired search order. Go to Step 7.
4. If a name server is in use and short host names only are to be used by the cluster nodes, edit the **/etc/hosts** file on this node to specify the address and short host name for this node. Also add any other nodes required for the type of cluster as indicated above, using the address and short host names for the required nodes. Go to Step 7.
5. If a name server is not in use and fully qualified host names only are to be used by the cluster nodes, edit the **/etc/hosts** file on this node to specify the address and fully qualified host name for this node. Also add any other nodes required for the type of cluster as indicated above, using the address and short host names for the required nodes. Go to Step 7.
6. If a name server is not in use and short host names only are to be used by the cluster nodes, edit the **/etc/hosts** file on this node to specify the address and fully qualified host name for this node. Also add any other nodes required for the type of cluster as indicated above, using the address and short host names for the required nodes. Go to Step 7.
7. If the system was changed to single user mode, change the system to multiuser mode.
8. Log off the node.

Repair Test:

Perform the diagnostic procedures in “UNIX Host Based Authentication Troubleshooting Procedures” on page 163.

Action 6:

Description:

Recovering from a security breach, when a node’s private key has become public knowledge or has otherwise been compromised.

Repair Actions:

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. When reforming the cluster, consider the following when configuring cluster security services in the new cluster:

1. Choose a new password for **root**. It is possible that the security breach may have started with the **root** password being compromised, because the private key file is only accessible to **root** users.
2. Consider using a stronger security protection within the private and public key. Use a more extensive key type such as **rsa1024** over smaller key types.

3. Ensure that only the **root** user is capable of accessing the private key file. No other system users should have any form of access to this file.
4. Ensure that the UNIX Host Based Authentication mechanism's configuration file **ctcasd.cfg** can only be modified by the **root** user.
5. Verify that the **ctcasd** binary file, located in **/usr/sbin/rsct/bin/ctcasd**, is the same as the binary file shipped in the RSCT installation media.
6. Monitor the private key file to ensure that the permissions on the file do not change.
7. Monitor the **ctcasd.cfg** configuration file to ensure that the permissions on the file do not change.
8. Monitor the **ctcasd** binary file for any changes in size or modification date.
9. Monitor the system more closely for security breaches.

Chapter 5. 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 12) or when you explicitly create a group using the **mkcomg** command (as described in “Creating a Communication Group” on page 24).

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.
- discusses diagnostic procedures and failure responses.

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 6, “The Group Services subsystem” on page 259. 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.

Crown Prince
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. This involves sending messages to other adapters in the network and monitoring the number of incoming packets. If no packets are received, the local 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 11).

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 **startrpdomain** or **mkcomg** commands. See “Step 3: Bring the Peer Domain Online” on page 12 and “Creating a Communication Group” on page 24 for more information on the **startrpdomain** 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 21. 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 pages or the *Reliable Scalable Cluster Technology for AIX 5L: 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 **cthatstune** 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)

In the **/var/ct/***cluster_name***/run/cthats** directory, a directory named **cthats**.*cluster_name* is created, where *cluster_name* is the RSCT peer domain name. This 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 syslog.

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 24 and “Modifying a Communication Group’s Characteristics” on page 21. You can also use the **cthatstune** command (as described in “The cthatstune tuning command” on page 189) 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.

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.

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**.

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 195 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.

The second step is to add the Topology Services daemon to the SRC using the **mkssys** command.

The 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, removes the entry from **/etc/inittab**, and removes the Topology Services daemon communications port number from **/etc/services**. It 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 16.

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, 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.

- **ipsrcroutesend** (default is 1)
- **ipsrcrouterecv** (default is 0)
- **ipsrcrouteforward** (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.

Tuning the Topology Services subsystem

The default settings for the frequency and sensitivity tunable attributes discussed in "Configuring Topology Services" on page 193 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 21). 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
```

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 **cthatsctrl** 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).
- Six 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.

Diagnosing Topology Services problems

This section discusses diagnostic procedures and failure responses for the Topology Services component of RSCT. The list of known error symptoms and the associated responses are in the section “Error symptoms, responses, and recoveries” on page 244. A list of the information to collect before contacting the IBM Support Center is in the section “Information to collect before contacting the IBM Support Center” on page 229.

Requisite function

This is a list of the software directly used by the Topology Services component of RSCT. Problems within the requisite software may manifest themselves as error symptoms in Topology Services. If you perform all the diagnostic routines and error responses listed in this chapter, and still have problems with the Topology Services component of RSCT, you should consider these components as possible sources of the error. They are listed with the most likely candidate first, least likely candidate last.

- UDP/IP communication
- Cluster adapter configuration
- Unix Domain sockets
- security libraries
- SRC
- First Failure Data Capture (FFDC) library
- **/var/ct/***cluster_name* directory

Error information

The error log file is stored in **/var/adm/ras/errlog** by default. One entry is logged for each occurrence of the condition. The condition is logged on every node where the event occurred.

The Error Log file may wrap, since the file has a limited size. Data is stored in a circular fashion. Also, the system is shipped with a crontab file to delete hardware errors more than 90 days old and software errors and operator messages more than 30 days old.

The command:

```
/usr/lib/errdemon -l
```

shows current settings for the error logging daemon.

The command:

```
/usr/lib/errdemon -s
```

is used to change the size of the error log file.

Both commands require **root** authority.

Unless otherwise noted, each entry refers to a particular instance of the Topology Services daemon on the local node. Unless otherwise noted, entries are created on each occurrence of the condition.

Unless otherwise noted, each entry refers to a particular instance of the Topology Services daemon on the local node. Unless otherwise noted, entries are created on each occurrence of the condition.

Error Logs and templates

Table 18 on page 202 lists the error log templates used by Topology Services, sorted by **Error Label**. An **Explanation** and **Details** are given for each error.

The Topology Services subsystem creates error log entries for the following conditions:

- TS_ASSERT_EM
- TS_CMDFLAG_ER
- TS_CPU_USE_ER
- TS_CTIPDUP_ER
- TS_CTLOCAL_ER
- TS_CTNODEDUP_ER
- TS_DEATH_TR
- TS_DUPNETNAME_ER
- TS_FD_INTFC_NAME_ST
- TS_FD_INVAL_ADDR_ST
- TS_HAIPDUP_ER
- TS_HALOCAL_ER
- TS_HANODEDUP_ER
- TS_IOCTL_ER
- TS_IPADDR_ER
- TS_LATEHB_PE
- TS_LIBERR_EM
- TS_LOC_DOWN_ST
- TS_LOGFILE_ER
- TS_LONGLINE_ER

- TS_LSOCK_ER
- TS_MACHLIST_ER
- TS_MISCFG_EM
- TS_NIM_DIED_ER
- TS_NIM_ERROR_STUCK_ER
- TS_NIM_ERROR_INTERNAL_ER
- TS_NIM_ERROR_RDWR_ER
- TS_NIM_ERROR_TRAF_ER
- TS_NIM_ERROR_MSG_ER
- TS_NIM_NETMON_ERROR_ER
- TS_NIM_OPEN_ERROR_ER
- TS_NODENUM_ER
- TS_NODEUP_ST
- TS_OFF_LIMIT_ER
- TS_REFRESH_ER
- TS_RSOCK_ER
- TS_SEMGET_ER
- TS_SERVICE_ER
- TS_SHMAT_ER
- TS_SHMEMKEY_ER
- TS_SHMGET_ER
- TS_SP_DIR_ER
- TS_SPIPDUP_ER
- TS_SPLOCAL_ER
- TS_SPNODEDUP_ER
- TS_START_ST
- TS_STOP_ST
- TS_THATTR_ER
- TS_THCREATE_ER
- TS_THREAD_STUCK_ER
- TS_UNS_SIN_TR

When you retrieve an error log entry, look for the Detail Data section near the bottom of the entry.

Table 18. Error Log templates for Topology Services

Label	Type	Description
TS_ASSERT_EM	PEND	**Explanation:** Topology Services daemon exited abnormally. **Details:** This entry indicates that the Topology Services daemon exited with an **assert** statement, resulting in a core dump being generated. Standard fields indicate that the Topology Services daemon exited abnormally. Detail Data fields contain the location of the **core** file. This is an internal error. Data needed for IBM Service to diagnose the problem is stored in the **core** file (whose location is given in the error log) and in the Topology Services daemon service log. See “Topology Services service log” on page 225. Since only six instances of the Topology Services daemon service log are kept, it should be copied to a safe place. Also, only three instances of the **core** file are kept. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.
TS_AUTHMETH_ER	PERM	**Explanation:** The Topology Services startup script cannot retrieve active authentication methods using command **/usr/sbin/rsct/bin/lsauthpts**. **Details:** This entry indicates that command **/usr/lpp/ssp/bin/lsauthpts**, run by the Topology Service startup script on the control workstation, was unable to retrieve the active authentication methods in a system partition. This error occurs when the startup script is running on the control workstation during initial startup or refresh. When this error occurs, all Topology Services daemons in the system partition will terminate their operations and exit. Diagnosing this problem requires collecting data only on the control workstation. Standard fields indicate that the startup script cannot retrieve active authentication methods in a system partition using command **lsauthpts**. The problem may be one of the following: • The system partition has an incorrect set of active partition methods. • The current system partition cannot be identified. Detail Data fields contain the return code of command **lsauthpts** and the location of the startup script log. The error message returned by command **lsauthpts** can be found in the startup script log.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_CMDFLAG_ER	PERM	**Explanation:** Topology Services cannot be started due to incorrect flags. **Details:** This entry indicates that the Topology Services daemon was unable to start because incorrect command line arguments were passed to it. This entry refers to a particular instance of Topology Services on the local node. Other nodes may have been affected by the same problem. Standard fields indicate that the daemon was unable to start because incorrect flags were passed to it. Detail Data fields show the path name to the daemon user log, which contains more detail about the problem. This problem may be one of the following: • Topology Services was started manually in an incorrect way. • Incompatible versions of the daemon and startup script are being used. • The SRC definition for the subsystem was manually set to an incorrect value. Information about the cause of the problem may not be available once the problem is cleared.
TS_CTIPDUP_ER	PERM	**Explanation:** See TS_HAIPDUP_ER.
TS_CTNODEDUP_ER	PERM	**Explanation:** See TS_HANODEDUP_ER.
TS_CTLOCAL_ER	PERM	**Explanation:** See TS_HALOCAL_ER.
TS_CPU_USE_ER	PERM	**Explanation:** The Topology Services daemon is using too much CPU. The daemon will exit. **Details:** This entry indicates that the Topology Services daemon will exit because it has been using almost 100% of the CPU. Since Topology Services runs in a real time fixed priority, exiting in this case is necessary. Otherwise, all other applications in the node will be prevented from running, Also, it is likely that the daemon is not working properly if it is using all the CPU. A **core** dump is created to allow debugging the cause of the problem. This entry refers to a particular instance of Topology Services running on a node. The standard fields indicate that the Topology Services daemon is exiting because it is using too much of the CPU, and explains some of the possible causes. The detailed fields show the amount of CPU used by the daemon (in milliseconds) and the interval (in milliseconds) where the CPU usage occurred. Collect the data described in “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center. In particular, the daemon log file and the most recent core files should be collected.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_DEATH_TR	UNKN	**Explanation:** Lost contact with a neighboring adapter. **Details:** This entry indicates that heartbeat messages are no longer being received from the neighboring adapter. This entry refers to a particular instance of the Topology Services daemon on the local node. The source of the problem could be either the local or remote node. Data from the remote node should also be obtained. Standard fields indicate that a local adapter is no longer receiving packets from the remote adapter. Detail Data fields contain the node number and IP address of the remote adapter. Data about the loss of connectivity may not be available after the problem is cleared. The local or remote adapter may have malfunctioned. Network connectivity to the remote adapter may have been lost. A remote node may have gone down. The Topology Services daemon on the remote node may have been blocked. If the problem is with the local adapter, an error log entry of type **TS_LOC_DOWN_ST** should follow in a few seconds. Information on the remote node should be collected to obtain a better picture of what failure has occurred.
TS_DMS_WARNING_ST	INFO	**Explanation:** The Dead Man Switch timer is close to triggering. **Details:** This entry indicates that the Dead Man Switch has been reset with a small time-to-trigger value left on the timer. This means that the system is in a state where the Dead Man Switch timer is close to triggering. This condition affects the node where the error log entry appears. If steps are not taken to correct the problem, the node may be brought down by the Dead Man Switch timer. This entry is logged on each occurrence of the condition. Some possible causes are outlined. Detailed fields contain the amount of time remaining in the Dead Man Switch timer and also the interval to which the Dead Man Switch timer is being reset. Program **/usr/sbin/rsct/bin/hatsdmsinfo** displays the latest time-to-trigger values and the values of time-to-trigger that are smaller than a given threshold. Small time-to-trigger values indicate that the Dead Man Switch timer is close to triggering.
TS_DUPNETNAME_ER	PERM	**Explanation:** Duplicated network name in **machines.lst** file. **Details:** This entry indicates that a duplicate network name was found by the Topology Services daemon while reading the **machines.lst** configuration file. This entry refers to a particular instance of Topology Services on the local node. Other nodes may be affected by the same problem, since the **machines.lst** file is the same on all nodes. If this problem occurs at startup time, the daemon exits. Standard fields indicate that a duplicate network name was found in the **machines.lst** file. Detail Data fields show the name that was duplicated.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_FD_INVAL_ADDR_ST	PERM	**Explanation:** An adapter is not configured or has an address outside the cluster configuration. **Details:** This entry indicates that a given adapter in the cluster configuration is either not configured, or has an address which is outside the cluster configuration. This entry affects the local node, and causes the corresponding adapter to be considered down. Detailed data fields show the interface name, current address of the interface, and expected boot-time address. Probable causes for the problem are: • There is a mismatch between the cluster adapter configuration and the actual addresses configured on the local adapters. • The adapter is not correctly configured. Save the output of the command **netstat -in**. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center if the source of the problem cannot be found.
TS_FD_INTFC_NAME_ST	PERM	**Explanation:** An interface name is missing from the adapter configuration. **Details:** The Topology Services startup script reads information from the cluster configuration, containing for each adapter its address, boot-time interface name, and node number. This error entry is created when the interface name information is missing. This usually points to a problem when generating the adapter configuration. The detailed data fields contain the address in the Topology Services configuration and the interface name which has been ″assigned″ to the adapter by the Topology Services daemon. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center. This problem, in most of the cases, will not prevent Topology Services from correctly monitoring the adapter. However, internal problems may occur if a subsequent Topology Services refresh.
TS_HAIPDUP_ER	PERM	**Explanation:** IP address duplication in Topology Services configuration file. **Details:** This entry indicates that Topology Services was not able to start or refresh because the same IP address appeared twice in the configuration. This entry refers to a particular instance of Topology Services on the local node, but the problem may affect all the nodes. If this problem occurs at startup time, the daemon exits. Standard fields indicate that the same IP address appeared twice in the Topology Services **machines.lst** configuration file. Detail Data fields show the node number of one of the nodes hosting the duplicated address and the duplicated IP address. Information about the cause of the problem may not be available once the problem is cleared.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_HALOCAL_ER	PERM	**Explanation:** Local node missing in Topology Services configuration file. **Details:** Standard fields indicate that the local node was not present in the **machines.lst** file. This is a problem with the cluster configuration.
TS_HANODEDUP_ER	PERM	**Explanation:** Node number duplicated in Topology Services configuration file. **Details:** This entry indicates that Topology Services was not able to start or refresh because the same node appeared twice on the same network. This entry refers to a particular instance of Topology Services on the local node, but the problem should affect all the nodes. If this problem occurs at startup time, the daemon exits. Standard fields indicate that the same node appeared twice in the same network in the Topology Services **machines.lst** configuration file. Detail Data fields show the interface name of one of the adapters and the node number that appears twice. Information about the cause of the problem may not be available once the problem is cleared.
TS_IOCTL_ER	PERM	**Explanation:** An **ioctl** call failed. **Details:** This entry indicates that an **ioctl()** call used by the Topology Services daemon to obtain local adapter information failed. This is a possible operating system-related problem. The Topology Services daemon issued an **ioctl()** call to obtain information about the network adapters currently installed on the node. If this calls fails, there is a potential problem in the operating system. The Topology Services daemon exits. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.
TS_IPADDR_ER	PERM	**Explanation:** Cannot convert IP address in dotted decimal notation to a number. **Details:** This entry indicates that an IP address listed in the **machines.lst** configuration file was incorrectly formatted and could not be converted by the Topology Services daemon. If this problem occurs at startup time, the daemon exits. Standard fields indicate that the daemon was unable to interpret an IP address listed in the **machines.lst** file. The Detail Data fields contain the given IP address in dotted decimal notation and the node number where the address was found. The problem may be that the file system where the **run** directory is located is corrupted, or information in the cluster configuration is not correct. The **machines.lst** file is kept in the daemon ″run″ directory (**/var/ct/***cluster_name***/run/cthats**). The file is overwritten each time the subsystem is restarted. A copy of the file is kept in the startup script’s log file, **/var/ct/***cluster_name***/log/cthats/cthats.***cluster_name*. A number of instances (currently 7) of this log file is kept, but the information is lost if many attempts are made to start the subsystem.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_KEYS_ER	PERM	**Explanation**: Topology Services startup script cannot obtain security key information using the **/usr/sbin/rsct/bin/ctmsskf** command. **Details**: This entry indicates that command **/usr/sbin/rsct/bin/ctmsskf**, run by the Topology Services startup script on the control workstation, was unable to retrieve the Topology Services key file. This error occurs when the startup script is running on the control workstation during initial startup or refresh. When this error occurs, all Topology Services daemons in the system partition will terminate their operations and exit. Diagnosing this problem requires collecting data only on the control workstation. In PSSP, the pathname of Topology Services DCE key file is **/spdata/sys1/keyfiles/rsct/***syspar_name***/hats**, where *syspar_name* is the name of the SP system partition. (the **hats** portion of the pathname can be redefined if file **/spdata/sys1/spsec/spsec_overrides** was used to override default DCE file names). The converted key file is located at **/var/ha/run/hats.***syspar_name***/hats_cts**. Standard fields indicate that the **ctmsskf** command, invoked by the startup script, was unable to retrieve the Topology Services key file, and present possible causes. Detail Data fields contain the return code of command **ctmsskf** and the location of the startup script log. The error message returned by command **ctmsskf** is in the startup script log. In PSSP, this error typically indicates problems in DCE. For DCE configuration problems, see the configuration log file **/opt/dcelocal/etc/cfgdce.log**. For other DCE problems, see log files in the **/opt/dcelocal/var/svc** directory. The problem may also occur in a RSCT peer domain, if security is enabled.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_LATEHB_PE	PERF	**Explanation:** Late in sending heartbeat to neighbors. **Details:** This entry indicates that the Topology Services daemon was unable to run for a period of time. This entry refers to a particular instance of the Topology Services daemon on the local node. The node that is the Downstream Neighbor may perceive the local adapter as dead and issue a **TS_DEATH_TR** error log entry. A node's Downstream Neighbor is the node whose IP address is immediately lower than the address of the node where the problem was seen. The node with the lowest IP address has a Downstream Neighbor of the node with the highest IP address. Standard fields indicate that the Topology Services daemon was unable to send messages for a period of time. Detail Data fields show how many seconds late the daemon was in sending messages. This entry is created when the amount of time that the daemon was late in sending heartbeats is equal to or greater than the amount of time needed for the remote adapter to consider the local adapter as down. Data about the reason for the Topology Services daemon being blocked is not usually kept, unless system tracing is being run on the node. The Service log file keeps information about Topology Services events happening on the node at the time the daemon was blocked. See "Topology Services service log" on page 225. Refer to the ″Node appears to go down and then up a few/several seconds later″ symptom in "Error symptoms, responses, and recoveries" on page 244.
TS_LIBERR_EM	PEND	**Explanation:** Topology Services client library error. **Details:** This entry indicates that the Topology Services library had an error. It refers to a particular instance of the Topology Services library on the local node. This problem will affect the client associated with the library (RSCT Event Manager or more likely RSCT Group Services). Standard fields indicate that the Topology Services library had an error. Detail Data fields contain the error code returned by the Topology Services API. Data needed for IBM Service to diagnose the problem is stored in the Topology Services daemon service log, located at **/var/ct/***cluster_name***log/cthats/cthats.***DD.hhmmss* The Group Services daemon (the probable client connected to the library) is likely to have exited with an assert and to have produced an error log entry with template **GS_TS_RETCODE_ER**. Refer to "Diagnosing Group Services problems" on page 267 for a list of the information to save. See "Information to collect before contacting the IBM Support Center" on page 229 and contact the IBM Support Center.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_LOC_DOWN_ST	INFO	**Explanation:** Local adapter down. **Details:** This entry indicates that one of the local adapters is down. This entry refers to a particular instance of the Topology Services daemon on the local node. Standard fields indicate that a local adapter is down. Detail Data fields show the interface name, adapter offset (index of the network in the **machines.lst** file), and the adapter address according to Topology Services. This address may differ from the adapter's actual address if the adapter is incorrectly configured. Information about the source of the problem may be lost after the condition is cleared. Possible problems are: • The adapter may have malfunctioned. • The adapter may be incorrectly configured. See entry for **TS_UNS_SIN_TR**. • There is no other adapter functioning in the network. • Connectivity has been lost in the network. • A problem in Topology Services' adapter health logic. Perform these steps: 1. Verify that the address of the adapter listed in the output of `ifconfig` *`interface_name`* is the same as the one shown in this error log entry. If they are different, the adapter has been configured with an incorrect address. 2. If the output of the **ifconfig** command does not show the **UP** flag, this means that the adapter has been forced down by the command: `ifconfig` *`interface_name`* `down` 3. Issue the command **netstat -in** to verify whether the `receive` and `send` counters are being incremented for the given adapter. On AIX, the counters are the numbers below the `Ipkts` (receive) and `Opkts` (send) columns. If both counters are increasing, the adapter is likely to be working and the problem may be in Topology Services. 4. Issue the **ping** command to determine whether there is connectivity to any other adapter in the same network. If **ping** receives responses, the adapter is likely to be working and the problem may be in Topology Services. 5. Refer to "Operational test 4 - Check address of local adapter" on page 237.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_LOGFILE_ER	PERM	**Explanation:** The daemon failed to open the log file. **Details:** This entry indicates that the Topology Services daemon was unable to open its log file. Standard fields indicate that the daemon was unable to open its log file. Detail Data fields show the name of the log file. The situation that caused the problem may clear when the file system problem is corrected. The Topology Services daemon exits. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.
TS_LONGLINE_ER	PERM	**Explanation:** The Topology Services daemon cannot start because the **machines.lst** file has a line that is too long. **Details:** This entry indicates that the Topology Services daemon was unable to start because there is a line which is too long in the **machines.lst** configuration file. This entry refers to a particular instance of Topology Services on the local node. If this problem occurs at startup time, the daemon exits. The problem is likely to affect other nodes, since the **machines.lst** file should be the same at all nodes. Standard fields indicate that the daemon was unable to start because the **machines.lst** configuration file has a line longer than 80 characters. Detail Data fields show the path name of the **machines.lst** configuration file. It is possible that the network name is too long, or there is a problem in the **/var/ct** file system.
TS_LSOCK_ER	PERM	**Explanation:** The daemon failed to open a listening socket for connection requests. **Details:** This entry indicates that the Topology Services daemon was unable to open a socket connection to communicate with its clients. Standard fields indicate that the daemon was unable to open the socket. Detail Data fields show the operation being attempted at the socket (in English) and the system error value returned by the system call. The situation that caused the problem may clear with a reboot. The **netstat** command shows the sockets in use in the node. The Topology Services daemon exits. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.
TS_MACHLIST_ER	PERM	**Explanation:** The Topology Services configuration file cannot be opened. **Details:** This entry indicates that the Topology Services daemon was unable to read its **machines.lst** configuration file. Standard fields indicate that the daemon was unable to read the **machines.lst** file. Detail Data fields show the path name of the file. Information about the cause of the problem is not available after the condition is cleared. If this problem occurs at startup time, the daemon exits. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_MIGRATE_ER	PERM	**Explanation:** Migration-refresh error. **Details:** This entry indicates that the Topology Services daemon has found a problem during a migration-refresh. The migration-refresh is a refresh operation issued at the end of an HACMP node by node migration, when the last node is moved to the newer release. The problem may be caused by the information placed on the Global ODM when the migration protocol is complete. This entry refers to a particular instance of the Topology Services daemon on the local node. It is likely that some of the other nodes have a similar problem. Standard fields indicate that the Topology Services daemon encountered problems during a migration-refresh. HACMP may have loaded incorrect information into the Global ODM. Data read by the Topology Services startup script is left on the Topology Services run directory and will be overwritten in the next refresh or startup operation. The data in the “run” directory should be saved. The Topology Services “Service” log file has a partial view of what was in the Global ODM at the time of the refresh operation.
TS_MISCFG_EM	PEND	**Explanation:** Local adapter incorrectly configured. **Details:** This entry indicates that one local adapter is either missing or has an address that is different from the address that Topology Services expects. Standard fields indicate that a local adapter is incorrectly configured. Detail Data fields contain information about the adapter, such as the interface name, adapter offset (network index in the **machines.lst** file), and expected address. Possible sources of the problem are: • The adapter may have been configured with a different IP address. • The adapter is not configured. • Topology Services was started after a “Force Down” in HACMP. This entry is created on the **first occurrence** of the condition. No data is stored about the condition after the problem is cleared. Use the interface name in the error report to find the adapter that is incorrectly configured. Command: **ifconfig** *interface_name* displays information about the adapter.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_NIM_DIED_ER	PERM	**Explanation:** One of the NIM processes terminated abnormally. **Details:** This entry is created when one of the NIM (Network Interface Modules)- processes used by Topology Services to monitor the state of each adapter, terminates abnormally. When a NIM terminates, the Topology Services daemon will restart another. If the replacement NIM also terminates quickly, no other NIM will be started, and the adapter will be flagged as down. Detailed data fields show: • Process exit value, if not terminated with a signal (A value from 1 to 99), will be an 'errno' value from invoking the NIM process. • Signal number (0: no signal). • Whether a core file was created (1: core file; 0: no core file). • Process id (PID). • Interface name being monitored by the NIM. • Path name of NIM executable file. See "Information to collect before contacting the IBM Support Center" on page 229 and contact the IBM Support Center.
TS_NIM_ERROR_INTERNAL_ER	PERM	**Explanation:** An internal error occurred at the NIM process. **Details:** This entry indicates that there was an error in the execution of the NIM. This could be a serious enough error that will cause the NIM process to exit. It could also be a less severe error. In case the NIM exits, a new NIM will be respawned in its place. The standard fields describe the most likely causes for the problem: an internal ″assert″ or some internal limit was exceeded. The detailed fields show the error level (serious, error, information), an error description, some error data, and the interface name to which the NIM is associated.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_NIM_ERROR_MSG_ER	PERM	**Explanation:** Too many incorrect messages exchanged between the Topology Services daemon and the NIM. **Details:** This entry indicates that the daemon was unable to interpret messages sent to it by the NIM via the Unix-domain socket. The probable causes for this are: • The NIM and the daemon lost the ″frame synchronization″ on the packets flowing through the Unix-domain socket. This causes the daemon to interpret packets incorrectly. • The daemon and the NIM are using different versions of the protocol, resulting in the daemon being unable to interpret messages sent by the NIM. • The NIM has an internal problem that causes it to send invalid packets to the daemon. After the daemon has received a number of messages from the NIM that it cannot handle, the daemon will issue this error log entry and then terminate the connection with the NIM. As soon as the NIM terminates, the daemon will start a new one. The standard fields describe the problem and offers some possible causes. The detailed fields show the last kind of error received, the last packet type received, the error count, the message's protocol version and the daemon's protocol version, and finally the interface name to which the NIM is associated.
TS_NIM_ERROR_RDWR_ER	PERM	**Explanation:** The NIM encountered a read or write error when sending data to or receiving data from the network adapter or non-IP device. **Details:** This entry indicates that there were I/O errors when trying to send data to the adapter or device, or when trying to receive data from it. The most likely causes are that the adapter is down (in the ″ifconfig″ sense) or has been unconfigured. For non-IP devices, it is possible that the remote side of the connection is no longer active. The standard fields present the possible causes for the problem. The detailed fields indicate whether the problem was a write or read error, and also some details about the error. For example, for errors when sending data, the detailed fields show the ″errno″ value and the number of times the error occurred. For RS232 links, an error entry will be issued if there are too many checksum errors. In this case the error count will be shown. The interface name to which the NIM is associated is also shown.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_NIM_ERROR_STUCK_ER	PERM	**Explanation**: One of the threads in a NIM process was blocked. **Details**: This entry indicates that a thread in one of the NIM processes did not make progress and was possibly blocked for a period of time. Depending on which of the threads was blocked and for how long, the adapter corresponding to the NIM process may be erroneously considered down. The standard fields indicate that the NIM was blocked and present possible causes and actions to prevent the problem from reoccurring. The problem may have been caused by resource starvation at the node, or possibly excessive I/O activity. The detailed fields show the name of the thread which was blocked, the interval in seconds during which the thread was blocked, and the interface name which is associated with this instance of the NIM. If there is no false adapter down event caused by the blockage then no action is needed. If there is then the cause for the blockage needs to be understood. To investigate the problem, follow the same steps as those taken to investigate the error entry TS_LATEHB_PE.
TS_NIM_ERROR_TRAF_ER	PERM	**Explanation**: The NIM has detected too much traffic being received from the adapter or being sent to the adapter. **Details**: This entry indicates either too much data has been received from the adapter or (more likely) the NIM detected that more data is being sent by the Topology Services daemon than what can be pumped into the adapter. This is more likely to happen with slow non-IP connections. Usually any device can support the ″normal traffic″ sent for heartbeating. However, in situations where Group Services protocols need to be run over these slow links then it is possible for this error to occur. If this error occurs repeatedly and a ″slow″ device is being used for heartbeating then a faster device should be pursued. The standard fields describe the problem and possible causes. The detailed fields indicate whether the problem occurred when sending or receiving data. For send errors, the size of the packet queue length at the NIM is shown. The interface name to which the NIM is associated is also shown.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_NIM_NETMON_ERROR_ER	PERM	**Explanation:** An error occurred in the netmon library, used by the NIM (Network Interface Module) - processes used by Topology Services to monitor the state of each adapter, in determining whether the local adapter is alive. **Details:** This entry is created when there is an internal error in the netmon library. As a result, the local adapter will be flagged as down, even though the adapter may still be working properly. A possible cause for the problem (other than a problem in the library) is the presence of some non-supported adapter in the cluster configuration. Detailed data fields show: • Errno value. • Error code from netmon library. • Function name in library that presented a problem. • Interface name being monitored. See "Information to collect before contacting the IBM Support Center" on page 229 and contact the IBM Support Center. It is important to collect the information as soon as possible, since log information for the netmon library is kept in log files that may wrap within 30 minutes.
TS_NIM_OPEN_ERROR_ER	PERM	**Explanation:** NIM (Network Interface Module) - processes used by Topology Services to monitor the state of each adapter, failed to connect to the local adapter that it is supposed to monitor. **Details:** This entry is created when the NIM is unable to connect to the local adapter that needs to be monitored. As a result, the adapter will be flagged as down, even though the adapter might still be working properly. Detailed data fields show: • Interface name. • Description 1: description of the problem. • Description 2: description of the problem. • Value 1 - used by the IBM Support Center. • Value 2 - used by the IBM Support Center. Some possible causes for the problem are: • NIM process was blocked while responding to NIM open command. • NIM failed to open non-IP device. • NIM received an unexpected error code from a system call. See "Information to collect before contacting the IBM Support Center" on page 229 and contact the IBM Support Center.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_NODENUM_ER	PERM	**Explanation:** The local node number is not known to Topology Services. **Details:** This entry indicates that Topology Services was not able to find the local node number. Standard fields indicate that the daemon was unable to find its local node number. The Topology Services daemon exits. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.

TS_NODEUP_ST	INFO	**Explanation:** Remote nodes that were previously down were seen as up by Topology Services. This is an indication that the Topology Services daemon detected one or more previously down nodes as being up. It refers to a particular instance of the Topology Services daemon.
		
		Details: In case the same nodes were seen as dead a short time before, data should be collected on the remote nodes. Standard fields indicate that remote nodes were seen as up and present possible causes. Detailed fields contain, in the section, a reference to the entry where the same nodes were seen as dead. If these nodes were seen as down before at different times, the reference code will be for one of these instances.
		
		The Detail Data also contains the path name of a file which stores the numbers of the nodes that were seen as up, along with the error id for the error log entry where each node was seen as dead previously. The file with the node numbers may eventually be deleted by the system. The file is located in: **/var/adm/ffdc/dumps/sh.***.
		
		If the same nodes were recently seen as dead (follow the REFERENCE CODE), examine the remote nodes for the reason why the nodes were temporarily seen as dead. This entry is logged when a remote node is seen as alive. The same node may have been seen as dead some time ago. If so, the **TS_NODEUP_ST** will have, as part of the Detail Data, a location of a file whose contents are similar to:
		
		`.ZOWYB/Z5Kzr.zBI14tVQ7...................`
		`1`

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_OFF_LIMIT_ER	PERM	**Explanation:** Number of network offsets exceeds Topology Services limit. **Details:** This entry is created whenever the number of adapters and networks in the cluster configuration exceeds the Topology Services daemon's internal limit for maximum number of ″heartbeat rings″ of 16. Notice that a single cluster network may map to multiple ″heartbeat rings″. This will happen when a node has multiple adapters in the same network, since a heartbeat ring is limited to a single adapter per node. If this error occurs, a number of adapters and networks in the configuration may remain unmonitored by Topology Services. The detailed data fields contain the first network in the configuration to be ignored and the maximum number of networks allowed. When attempting to eliminate the problem, initially focus on the nodes that have the most adapters in the configuration, and proceed to remove some adapters from the configuration.
TS_REFRESH_ER	PERM	**Explanation:** Topology Services refresh error. **Details:** This entry indicates that a problem occurred during a Topology Services refresh operation. A refresh operation can be a result of a configuration change, such as adding or deleting a node in the cluster, or changing characteristics of a communication group. It can also be the result of the **cthatstune -r** command. In HACMP/ES, a refresh occurs as a result of synchronizing topology changes in a cluster. This entry refers to a particular instance of the Topology Services daemon on the local node. On HACMP, or in an RSCT peer domain, the problem may have occurred in other nodes as well. Standard fields indicate that a refresh error occurred. The machines.lst file has some incorrect information. The problem is probably created during a migration-refresh on an HACMP node by node migration. Data used to build the machines.lst file is stored in the daemon's "run" directory and may be lost if Topology Services is restarted or a new refresh is attempted. More details about the problem are in the User log file. See "Topology Services user log" on page 227. Additional details are stored in the Service log. See "Topology Services service log" on page 225. If this problem occurs at startup time, the Topology Services daemon may exit. See "Information to collect before contacting the IBM Support Center" on page 229 and contact the IBM Support Center.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_RSOCK_ER	PERM	**Explanation:** The daemon failed to open socket for peer daemon communication. **Details:** This entry indicates that the Topology Services daemon was unable to open a UDP socket for communication with peer daemons in other nodes. Standard fields indicate that the daemon was unable to open the socket. Detail Data fields describe the operation being attempted at the socket (in English), the reason for the error, the system error value, and the port number. The port number may be in use by either another subsystem or by another instance of the Topology Services daemon. If the SRC subsystem loses its connection to the Topology Services daemon, the SRC may erroneously allow a second instance of the daemon to be started, leading to this error. The situation that caused the problem may clear with a node reboot. Follow the procedures described for the ″Nodes or adapters leave membership after refresh″ symptom in “Error symptoms, responses, and recoveries” on page 244 to find a possible Topology Services daemon running at the node and stop it. If no process is found that is using the peer socket, see “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center. Include also a System Dump.
TS_SECURITY_ST	INFO	**Explanation:** Authentication failure in Topology Services. **Details:** This entry indicates that the Topology Services daemon cannot authenticate a message from one of the peer daemons running in a remote node. This entry refers to a particular instance of the Topology Services daemon on the local node. The node which is sending these messages must also be examined. Standard fields indicate that a message cannot be authenticated. Detail Data fields show the source of the message. The possible problems are: • There is an attempt at a security breach. • The Time-Of-Day clocks in the nodes are not synchronized. • There are stale packets flowing through the network. • IP packets are being corrupted. • The security key file is not in sync across all nodes in the domain. An entry is created the first time a message cannot be authenticated. After that, entries are created less frequently. Information about the network must be collected while the messages are still being received. The command **tcpdump** should be used to examine the packets arriving at the node. Perform the following steps: 1. Examine the output of the **lssrc -ls hats** command (PSSP) or **lssrc -ls cthats** (RSCT peer domain) on the local node and on the node sending the message. Look for field “Key version” in the output and check whether the numbers are the same on both nodes. 2. Check that the key file is the same in all the nodes in the domain.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_SECURITY2_ST	INFO	**Explanation:** More authentication failures in Topology Services. **Details:** This entry indicates that there have been additional incoming messages that could not be authenticated. For the first such message, error log entry **TS_SECURITY_ST** is created. If additional messages cannot be authenticated, error log entries with label **TS_SECURITY2_ST** are created less and less frequently. The standard fields indicate that incoming messages cannot be authenticated. The detailed fields show an interval in seconds and the number of messages in that interval that could not be authenticated. For more details and diagnosis steps, see the entry for the **TS_SECURITY_ST** label.
TS_SEMGET_ER	PERM	**Explanation:** Cannot get shared memory or semaphore segment. This indicates that the Topology Services daemon was unable to start because it could not obtain a shared memory or semaphore segment. This entry refers to a particular instance of the Topology Services daemon on the local node. The daemon exits **Details:** Standard fields indicate that the daemon could not start because it was unable to get a shared memory or a semaphore segment. The Detail Data fields contain the key value and the number of bytes requested for shared memory, or the system call error value for a semaphore. The reason why this error has occurred may not be determined if the subsystem is restarted and this error no longer occurs.
TS_SERVICE_ER	PERM	**Explanation:** Unable to obtain port number from the **/etc/services** file. **Details:** This entry indicates that the Topology Services daemon was unable to obtain the port number for daemon peer communication from **/etc/services**. This entry refers to a particular instance of the Topology Services daemon on the local node. The daemon exits. Other nodes may be affected if their **/etc/services** have similar contents as that on the local node. Standard fields indicate that the daemon was unable to obtain the port number from **/etc/services**. Detail Data fields show the service name used as search key to query **/etc/services**.
TS_SHMAT_ER	PERM	**Explanation:** Cannot attach to shared memory segment. **Details:** This entry indicates that the Topology Services daemon was unable to start because it could not attach to a shared memory segment. Standard fields indicate that the daemon could not start because it was unable to attach to a shared memory segment. The daemon exits. The Detail Data fields contain the shared memory identifier and number of bytes requested. The reason why the error occurred may not be found if the subsystem is restarted and the same error does not occur.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_SHMEMKEY_ER	PERM	**Explanation:** Cannot get IPC key. **Details:** This indicates that the Topology Services daemon was unable to start because it could not obtain an IPC key. This refers to a particular instance of the Topology Services daemon on the local node. The daemon exits. Standard fields indicate that the daemon could not start because it was unable to obtain an IPC key. The Detail Data fields contain the path name of the UNIX-domain socket used for daemon-client communication. This path name is given to the **ftok()** subroutine in order to obtain an IPC key. This entry is created when the UNIX-domain socket file has been removed. The reason why this error has occurred may not be determined if the subsystem is restarted and this error no longer occurs.
TS_SHMGET_ER	PERM	See TS_SEMGET_ER
TS_SP_DIR_ER	PERM	**Explanation:** Cannot create directory. **Details:** This entry indicates that the Topology Services startup script **cthats** was unable to create one of the directories it needs for processing. Standard fields indicate that a directory could not be created by the startup script **cthats**. Detail Data fields show the directory that could not be created. Information about the cause of the problem may not be available once the problem is cleared.
TS_SPIPDUP_ER	PERM	See TS_HAIPDUP_ER
TS_SPLOCAL_ER	PERM	See TS_HALOCAL_ER
TS_SPNODEDUP_ER	PERM	See TS_HANODEDUP_ER
TS_START_ST	INFO	**Explanation:** The Topology Services daemon has started. This is an indication that the Topology Services daemon has started. This entry refers to a particular instance of the Topology Services daemon on the local node. **Details:** Standard fields indicate that the daemon started. The Topology Services subsystem was started by a user or during system boot. Detail Data will be in the language where the **errpt** (or **fcslogrpt**) command is run. The Detail Data contains the location of the log and run directories and also which user or process started the daemon.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_STOP_ST	INFO	**Explanation:** The Topology Services daemon has stopped. This is an indication that the Topology Services daemon has stopped. This entry refers to a particular instance of the Topology Services daemon on the local node. **Details:** The Topology Services subsystem shutdown was caused by a signal sent by a user or process. Standard fields indicate that the daemon stopped. The standard fields are self-explanatory. If stopping the daemon is not desired, you must quickly understand what caused this condition. If the daemon was stopped by the SRC, the word ″SRC″ is present in the Detail Data . The REFERENCE CODE field in the Detail Data section refers to the error log entry for the start of Topology Services. Detail Data is in English. Detail Data fields point to the process (SRC) or signal that requested the daemon to stop.
TS_THATTR_ER	PERM	**Explanation:** Cannot create or destroy a thread attributes object. **Details:** This entry indicates that Topology Services was unable to create or destroy a thread attributes object. Standard fields indicate that the daemon was unable to create or destroy a thread attributes object. Detail Data fields show which of the Topology Services threads was being handled. The Topology Services daemon exits. See “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.
TS_THCREATE_ER	PERM	**Explanation:** Cannot create a thread. **Details:** This entry indicates that Topology Services was unable to create one of its threads. Standard fields indicate that the daemon was unable to create a thread. Detail Data fields show which of the Topology Services threads was being created.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_THREAD_STUCK_ER	PERM	**Explanation:** Main thread is blocked. Daemon will exit. **Details:** This entry indicates that the Topology Services daemon will exit because its main thread was blocked for longer than a pre-established time threshold. If the main thread remains blocked for too long, it is possible that the node is considered dead by the other nodes. The main thread needs to have timely access to the CPU, otherwise it would fail to send ″heartbeat″ messages, run adapter membership protocols, and notify Group Services about adapter and node events. If the main thread is blocked for too long, the daemon exits with a core dump, to allow debugging of the cause of the problem. This entry refers to a particular instance of Topology Services running on a node. The standard fields indicate that the Topology Services daemon will exit because the main thread was blocked for too long, and explains some of the possible causes. The detailed fields show the number of seconds that the main thread appeared to be blocked, the number of recent page faults involving I/O operations, and the interval in milliseconds where these page faults occurred. If the number of page faults is non-zero, the problem could be related to memory contention. For information about diagnosing and working around the problem in case its root cause is a resource shortage, see “Action 5 - Investigate hatsd problem” on page 247. If a resource shortage does not seem to be a factor, the cause could be a problem in the daemon or in a service invoked by it. Contact the IBM Support Center.

Table 18. Error Log templates for Topology Services (continued)

Label	Type	Description
TS_UNS_SIN_TR	UNKN	**Explanation:** Local adapter in unstable singleton state. **Details:** This entry indicates that a local adapter is staying too long in a singleton unstable state. Though the adapter is able to receive some messages, there could be a problem with it, which may prevent outgoing messages from reaching their destinations. This entry refers to a particular instance of the Topology Services daemon on the local node. Examine the Service log on other nodes to determine if other nodes are receiving messages from this adapter. See "Topology Services service log" on page 225. Standard fields indicate that a local adapter is in an unstable singleton state. Detail Data fields show the interface name, adapter offset (index of the network in the **machines.lst** file), and the adapter address according to Topology Services, which may differ from the adapter's actual address if the adapter is incorrectly configured. The adapter may be unable to send messages. The adapter may be receiving broadcast messages but not unicast messages. Information about the adapter must be collected while the adapter is still in this condition. Issue the commands: **ifconfig** *interface_name* and **netstat -in** and record the output. Perform these steps: 1. Check if the address displayed in the error report entry is the same as the actual adapter address, which can be obtained by issuing this command: **ifconfig** *interface_name*. If they are not the same, the adapter has been configured with the wrong address. 2. Issue command **ping** *address* from the local node for all the other addresses in the same network. If **ping** indicates that there is no reply (for example: `10 packets transmitted, 0 packets received, 100% packet loss`) for all the destinations, the adapter may be incorrectly configured. 3. Refer to "Operational test 6 - Check whether the adapter can communicate with other adapters in the network" on page 238.

Dump information

Topology services provides two dumps, a core dump which is created automatically when certain errors occur, and a **ctsnap** dump which is created manually.

Core dump

There is a core dump generated by the Topology Services daemon. It contains information normally saved in a core dump: user-space data segments for the Topology Services daemon. It refers to a particular instance of the Topology Services daemon on the local node. Other nodes may have a similar core dump. The dump is located in: **/var/ct/***cluster_name***/run/cthats/core**. An approximate size for the core dump file is between 7 and 10MB.

The dump is created automatically when the daemon invokes an **assert()** statement, or when the daemon receives a segmentation violation signal for

accessing its data incorrectly. Forcing **hatsd** to generate a dump is necessary, especially if the daemon is believed to be in a hung state. The dump is created manually by issuing the command:

```
kill -6 pid_of_daemon
```

The *pid_of_daemon* is obtained by issuing: **lssrc -s cthats**.

The dump remains valid as long as the executable file **/usr/sbin/rsct/bin/hatsd** is not replaced. Only the last three core file instances are kept. The core dumps and the executable should be copied to a safe place. To analyze the dump, issue the command:

```
dbx /usr/sbin/rsct/bin/hatsd core_file
```

Good results are similar to the following:

```
Type 'help' for help.
reading symbolic information ...
[using memory image in core]

IOT/Abort trap in evt._pthread_ksleep [/usr/lib/libpthreads.a] at
0xd02323e0 ($t6) 0xd02323e0 (_pthread_ksleep+0x9c) 80410014 lwz r2,0x14(r1)
```

Some of the error results are:

1. This means that the current executable file was not the one that created the core dump.

   ```
   Type 'help' for help.
   Core file program (hatsd) does not match current program (core ignored)
   reading symbolic information ...
   (dbx)
   ```

2. This means that the core file is incomplete due to lack of disk space.

   ```
   Type 'help' for help.
   warning: The core file is truncated. You may need to increase the
   ulimit for file and coredump, or free some space on the filesystem.
   reading symbolic information ...
   [using memory image in core]

   IOT/Abort trap in evt._pthread_ksleep [/usr/lib/libpthreads.a]
    at 0xd02323e0 0xd02323e0 (_pthread_ksleep+0x9c) 80410014
    lwz r2,0x14(r1) (dbx)
   ```

ctsnap dump

This dump contains diagnostic data used for RSCT problem determination in a Unix environment. It is a collection of log files and other trace information used to obtain a global picture of the state of RSCT. The dump is specific to each node. It is located (by default) in the **/tmp/ctsupt** directory. The dump is created by command:

```
/usr/sbin/rsct/bin/ctsnap
```

The command collects data only from the invoking node. Depending on the nature of the problem, it may be necessary to invoke the command from multiple nodes.

The dump is in a **tar- compressed** file. The name of the dump is: **ctsnap.***hostname***.***timestamp***.tar.Z**. Because of this name convention, the dump will not be overwritten by a subsequent invocation of **ctsnap** on the same node.

The **-d** flag can be used to specify the directory where the command will place the dump.

Good results from the **ctsnap** command are indicated by an output similar to the following:

```
.....................................................................
......
```

Error results are indicated by a non-zero exit code from **ctsnap**. A diagnostic message should be in the log file generated by **ctsnap**, which is in the same directory as the dump file.

Contents of the ctsnap dump: The dump is a collection of files archived with the **tar** command and compressed with the **compress** command. Some of these files are copies of daemon log files, and some are the output from certain commands.

This is a partial list of the data items collected:

1. Output of these commands:
   - **netstat** using several options.
   - **ifconfig -a**.
   - **ps -edf|grep -E -e ″IBM|rmc|cthats|cthags″**
   - **lssrc -a | grep -E -e ″rsct|cthats|cthags″**
   - **df /tmp**
   - **lslpp -l ″rsct.*″**
2. Log and run directories:
   - All files in **/var/ct**, including all log files and core files
   - Executable files that correspond to core files
3. Output of component-specific commands, such as:
   - **lssrc -l**
   - **hagsgr**, **hagsns**, **hagsvote** and other Group Services programs
   - **ct_hats_info**, **ct_hags_info**, **ct_topology_info**
   - **ctrmc**

Trace information

ATTENTION - READ THIS FIRST

Do *not* activate this trace facility until you have read this section completely, and understand this material. If you are not certain how to properly use this facility, or if you are not under the guidance of IBM Service, do *not* activate this facility.

Activating this facility may result in degraded performance of your system. Activating this facility may also result in longer response times, higher processor loads, and the consumption of system disk resources. Activating this facility may also obscure or modify the symptoms of timing-related problems.

Consult these logs for debugging purposes. They all refer to a particular instance of the Topology Services daemon running on the local node.

Topology Services service log

This log contains trace information about the activities performed by the daemon. When a problem occurs, logs from multiple nodes will often be needed. These log files must be collected before they wrap or are removed.

The trace is located in: **/var/ct/***cluster_name***/log/cthats/cthats.***DD.hhmmss* where *cluster_name* is the name of the cluster, *DD* is the day of the month when the daemon was started, and *hhmmss* is the time when the daemon was started.

If obtaining logs from all nodes is not feasible, the following is a list of nodes from which logs should be collected:

1. The node where the problem was seen
2. The Group Leader node on each network

   The Group Leader is the node which has the highest IP address on a network.
3. The Downstream Neighbor on each network

   This is the node whose IP address is immediately lower than the address of the node where the problem was seen. The node with the lowest IP address has a Downstream Neighbor of the node with the highest IP address.

Service Log long tracing: The most detailed level of tracing is Service log long tracing. It is started with either the command:

```
traceson -l -s cthats
```

or the command:

```
cthatsctrl -t
```

The long trace is stopped with this command: **tracesoff -s** *subsystem_name*, or **cthatsctrl -o** which causes normal tracing to be in effect. When the log file reaches the maximum line number, the current log is saved in a file with a suffix of **.bak**, and the original file is truncated. When the daemon is restarted, a new log file is created. Only the last five log files are kept.

With service log long tracing, trace records are generated under the following conditions:

- Each message sent or received
- Each adapter that is disabled or re-enabled
- Details of protocols being run
- Details of node reachability information
- Refresh
- Client requests and notifications
- Groups formed, elements added and removed

Data in the Service log is in English. Each Service log entry has this format:

```
date     daemon name     message
```

Adapters are identified by a pair:

```
(IP address:incarnation number)
```

Groups are identified by a pair:

```
(IP address of Group Leader:incarnation number of group)
```

Long tracing should be activated on request from IBM Service. It can be activated (just for about one minute, to avoid overwriting other data in the log file), when the error condition is still present.

Service Log normal tracing: Service log normal tracing is the default, and is always running. There is negligible impact if no node or adapter events occur on the system. An adapter death event may result in approximately 50 lines of log information for the Group Leader and ″mayor″ nodes, or up to 250 lines for the Group Leader and ″mayor″ nodes on systems of approximately 400 nodes. All other nodes will produce less than 20 lines. Log file sizes can be increased as described in “Changing the service log size”.

With normal tracing, trace records are generated for these conditions:

- Each adapter that is disabled or re-enabled
- Some protocol messages sent or received
- Refresh
- Client requests and notifications
- Groups formed, members added and removed

No entries are created when no adapter or node events are happening on the system.

With normal tracing, the log trimming rate depends heavily on the frequency of adapter or node events on the system. The location of the log file and format of the information is the same as that of the long tracing described previously.

If the Service log file, using normal tracing, keeps growing even when no events appear to be happening on the system, this may indicate a problem. Search for possible entries in the syslog or in the User log. See “Topology Services user log”.

Changing the service log size: The long trace generates approximately 10KB of data per minute of trace activity. By default, log files have a maximum of 5000 lines, which will be filled in 30 minutes or less if long tracing is requested. To change the log file size, issue the **cthatstune** command on any node:

```
cthatstune -l new_max_lines -r
```

The full path name of this command is: **/usr/sbin/rsct/bin/cthatstune**.

For example, **cthatstune -l 10000 -r** changes the maximum number of lines in a log file to 10000. The **-r** flag causes the Topology Services subsystem to be refreshed in all the nodes.

Topology Services user log

The Topology Services user log contains error and informational messages produced by the daemon. This trace is always running. It has negligible impact on the performance of the system, under normal circumstances.

The trace is located in: **/var/ct/***cluster_name***log/cthats/cthats.***DD.hhmmss.lang*, where *cluster_name* is the name of the cluster, *DD* is the day of the month when the daemon was started, *hhmmss* is the time when the daemon was started, and *lang* is the language used by the daemon.

Data in the user log is in the language where the daemon is run, which is the node’s administrative language. Messages in the user log have a catalog message number, which can be used to obtain a translation of the message in the desired language.

The size of the log file is changed using the same commands that change the size of the service log. Truncation of the log, saving of log files, and other considerations are the same as for the service log.

Each user log entry has this format:

```
date     daemon name       message
```

Adapters are identified by a pair:

```
(IP address:incarnation number)
```

Groups are identified by a pair:

```
(IP address of Group Leader:incarnation number of group)
```

The main source for diagnostics is the error log. Some of the error messages produced in the user log occur under normal circumstances, but if they occur repeatedly they indicate an error. Some error messages give additional detail for an entry in the error log. Therefore, this log file should be examined when an entry is created in the system error log.

cthats script log

This is the Topology Services startup script log. It contains configuration data used to build the **machines.lst** configuration file. This log also contains error messages if the script was unable to produce a valid **machines.lst** file and start the daemon. The startup script is run at subsystem startup time and at refresh time. This log refers to a particular instance of the Topology Services script running on the local node.

The size of the file varies according to the size of the machine. It is about 500 bytes in size for a three-node system, and is larger for systems with more nodes. The trace runs whenever the startup script runs. The trace is located in: **/var/ct/***cluster_name***log/cthats/cthats.***cluster_name*. A new instance of the **cthats** startup script log is created each time the script starts. A copy of the script log is made just before the script exits. Only the last seven instances of the log file are kept, and they are named *file***.1** through *file***.7**. Therefore, the contents of the log must be saved before the subsystem is restarted or refreshed many times.

The *file***.1** is an identical copy of the current startup script log. At each startup, *file***.1** is renamed to *file***.2**; *file***.2** is renamed to *file***.3**, and so on. Therefore, the previous *file***.7** is lost.

Entries in the startup script log are kept both in English and in the node's language (if different). Trace records are created for these conditions:

- The **machines.lst** file is retrieved
- The **machines.lst** file is built using information propagated by the configuration resource manager.
- An error is encountered that prevents the **cthats** script from making progress.

There is no fixed format for the records of the log. The following information is in the file:

- The date and time when the **cthats** script started running
- A copy of **machines.lst** file generated
- The date and time when the **cthats** script finished running

- If the script was called for a refresh operation, the output of the **refresh** command is included in the log file.

The main source for diagnostics is the error log. The **cthats** script log file should be used when the error log shows that the startup script was unable to complete its tasks and start the daemon.

Network Interface Module (NIM) log

This log contains trace information about the activities of the Network Interface Modules (NIMs), which are processes used by the Topology Services daemon to monitor each network interface. These logs need to be collected before they wrap or are removed.

The trace is located in:
/var/ct/*cluster_name***/log/cthats/nim.cthats.***interface_name*[**.00***n*], where *interface_name* is the network interface name and **00***n* is a sequence number of 001, 002, or 003. These three logs are always kept. Log file 003 is overwritten by 002, 002 is overwritten by 001, and 001 is overwritten by 003. The current log file does not have a **00n** suffix.

Trace records are generated under the following conditions:

1. A connection with a given adapter is established.
2. A connection with a given adapter is closed.
3. A daemon has sent a command to start or stop heartbeating.
4. A daemon has sent a command to start or stop monitoring heartbeats.
5. A local adapter goes up or down.
6. A message is sent or received.
7. A heartbeat from the remote adapter has been missed

Data in the NIM log is in English only. The format of each message is:

```
time-of-day    message
```

An instance of the NIM log file will wrap when the file reaches around 200kB. Normally, it takes around 10 minutes to fill an instance of the log file. Since 3 instances are kept, the NIM log files needs to be saved within 30 minutes of when the adapter-related problem occurred.

Information to collect before contacting the IBM Support Center

The following information needs to be collected from the node that presents the problem. For connectivity-related problems, the same information is needed from the other nodes. If collecting data from all the nodes is not feasible, data should be collected from at least the following nodes:

1. The node's Downstream Neighbor on all networks. This is the node whose IP address is immediately lower than the address of the node where the problem was seen. The node with the lowest IP address has a Downstream Neighbor of the node with the highest IP address.
2. The Group Leader node, which is the node with the highest IP address in the network.

Collect the output of command:

```
/usr/sbin/rsct/bin/ctsnap
```

Refer to "ctsnap dump" on page 224 for more information on the **ctsnap** command.

For problems related to connectivity and adapter status, use command **tcpdump** to collect a sample of the traffic on the network. Invoke command:

```
tcpdump -n -x [-i interface name]  > output_file
```

and then after at least 30 seconds (or as instructed by the IBM Support Center), terminate it with a signal.

Save the output of the command, along with the data collected by **ctsnap**.

See Chapter 7, “How to contact the IBM Support Center” on page 295.

Diagnostic procedures

These tests verify the installation, configuration and operation of Topology Services.

Installation verification test

This test determines whether RSCT has been successfully installed.Perform the following steps:

1. Verify if RSCT has been installed. Issue the command:

   ```
   lslpp -l 'rsct.*'
   ```

   The expected output is:

   ```
   Fileset                      Level  State      Description
     ----------------------------------------------------------------------------
   Path: /usr/lib/objrepos
     rsct.basic.hacmp           2.3.0.0  COMMITTED  RSCT Basic Function (HACMP/ES
                                                    Support)
     rsct.basic.rte             2.3.0.0  COMMITTED  RSCT Basic Function
     rsct.clients.rte     99.99.999.999  COMMITTED  Supersede Entry - Not really
                                                    installed
     rsct.compat.basic.hacmp    2.3.0.0  COMMITTED  RSCT Event Management Basic
                                                    Function (HACMP/ES Support)
     rsct.compat.basic.rte      2.3.0.0  COMMITTED  RSCT Event Management Basic
                                                    Function
     rsct.compat.clients.hacmp  2.3.0.0  COMMITTED  RSCT Event Management Client
                                                    Function (HACMP/ES Support)
     rsct.compat.clients.rte    2.3.0.0  COMMITTED  RSCT Event Management Client
                                                    Function
     rsct.core.auditrm          2.3.0.0  COMMITTED  RSCT Audit Log Resource
                                                    Manager
     rsct.core.errm             2.3.0.0  COMMITTED  RSCT Event Response Resource
                                                    Manager
     rsct.core.fsrm             2.3.0.0  COMMITTED  RSCT File System Resource
                                                    Manager
     rsct.core.hostrm           2.3.0.0  COMMITTED  RSCT Host Resource Manager
     rsct.core.rmc              2.3.0.0  COMMITTED  RSCT Resource Monitoring and
                                                    Control
     rsct.core.sec              2.3.0.0  COMMITTED  RSCT Security
     rsct.core.sr               2.3.0.0  COMMITTED  RSCT Registry
     rsct.core.utils            2.3.0.0  COMMITTED  RSCT Utilities

   Path: /etc/objrepos
     rsct.basic.rte             2.3.0.0  COMMITTED  RSCT Basic Function
     rsct.compat.basic.rte      2.3.0.0  COMMITTED  RSCT Event Management Basic
                                                    Function
     rsct.core.rmc              2.3.0.0  COMMITTED  RSCT Resource Monitoring and
                                                    Control
     rsct.core.sec              2.3.0.0  COMMITTED  RSCT Security
     rsct.core.sr               2.3.0.0  COMMITTED  RSCT Registry
     rsct.core.utils            2.3.0.0  COMMITTED  RSCT Utilities
   ```

Error results are indicated by no output from the command.

2. Issue the command:

```
lppchk -c "rsct*"
```

Good results are indicated by the absence of error messages and the return of a zero exit status from this command. The command produces no output if it succeeds.

Error results are indicated by a non-zero exit code and by error messages similar to these:

```
ppchk: 0504-206 File /usr/lib/nls/msg/en_US/hats.cat could not be located.
lppchk: 0504-206 File /usr/sbin/rsct/bin/hatsoptions could not be located.
lppchk: 0504-208 Size of /usr/sbin/rsct/bin/phoenix.snap is 29356,
                     expected value was 29355.
```

Some error messages may appear if an EFIX is applied to a file set. An EFIX is an emergency fix, supplied by IBM, to correct a specific problem.

If the test failed, verify the installation of RSCT. The following file sets need to be installed:

- **rsct.basic.hacmp**
- **rsct.basic.rte**
- **rsct.clients.rte**
- **rsct.compat.basic.hacmp**
- **rsct.compat.basic.rte**
- **rsct.compat.clients.hacmp**
- **rsct.compat.clients.rte**
- **rsct.core.auditrm**
- **rsct.core.errm**
- **rsct.core.fsrm**
- **rsct.core.hostrm**
- **rsct.core.rmc**
- **rsct.core.sec**
- **rsct.core.sr**
- **rsct.core.utils**
- **rsct.basic.rte**
- **rsct.compat.basic.rte**
- **rsct.core.rmc**
- **rsct.core.sec**
- **rsct.core.sr**
- **rsct.core.utils**

If the test succeeds, proceed to “Configuration verification test”. If the test fails, see if RSCT was installed, and install RSCT if it was not.

Configuration verification test

This test verifies that Topology Services has the configuration data it needs to build the machines.lst file.

The configuration data is propagated by the configuration resource manager and can be retrieved with the commands:

- **/usr/sbin/rsct/bin/ct_clusterinfo**
- **/usr/sbin/rsct/bin/ct_hats_info**
- **/usr/sbin/rsct/bin/ct_topology_info**

The output of **ct_clusterinfo** is similar to the following:

```
CLUSTER_NAME  gpfs
CLUSTER_ID    b181ecec-7055-4374-a998-ccd3f71db16a
NODE_NUMBER   2
```

The node number information is probably the most important.

The output of **ct_hats_info** is similar to the following:

```
REALM CLUSTER
LOGFILELEN 5000
FIXED_PRI -1
PORT 12347
PIN NONE
```

This command displays overall options for Topology Services. Any ″-1″ or ″DEFAULT″ values will prompt the Topology Services scripts to use appropriate default values.

- REALM: execution environment. Should be always ″CLUSTER″.
- LOGFILELEN: maximum number of lines in the Topology Services daemon log file.
- FIXED_PRI: fixed priority value.
- PORT: UDP port number for peer-to-peer communication.
- PIN: whether to pin the Topology Services daemon in memory.

The output of **ct_topology_info** is similar to the following:

```
NETWORK_NAME gpfs
NETWORK_SENS -1
NETWORK_NIM_PAR
NETWORK_BCAST 0
NETWORK_NIM_EXEC
NETWORK_SRC_ROUTING 0
NETWORK_FREQ -1
NETWORK_TYPE myrinet
ADAPTER 192.168.1.43 myri0 1 gpfs
ADAPTER 192.168.1.44 myri0 2 gpfs
```

The output has a section for each of the configured networks. For each network, tunable information is given, along with a list of all the adapters in the network. For each adapter, its IP address, interface name, node number, and network to which it belongs are given. Note that the node number for each node is given by the output of the **ct_clusterinfo** command.

The tunable values for each network are:

- NETWORK_FREQ: ″frequency″ value: how often to send heartbeat messages in seconds.
- NETWORK_SENS: ″sensitivity″ value: how many missed heartbeats before declaring the adapter dead.
- NETWORK_NIM_EXEC: Path name for NIM executable file.
- NETWORK_NIM_PAR: command-line argument to NIM.
- NETWORK_BCAST: 1 if network supports broadcast; 0 otherwise.

- NETWORK_SRC_ROUTING: 1 if network supports IP loose source routing, 0 otherwise.

Good results are indicated by the configuration, in terms of tunable values and network configuration, matching the user expectation for the cluster topology.

Error results are indicated if there is any inconsistency between the displayed configuration data and the desired configuration data. Issue the **cthatstune** command with the desired values.

Operational verification tests

The following names apply to the operational verification tests in this section. In a configuration resource manager environment (RSCT peer domain):

- Subsystem name: **cthats**
- User log file: **/var/ct/***cluster_name***/log/cthats/cthats.***DD.hhmmss.lang*
- Service log file: **/var/ct/***cluster_name***/log/cthats/cthats.***DD.hhmmss*
- **run** directory: **/var/ct/***cluster_name***/run/cthats**
- **machines.lst** file: **/var/ct/***cluster_name***/run/cthats/machines.lst**

In an HACMP environment:

- Subsystem name: **topsvcs**
- User log file: **/var/ha/log/topsvcs**.*DD.hhmmss.cluster_name.lang*
- Service log file: **/var/ha/log/topsvcs**.*DD.hhmmss.cluster_name*
- **run** directory: **/var/ha/run/topsvcs**.*cluster_name/*
- **machines.lst** file: **/var/ha/run/topsvcs**.*cluster_name/***machines**.*cluster_id*.**lst**

Operational test 1 - Verify status and adapters: This test verifies whether Topology Services is working and that all the adapters are up. Issue the **lssrc** command:

```
lssrc -ls subsystem_name
```

Good results are indicated by an output similar to the following:

```
Subsystem         Group            PID     Status
 cthats           cthats           20494    active
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
ethernet1      [ 0]   15   15  S 9.114.61.195    9.114.61.195
ethernet1      [ 0] eth0          0x3740dd5c      0x3740dd62
HB Interval = 1 secs. Sensitivity = 4 missed beats
SPswitch       [ 1]   14   14  S 9.114.61.139    9.114.61.139
SPswitch       [ 1] css0         0x3740dd5d      0x3740dd62
HB Interval = 1 secs. Sensitivity = 4 missed beats
  Configuration Instance = 926566126
  Default: HB Interval = 1 secs. Sensitivity = 4 missed beats
  Daemon employs no security
  Data segment size: 6358 KB. Number of outstanding malloc: 588
  Number of nodes up: 15. Number of nodes down: 0.
```

If the number under the `Mbrs` heading is the same as the number under `Defd`, all adapters defined in the configuration are part of the adapter membership group. The numbers under the `Group ID` heading should remain the same over subsequent invocations of **lssrc** several seconds apart. This is the expected behavior of the subsystem.

Error results are indicated by outputs similar to the following:

1. `0513-036 The request could not be passed to the cthats subsystem. Start the subsystem and try your command again.`

   In this case, the subsystem is down. Issue the **errpt -a** command and look for an entry for the subsystem name. Proceed to "Operational test 2 - Determine why the Topology Services subsystem is inactive" on page 236.

2. `0513-085 The cthats Subsystem is not on file.`

   The subsystem is not defined to the SRC.

3. This output requires investigation because the number under `Mbrs` is smaller than the number under `Defd`.

```
Subsystem         Group            PID     Status
cthats            cthats           20494    active
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
ethernet1      [ 0]   15    8  S 9.114.61.195    9.114.61.195
ethernet1      [ 0] eth0          0x3740dd5c      0x3740dd62
HB Interval = 1 secs. Sensitivity = 4 missed beats
SPswitch       [ 1]   14    7  S 9.114.61.139    9.114.61.139
SPswitch       [ 1] css0         0x3740dd5d      0x3740dd62
HB Interval = 1 secs. Sensitivity = 4 missed beats
  Configuration Instance = 926566126
  Default: HB Interval = 1 secs. Sensitivity = 4 missed beats
  Daemon employs no security
  Data segment size: 6358 KB. Number of outstanding malloc: 588
  Number of nodes up: 8. Number of nodes down: 7.
  Nodes down: 17-29(2)
```

   Some remote adapters are not part of the local adapter's group. Proceed to "Operational test 3 - Determine why remote adapters are not in the local adapter's membership group" on page 236.

4. This output requires investigation because a local adapter is disabled.

```
Subsystem         Group            PID     Status
 cthats           cthats           20494    active
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
ethernet1      [ 0]   15   15  S 9.114.61.195    9.114.61.195
ethernet1      [ 0] eth0          0x3740dd5c      0x3740dd62
HB Interval = 1 secs. Sensitivity = 4 missed beats
SPswitch       [ 1]   14    0  D 9.114.61.139
SPswitch       [ 1] css0
HB Interval = 1 secs. Sensitivity = 4 missed beats
  Configuration Instance = 926566126
  Default: HB Interval = 1 secs. Sensitivity = 4 missed beats
  Daemon employs no security
  Data segment size: 6358 KB. Number of outstanding malloc: 588
  Number of nodes up: 15. Number of nodes down: 0.
```

   A local adapter is disabled. Proceed to "Operational test 4 - Check address of local adapter" on page 237.

5. This output requires investigation because there is a **U** below the `St` heading.

```
Subsystem         Group            PID     Status
 ctchats         cthats           20494    active
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
ethernet1      [ 0]   15    8  S 9.114.61.195    9.114.61.195
ethernet1      [ 0] eth0          0x3740dd5c      0x3740dd62
HB Interval = 1 secs. Sensitivity = 4 missed beats
SPswitch       [ 1]   14    1  U 9.114.61.139    9.114.61.139
SPswitch       [ 1] css0         0x3740dd5d      0x3740dd5d
HB Interval = 1 secs. Sensitivity = 4 missed beats
  Configuration Instance = 926566126
  Default: HB Interval = 1 secs. Sensitivity = 4 missed beats
```

```
Daemon employs no security
Data segment size: 6358 KB. Number of outstanding malloc: 588
Number of nodes up: 8. Number of nodes down: 7.
Nodes down: 17-29(2)
```

The last line of the output shows a list of nodes that are either up or down, whichever is smaller. The list of nodes that are down includes only the nodes that are configured and have at least one adapter that Topology Services monitors. Nodes are specified by a list of node ranges, as follows:

N1-*N2*(*I1*) *N3*-*N4*(*I2*) ...

Here, there are two ranges, *N1*-*N2*(*I1*) and *N3*-*N4*(*I2*). They are interpreted as follows:

- *N1* is the first node in the first range
- *N2* is the last node in the first range
- *I1* is the increment for the first range
- *N3* is the first node in the second range
- *N4* is the last node in the second range
- *I2* is the increment for the second range

If the increment is 1, it is omitted. If the range has only one node, only that node's number is displayed. Examples are:

a. `Nodes down: 17-29(2)` means that nodes 17 through 29 are down. In other words, nodes 17, 19, 21, 23, 25, 27, and 29 are down.

b. `Nodes up: 5-9(2) 13` means that nodes 5, 7, 9, and 13 are up.

c. `Nodes up: 5-9 13-21(4)` means that nodes 5, 6, 7, 8, 9, 13, 17, and 21 are up.

An adapter stays in a singleton unstable membership group. This normally occurs for a few seconds after the daemon starts or after the adapter is re-enabled. If the situation persists for more than one minute, this may indicate a problem. This usually indicates that the local adapter is receiving some messages, but it is unable to obtain responses for its outgoing messages. Proceed to "Operational test 7 - Check for partial connectivity" on page 239.

6. An output similar to the expected output, or similar to output 3 on page 234, but where the numbers under the `Group ID` heading (either the address of the Group Leader adapter or the "incarnation number" of the group) change every few seconds without ever becoming stable.

   This kind of output indicates that there is some partial connectivity on the network. Some adapters may be able to communicate only with a subset of adapters. Some adapters may be able to send messages only or receive messages only. This output indicates that the adapter membership groups are constantly reforming, causing a substantial increase in the CPU and network resources used by the subsystem.

   A partial connectivity situation is preventing the adapter membership group from holding together. Proceed to "Operational test 10 - Check neighboring adapter connectivity" on page 242.

If this test is successful, proceed to "Operational test 11 - Verify node reachability information" on page 243.

Operational test 2 - Determine why the Topology Services subsystem is inactive: This test is to determine why the Topology Services subsystem is not active.

For HACMP/ES, issue the command: **errpt -N topsvcs -a**

For an RSCT peer domain, issue the command: **errpt -N cthats -a**

The AIX error log entries produced by this command, together with their description in Table 18 on page 202, explain why the subsystem is inactive. If no entry that explains why the subsystem went down or could not start exists, it is possible that the daemon may have exited abnormally.

In this case, issue the **errpt -a** command and look for an error. Look for an error entry with a LABEL: of CORE_DUMP and PROGRAM NAME of **hatsd**. (Issue the command: **errpt -J CORE_DUMP -a.**) If such an entry is found, see “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.

Another possibility when there is no **TS**_ error log entry, is that the Topology Services daemon could not be loaded. In this case a message similar to the following may be present in the Topology Services User startup log:

```
0509-036 Cannot load program hatsd because of the following errors:
0509-023 Symbol dms_debug_tag in hatsd is not defined.
0509-026 System error: Cannot run a file that does not have a valid format.
```

The message may refer to the Topology Services daemon, or to some other program invoked by the startup script. If such an error is found, contact the IBM Support Center.

For errors where the daemon did start up but exited during initialization, detailed information about the problem is in the Topology Services User error log.

Operational test 3 - Determine why remote adapters are not in the local adapter's membership group: Issue the **lssrc** command:

```
lssrc -ls subsystem
```

on all the nodes.

Issue the **lssrc** command on all the nodes.

If this test follows output 3 on page 234, at least one node will not have the same output as the node from where output 3 on page 234 was taken.

Some of the possibilities are:

1. The node is down or unreachable. Diagnose that node by using “Operational test 1 - Verify status and adapters” on page 233.
2. The output is similar to output of 3 on page 234, but with a different group id, such as in this output:

```
Subsystem         Group            PID     Status
 cthats           cthats           20494   active
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
ethernet1      [ 0]   15    7  S 9.114.61.199    9.114.61.201
ethernet1      [ 0] eth0          0x3740dd5c      0x3740dd72
HB Interval = 1 secs. Sensitivity = 4 missed beats
```

```
SPswitch       [ 1]   14    7  S 9.114.61.141    9.114.61.141
SPswitch       [ 1] css0         0x3740dd5d      0x3740dd72
HB Interval = 1 secs. Sensitivity = 4 missed beats
  Configuration Instance = 926566126
  Default: HB Interval = 1 secs. Sensitivity = 4 missed beats
  Daemon employs no security
  Data segment size: 6358 KB. Number of outstanding malloc: 588
  Number of nodes up: 7. Number of nodes down: 8.
  Nodes up: 17-29(2)
```

Compare this with the output from 3 on page 234. Proceed to "Operational test 8 - Check if configuration instance and security status are the same across all nodes" on page 240.

3. The output is similar to the outputs of 1 on page 234, 2 on page 234, 4 on page 234, or 5 on page 234. Return to "Operational test 1 - Verify status and adapters" on page 233, but this time focus on this new node.

Operational test 4 - Check address of local adapter: This test verifies whether a local adapter is configured with the correct address. Assuming that this test is being run because the output of the **lssrc** command indicates that the adapter is disabled, there should be an entry in the error log that points to the problem.

Issue the command:

```
errpt -J TS_LOC_DOWN_ST,TS_MISCFG_EM -a | more
```

Examples of the error log entries that appear in the output are:

•
```
LABEL:          TS_LOC_DOWN_ST
IDENTIFIER:     D17E7B06

Date/Time:       Mon May 17 23:29:34
Sequence Number: 227
Machine Id:      000032054C00
Node Id:         c47n11
Class:           S
Type:            INFO
Resource Name:   cthats.c47s

Description
Possible malfunction on local adapter
```
•
```
LABEL:          TS_MISCFG_EM
IDENTIFIER:     6EA7FC9E

Date/Time:       Mon May 17 16:28:45
Sequence Number: 222
Machine Id:      000032054C00
Node Id:         c47n11
Class:           U
Type:            PEND
Resource Name:   cthats.c47s
Resource Class:  NONE
Resource Type:   NONE
Location:        NONE
VPD:

Description
Local adapter misconfiguration detected
```

Good results are indicated by the absence of the **TS_MISCFG_EM** error entry. To verify that the local adapter has the expected address, issue the command:

```
ifconfig interface_name
```

where *interface_name* is the interface name listed on the output of **lssrc**, such as:

```
SPswitch       [ 1]   14    0  D 9.114.61.139
SPswitch       [ 1] css0
```

For the **lssrc** command output, the output of **ifconfig css0** is similar to:

```
css0: flags=800847 <UP,BROADCAST,DEBUG,RUNNING,SIMPLEX>
        inet 9.114.61.139 netmask 0xffffffc0 broadcast 9.114.61.191
```

Error results are indicated by the **TS_MISCFG_EM** error entry and by the output of the **ifconfig** command not containing the address displayed in the **lssrc** command output. Diagnose the reason why the adapter is configured with an incorrect address

If this test is a success, proceed to “Operational test 5 - Check if the adapter is enabled for IP”.

Operational test 5 - Check if the adapter is enabled for IP: Issue the command:

```
ifconfig interface_name
```

The output is similar to the following:

```
css0: flags=800847 <UP,BROADCAST,DEBUG,RUNNING,SIMPLEX>
          inet 9.114.61.139 netmask 0xffffffc0 broadcast 9.114.61.191
```

Good results are indicated by the presence of the `UP` string in the third line of the output. In this case, proceed to “Operational test 6 - Check whether the adapter can communicate with other adapters in the network”.

Error results are indicated by the absence of the `UP` string in the third line of the output.

Issue the command:

```
ifconfig interface_name up
```

to re-enable the adapter to IP.

Operational test 6 - Check whether the adapter can communicate with other adapters in the network: **Root** authority is needed to access the contents of the **machines.lst** file. Display the contents of the **machines.lst** file. The output is similar to the following:

```
*InstanceNumber=925928580
*configId=1244520230
*!HaTsSeCStatus=off
*FileVersion=1
*!TS_realm=CLUSTER
TS_Frequency=1
TS_Sensitivity=4
TS_FixedPriority=38
TS_LogLength=5000
*!TS_PinText
Network Name ethernet1
Network Type ether
```

```
*
*Node Type Address
    0 en0 9.114.61.125
    1 en0  9.114.61.65
    3 en0  9.114.61.67
    11 en0  9.114.61.195
...
Network Name SPswitch
Network Type hps
*
*Node Type Address
    1 css0 9.114.61.129
    3 css0 9.114.61.131
    11 css0 9.114.61.139
```

Locate the network to which the adapter under investigation belongs. For example, the css0 adapter on node 11 belongs to network `SPswitch`. Issue the command:

```
ping -c 5 address
```

for the addresses listed in the **machines.lst** file.

Good results are indicated by outputs similar to the following.

```
PING 9.114.61.129: (9.114.61.129): 56 data bytes
64 bytes from 9.114.61.129: icmp_seq=0 ttl=255 time=0 ms
64 bytes from 9.114.61.129: icmp_seq=1 ttl=255 time=0 ms
64 bytes from 9.114.61.129: icmp_seq=2 ttl=255 time=0 ms
64 bytes from 9.114.61.129: icmp_seq=3 ttl=255 time=0 ms
64 bytes from 9.114.61.129: icmp_seq=4 ttl=255 time=0 ms

----9.114.61.129 PING Statistics----
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0/0/0 ms
```

The number before `packets received` should be greater than 0.

Error results are indicated by outputs similar to the following:

```
PING 9.114.61.129: (9.114.61.129): 56 data bytes

----9.114.61.129 PING Statistics----
5 packets transmitted, 0 packets received, 100% packet loss
```

The command should be repeated with different addresses until it succeeds or until several different attempts are made. After that, pursue the problem as an adapter or IP-related problem.

If this test succeeds, but the adapter is still listed as disabled in the **lssrc** command output, collect the data listed in “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.

Operational test 7 - Check for partial connectivity: Adapters stay in a singleton unstable state when there is partial connectivity between two adapters. One reason for an adapter to stay in this state is that it keeps receiving PROCLAIM messages, to which it responds with a JOIN message, but no PTC message comes in response to the JOIN message.

Check in the Topology Services User log file to see if a message similar to the following appears repeatedly:

```
2523-097 JOIN time has expired. PROCLAIM message was sent
                   by (10.50.190.98:0x473c6669)
```

If this message appears repeatedly in the Topology Services User log, investigate IP connectivity between the local adapter and the adapter whose address is listed in the User log entry (10.50.190.98 in the example here). Issue command:

```
ping -c 5 address
```

address is 10.50.190.98 in this example.

See “Operational test 5 - Check if the adapter is enabled for IP” on page 238 for a description of **good results** for this command.

The local adapter cannot communicate with a Group Leader that is attempting to attract the local adapter into the adapter membership group. The problem may be with either the local adapter or the Group Leader adapter (″proclaimer″ adapter). Pursue this as an IP connectivity problem. Focus on both the local adapter and the Group Leader adapter.

If the **ping** command succeeds, but the local adapter still stays in the singleton unstable state, contact the IBM Support Center.

In an HACMP/ES environment, it is possible that there are two adapters in different nodes both having the same service address. This can be verified by issuing:

```
lssrc -ls subsystem_name
```

and looking for two different nodes that have the same IP address portion of Adapter ID. In this case, this problem should be pursued as an HACMP/ES problem. Contact the IBM Support Center.

If this test fails, proceed to “Operational test 4 - Check address of local adapter” on page 237, concentrating on the local and Group Leader adapters.

Operational test 8 - Check if configuration instance and security status are the same across all nodes: This test is used when there seem to be multiple partitioned adapter membership groups across the nodes, as in output 2 on page 236.

This test verifies whether all nodes are using the same configuration instance number and same security setting. The instance number changes each time the **machines.lst** file is generated by the startup script. In an RSCT peer domain, the configuration instance always increases.

Issue the **lssrc** command:

```
lssrc -ls subsystem_name
```

on all nodes. If this is not feasible, issue the command at least on nodes that produce an output that shows a different `Group ID`.

Compare the line `Configuration Instance = (number)` in the **lssrc** outputs. Also, compare the line `Daemon employs` in the **lssrc** command outputs.

Good results are indicated by the number after the `Configuration Instance` phrase being the same in all the **lssrc** outputs. This means that all nodes are working with the same version of the **machines.lst** file.

Error results are indicated by the configuration instance being different in the two ″node partitions″. In this case, the adapters in the two partitions cannot merge into a single group because the configuration instances are different across the node partitions. This situation is likely to be caused by a refresh-related problem. One of the node groups, probably that with the lower configuration instance, was unable to run a refresh. If a refresh operation was indeed attempted, consult the description of the ″Nodes or adapters leave membership after refresh″ problem in “Error symptoms, responses, and recoveries” on page 244.

The situation may be caused by a problem in the SRC subsystem, which fails to notify the Topology Services daemon about the refresh. The description of the ″Nodes or adapters leave membership after refresh″ problem in “Error symptoms, responses, and recoveries” on page 244 explains how to detect the situation where the Topology Services daemon has lost its connection with the SRC subsystem. In this case, contact the IBM Support Center.

If this test is successful, proceed to “Operational test 9 - Check connectivity among multiple node partitions”.

Operational test 9 - Check connectivity among multiple node partitions: This test is used when adapters in the same Topology Services network form multiple adapter membership groups, rather than a single group encompassing all the adapters in the network.

Follow the instructions in “Operational test 8 - Check if configuration instance and security status are the same across all nodes” on page 240 to obtain **lssrc** outputs for each of the node partitions.

The IP address listed in the **lssrc** command output under the `Group ID` heading is the IP address of the Group Leader. If two node partitions are unable to merge in to one, this is caused by the two Group Leaders being unable to communicate with each other. Note that even if some adapters in different partitions can communicate, the group merge will not occur unless the Group Leaders are able to exchange point-to-point messages. Use **ping** (as described in “Operational test 6 - Check whether the adapter can communicate with other adapters in the network” on page 238) to determine whether the Group Leaders can communicate with each other.

For example, assume on one node the output of the **lssrc -ls cthats** command is:

```
Subsystem         Group            PID     Status
cthats        cthats             15750     active
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
ethernet1      [0]   15    9   S 9.114.61.65     9.114.61.195
ethernet1      [0]              0x373897d2      0x3745968b
HB Interval = 1 secs. Sensitivity = 4 missed beats
SPswitch       [1]   14    14  S 9.114.61.129    9.114.61.153
SPswitch       [1]              0x37430634      0x374305f1
HB Interval = 1 secs. Sensitivity = 4 missed beats
```

and on another node it is:

```
Subsystem         Group            PID     Status
 cthats          cthats           13694    active
```

```
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
ethernet1      [0]  15    6  S 9.114.30.69     9.114.61.71
ethernet1      [0]             0x37441f24      0x37459754
HB Interval = 1 secs. Sensitivity = 4 missed beats
SPswitch       [1]   14   14  S 9.114.61.149    9.114.61.153
SPswitch       [1]             0x374306a4      0x374305f1
```

In this example, the partition is occurring on network ethernet1. The two Group Leaders are IP addresses 9.114.61.195 and 9.114.61.71. Login to the node that hosts one of the IP addresses and issue the **ping** test to the other address. In case the two adapters in question are in the same subnet, verify whether they have the same subnet mask.

Good results and **error results** for the **ping** test are described in “Operational test 6 - Check whether the adapter can communicate with other adapters in the network” on page 238. If the **ping** test is not successful, a network connectivity problem between the two Group Leader nodes is preventing the groups from merging. Diagnose the network connectivity problem.

Good results for the subnet mask test are indicated by the adapters that have the same subnet id also having the same subnet mask. If the subnet mask test fails, the subnet mask at one or more nodes must be corrected by issuing the command:

```
ifconfig interface_name address netmask netmask
```

All the adapters that belong to the same subnet must have the same subnet mask.

If the **ping** test is successful (the number of `packets received` is greater than 0), and the subnet masks match, there is some factor other than network connectivity preventing the two Group Leaders from contacting each other. The cause of the problem may be identified by entries in the Topology Services User log. If the problem persists, collect the data listed in “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center. Include information about the two Group Leader nodes.

Operational test 10 - Check neighboring adapter connectivity: This test checks neighboring adapter connectivity, in order to investigate partial connectivity situations. Issue the command **errpt -J TS_DEATH_TR | more** on all the nodes. Look for recent entries with label **TS_DEATH_TR**. This is the entry created by the subsystem when the local adapter stops receiving heartbeat messages from the neighboring adapter. For the adapter membership groups to be constantly reforming, such entries should be found in the error log.

Issue the **ping** test on the node where the **TS_DEATH_TR** entry exists. The target of the **ping** should be the adapter whose address is listed in the Detail Data of the error log entry. “Operational test 6 - Check whether the adapter can communicate with other adapters in the network” on page 238 describes how to perform the **ping** test and interpret the results.

If the **ping** test fails, this means that the two neighboring adapters have connectivity problems, and the problem should be pursued as an IP connectivity problem.

If the **ping** test is successful, the problem is probably not due to lack of connectivity between the two neighboring adapters. The problem may be due to one of the two adapters not receiving the COMMIT message from the ″mayor adapter″ when the group is formed. The **ping** test should be used to probe the connectivity between the two adapters and all other adapters in the local subnet.

Operational test 11 - Verify node reachability information: Issue the **lssrc** command:

```
lssrc -ls subsystem_name
```

and examine lines:

1. `Number of nodes up: # . Number of nodes down: #.`
2. `Nodes down: [...]` or `Nodes up: [...]`

in the command output.

Good results are indicated by the line `Number of Nodes down: 0`. For example,

```
Number of nodes up: 15    Number of nodes down: 0
```

However, such output can only be considered correct if indeed all nodes in the system are known to be up. If a given node is indicated as being up, but the node seems unresponsive, perform problem determination on the node. Proceed to "Operational test 12 - Verify the status of an unresponsive node that is shown to be up by Topology Services".

Error results are indicated by `Number of Nodes down:` being nonzero. The list of nodes that are flagged as being up or down is given in the next output line. An output such as `Nodes down: 17-23(2)` indicates that nodes 17, 19, 21, and 23 are considered down by Topology Services. If the nodes in the list are known to be down, this is the expected output. If, however, some of the nodes are thought to be up, it is possible that a problem exists with the Topology Services subsystem on these nodes. Proceed to "Operational test 1 - Verify status and adapters" on page 233, focusing on each of these nodes.

Operational test 12 - Verify the status of an unresponsive node that is shown to be up by Topology Services: Examine the **machines.lst** configuration file and obtain the IP addresses for all the adapters in the given node that are in the Topology Services configuration. For example, for node 9, entries similar to the following may be found in the file:

```
 9 eth0  9.114.61.193
 9 css0 9.114.61.137
```

Issue this command.

```
ping -c5 IP_address
```

If there is no response to the **ping** packets (the output of the command shows `100% packet loss`) for all the node's adapters, the node is either down or unreachable. Pursue this as a node health problem. If Topology Services still indicates the node as being up, contact the IBM Support Center because this is probably a Topology Services problem. Collect long tracing information from the Topology Services logs. See "Topology Services service log" on page 225. Run the **tcpdump** command as described in "Information to collect before contacting the IBM Support Center" on page 229.

If the output of the **ping** command shows some response (for example, `0% packet loss`), the node is still up and able to send and receive IP packets. The Topology

Services daemon is likely to be running and able to send and receive heartbeat packets. This is why the node is still seen as being up. This problem should be pursued as a Linux-related problem.

If there is a response from the **ping** command, and the node is considered up by remote Topology Services daemons, but the node is unresponsive and no user application is apparently able to run, a system dump must be obtained to find the cause of the problem.

Error symptoms, responses, and recoveries

Use the following table to diagnose problems with the Topology Services component of RSCT. Locate the symptom and perform the action described in the following table.

Table 19. Topology Services symptoms

Symptom	Recovery
Adapter membership groups do not include all the nodes in the configuration.	See “Operational test 1 - Verify status and adapters” on page 233.
Topology Services subsystem fails to start.	See “Action 1 - Investigate startup failure”.
The refresh operation fails or has no effect.	See “Action 2 - Investigate refresh failure” on page 245.
A local adapter is notified as being down by Topology Services.	See “Action 3 - Investigate local adapter problems” on page 245.
Adapters appear to be going up and down continuously.	See “Action 4 - Investigate partial connectivity problem” on page 246.
A node appears to go down and then up a few seconds later.	See “Action 5 - Investigate hatsd problem” on page 247.
Adapter appears to go down and then up a few seconds later.	See “Action 6 - Investigate IP communication problem” on page 253.
Group Services exits abnormally because of a Topology Services Library error. Error log entry with template **GS_TS_RETCODE_ER** is present.	See “Action 7 - Investigate Group Services failure” on page 253.
Nodes or adapters leave membership after a refresh.	See “Action 8 - Investigate problems after a refresh” on page 254.
A node has crashed.	See “Action 9 - Investigate a node crash” on page 255.

Actions

Action 1 - Investigate startup failure: Some of the possible causes are:

- Adapter configuration problems, such as duplicated IP addresses in the configuration.
- Operating system-related problems, such as a shortage of space in the **/var** directory or a port number already in use.
- Security services problems that prevent Topology Services from obtaining credentials, determining the active authentication method, or determining the authentication keys to use.

See “Operational test 2 - Determine why the Topology Services subsystem is inactive” on page 236. To verify the correction, see “Operational test 1 - Verify status and adapters” on page 233.

Action 2 - Investigate refresh failure: The most probable cause is that an incorrect adapter or network configuration was passed to Topology Services. Refresh errors are listed in the **/var/ct/***cluster_name***log/cthats/refreshOutput** file, and the startup script log. See “cthats script log” on page 228 for more information on the startup script log.

Also, configuration errors result in AIX error log entries being created. Some of the template labels that may appear are:

- TS_CTNODEUP_ER
- TS_CTIPDUP_ER
- TS_CL_FATAL_GEN_ER
- TS_HANODEDUP_ER
- TS_HAIPDUP_ER

The syslog entries should provide enough information to determine the cause of the problem. Detailed information about the configuration and the error or can be found in the startup script log and the Topology Services user log.

For the problems that result in the syslog entries listed here, the solution involves changing the IP address of one or more adapters.

A Topology Services refresh will occur whenever changes are made to the topology, such as when a communication group is modified by the **chcomg** command (as described in “Modifying a Communication Group’s Characteristics” on page 21).

Incorrect or conflicting adapter information will result in the refresh having no effect, and in error log entries to be created in syslog.

Action 3 - Investigate local adapter problems: The most common local adapter problems are:

1. The adapter is not working.
2. The network may be down.
3. The adapter may have been configured with an incorrect IP address.
4. Topology Services is unable to get response packets back to the adapter.
5. There is a problem in the subsystem’s ″adapter self-death″ procedures.

See “Operational test 4 - Check address of local adapter” on page 237 to analyze the problem. The repair action depends on the nature of the problem. For problems 1 through 3, the underlying cause for the adapter to be unable to communicate must be found and corrected.

For problem 4, Topology Services requires that at least one other adapter in the network exist, so that packets can be exchanged between the local and remote adapters. Without such an adapter, a local adapter would be unable to receive any packets. Therefore, there would be no way to confirm that the local adapter is working.

To verify the repair, issue the **lssrc** command as described in “Operational test 1 - Verify status and adapters” on page 233. If the problem is due to Topology Services being unable to obtain response packets back to the adapter (problem 4), the problem can be circumvented by adding machine names to file **/usr/sbin/cluster/netmon.cf**. These machines should be routers or any machines that are external to the configuration, but are reachable from one of the networks being monitored by the subsystem. Any entry in this file is used as a target for a

probing packet when Topology Services is attempting to determine the health of a local adapter. The format of the file is as follows:

```
machine name or IP address 1
machine name or IP address 2
..........
```

where the IP addresses are in dotted decimal format. If the file does not exist, it should be created. To remove this recovery action, remove the entries added to the file, delete the file, or rename the file.

Action 4 - Investigate partial connectivity problem: The most probable cause is a partial connectivity scenario. This means that one adapter or a group of adapters can communicate with some, but not all, remote adapters. Stable groups in Topology Services require that all adapters in a group be able to communicate with each other.

Some possible sources of partial connectivity are:

1. Physical connectivity
2. Incorrect routing at one or more nodes
3. Adapter or network problems which result in packets larger than a certain size being lost
4. Incorrect ARP setting in large machine configurations.

   The total number of entries in the ARP table must be a minimum of two times the number of nodes. The number of entries in the ARP table is calculated by multiplying the **arptab_bsiz** parameter by the **arptab_nb** parameter. The parameters **arptab_bsiz** and **arptab_nb** are tunable parameters controlled by the AIX **no** (**n**etwork **o**ptions) command.
5. High network traffic, which causes a significant portion of the packets to be lost.

To check whether there is partial connectivity on the network, run “Operational test 10 - Check neighboring adapter connectivity” on page 242. The underlying connectivity problem must be isolated and corrected. To verify the correction, issue the **lssrc** command from “Operational test 1 - Verify status and adapters” on page 233.

The problem can be bypassed if the connectivity test revealed that one or more nodes have only partial connectivity to the others. In this case, Topology Services can be stopped on these partial connectivity nodes. If the remaining adapters in the network have complete connectivity to each other, they should form a stable group.

Topology Services subsystem can be stopped on a node by issuing the **cthatsctrl** command:

```
/usr/sbin/rsct/bin/cthatsctrl -k
```

Note that the nodes where the subsystem was stopped will be marked as down by the others. Applications such as IBM Virtual Shared Disk and GPFS will be unable to use these nodes.

To test and verify this recovery, issue the **lssrc** command as described in “Operational test 1 - Verify status and adapters” on page 233. The `Group ID` information in the output should not change across two invocations approximately one minute apart.

Once this recovery action is no longer needed, restart Topology Services by issuing the **cthatsctrl** command:

```
/usr/sbin/rsct/bin/cthatsctrl -s
```

Action 5 - Investigate hatsd problem: Probable causes of this problem are:

1. The Topology Services daemon is temporarily blocked.
2. The Topology Services daemon exited on the node.
3. IP communication problem, such as mbuf shortage or excessive adapter traffic.

Probable cause 1 can be determined by the presence of an error log entry with **TS_LATEHB_PE** template on the affected node. This entry indicates that the daemon was blocked and for how long. When the daemon is blocked, it cannot send messages to other adapters, and as a result other adapters may consider the adapter dead in each adapter group. This results in the node being considered dead.

The following are some of the reasons for the daemon to be blocked:

1. A memory shortage, which causes excessive paging and thrashing behavior; the daemon stays blocked, awaiting a page-in operation.
2. A memory shortage combined with excessive disk I/O traffic, which results in slow paging operations.
3. The presence of a fixed-priority process with higher priority than the Topology Services daemon, which prevents the daemon from running.
4. Excessive interrupt traffic, which prevents any process in the system from being run in a timely manner.

A memory shortage is usually detected by the **vmstat** command. Issue the command:

```
vmstat -s
```

to display several memory-related statistics. Large numbers for paging space page ins or paging space page outs (a significant percentage of the page ins counter) indicate excessive paging.

Issue the command: **vmstat 5 7** to display some virtual memory counters over a 30-second period or time. If the number of free pages (number under the fre heading) is close to 0 (less than 100 or so), this indicates excessive paging. A nonzero value under po (pages paged out to paging space) occurring consistently also indicates heavy paging activity.

In a system which appears to have enough memory, but is doing very heavy I/O operations, it is possible that the virtual memory manager may “steal” pages from processes (“computational pages”) and assign them to file I/O (“permanent pages”). In this case, to allow more computational pages to be kept in memory, the **vmtune** command can be used to change the proportion of computational pages and permanent pages.

The same command can also be used to increase the number of free pages in the node, below which the virtual memory manager starts stealing pages and adding them to the free list. Increasing this number should prevent the number of free pages from reaching zero, which would force page allocation requests to wait. This number is controlled by the **minfree** parameter of the **vmtune** command.

Command:

```
/usr/samples/kernel/vmtune -f 256 -F 264 -p 1 -P 2
```

can be used to increase **minfree** to 256 and give more preference to computational pages. More information is in the **minfree** parameter description of the Appendix "Summary of Tunable AIX Parameters", in *AIX Versions 3.2 and 4 Performance Tuning Guide*.

If the reason for the blockage cannot be readily identified, AIX tracing can be set up for when the problem recurs. The command:

```
/usr/bin/trace -a -l -L 16000000 -T 8000000 -o /tmp/trace_raw
```

should be run in all the nodes where the problem is occurring. Enough space for a 16MB file should be reserved on the file system where the trace file is stored (**/tmp** in this example).

The trace should be stopped with the command:

```
/usr/bin/trcstop
```

as soon as the **TS_LATEHB_PE** entry is seen in the AIX error log. The resulting trace file and the **/unix** file should be saved for use by the IBM Support Center.

The underlying problem that is causing the Topology Services daemon to be blocked must be understood and solved. Problems related to memory thrashing behavior are addressed by *AIX Versions 3.2 and 4 Performance Tuning Guide*. In most cases, obtaining the AIX trace for the period that includes the daemon blockage (as outlined previously) is essential to determine the source of the problem.

For problems related to memory thrashing, it has been observed that if the Topology Services daemon is unable to run in a timely manner, this indicates that the amount of paging is causing little useful activity to be accomplished on the node.

Memory contention problems in Topology Services can be reduced by using the AIX Workload Manager. See "Preventing memory contention problems with the AIX Workload Manager" on page 251.

For problems related to excessive disk I/O, these steps can be taken in AIX to reduce the I/O rate:

1. Set I/O pacing.

   I/O pacing limits the number of pending write operations to file systems, thus reducing the disk I/O rate. AIX is installed with I/O pacing disabled. I/O pacing can be enabled with the command:

   ```
   chdev -l sys0 -a maxpout='33' -a minpout='24'
   ```

   This command sets the high-water and low-water marks for pending write-behind I/Os per file. The values can be tuned if needed.
2. Change the frequency of **syncd**.

   If this daemon is run more frequently, fewer number of pending I/O operations will need to be flushed to disk. Therefore, the invocation of **syncd** will cause less of a peak in I/O operations.

   To change the frequency of **syncd**, edit (as **root**) the **/sbin/rc.boot** file. Search for the following two lines:

```
echo "Starting the sync daemon" | alog -t boot
nohup /usr/sbin/syncd 60 > /dev/null 2>&1 &
```

The period is set in seconds in the second line, immediately following the invocation of **/usr/sbin/syncd**. In this example, the interval is set to 60 seconds. A recommended value for the period is 10 seconds. A reboot is needed for the change to take effect.

In a system which appears to have enough memory, but is doing very heavy I/O operations, it is possible that the virtual memory manager may ″steal″ pages from processes (″computational pages″) and assign them to file I/O (″permanent pages″).

The underlying problem that is causing the Topology Services daemon to be blocked must be understood and resolved.

For problems related to memory thrashing, it has been observed that if the Topology Services daemon is unable to run in a timely manner, this indicates that the amount of paging is causing little useful activity to be accomplished on the node.

If the problem is related to a process running with a fixed priority which is higher (that is, a larger number) than that of the Topology Services daemon, the problem may be corrected by changing the daemon's priority. This can be done by issuing the **cthatstune** command:

```
/usr/sbin/rsct/bin/cthatstune -p new_value -r
```

Probable cause 2 on page 247 can be determined by the presence of an syslog entry that indicates that the daemon exited. See "Error Logs and templates" on page 200 for the list of possible error templates used. Look also for an error entry with a LABEL of CORE_DUMP and PROGRAM NAME of **hatsd**. This indicates that the daemon exited abnormally, and a **core** file should exist in the daemon's **run** directory.

If the daemon produced one of the error log entries before exiting, the error log entry itself, together with the information from "Error Logs and templates" on page 200, should provide enough information to diagnose the problem. If the CORE_DUMP entry was created, follow instructions in "Information to collect before contacting the IBM Support Center" on page 229 and contact the IBM Support Center.

Probable cause 3 on page 247 is the most difficult to analyze, since there may be multiple causes for packets to be lost. Some commands are useful in determining if packets are being lost or discarded at the node. Issue these commands:

1. `netstat -D`

   The `Idrops` and `Odrops` headings are the number of packets dropped in each interface or device.

2. `netstat -m`

   The `failed` heading is the number of mbuf allocation failures.

3. `netstat -s`

   The `socket buffer overflows` text is the number of packets discarded due to lack of socket space.

The `ipintrq overflows` text is the number of input packets discarded because of lack of space in the packet interrupt queue.

4. `netstat -v`

   This command shows several adapter statistics, including packets lost due to lack of space in the adapter transmit queue, and packets lost probably due to physical connectivity problems (“CRC Errors”).

5. `vmstat -i`

   This command shows the number of device interrupts for each device, and gives an idea of the incoming traffic.

There can be many causes for packets to be discarded or lost, and the problem needs to be pursued as an IP-related problem. Usually the problem is caused by one or more of the following:

1. Excessive IP traffic on the network or the node itself.
2. Inadequate IP or UDP tuning.
3. Physical problems in the adapter or network.

If causes 1 and 2 do not seem to be present, and cause 3 could not be determined, some of the commands listed previously should be issued in loop, so that enough IP-related information is kept in case the problem happens again.

The underlying problem that is causing packets to be lost must be understood and solved. The repair is considered effective if the node is no longer considered temporarily down under a similar workload.

In some environments (probable causes 1 on page 247 and 3 on page 247), the problem may be bypassed by relaxing the Topology Services tunable parameters, to allow a node not to be considered down when it cannot temporarily send network packets. Changing the tunable parameters, however, also means that it will take longer to detect a node or adapter as down.

Note: Before the tunable parameters are changed, record the current values, so that they can be restored to their original values if needed.

This solution can only be applied when:

1. There seems to be an upper bound on the amount of ″outage″ the daemon is experiencing.
2. The applications running on the system can withstand the longer adapter or node down detection time.

The **cthatstune** command:

```
cthatstune -f VIEW -s VIEW
```

can be used to display the current *Frequency* and *Sensitivity* values for all the networks being monitored.

The adapter and node detection time is given by the formula:

```
2 * Sensitivity * Frequency
```

(two multiplied by the value of *Sensitivity* multiplied by the value of *Frequency*)

These values can be changed with:

```
cthatstune [-f [network:]frequency] [-s [network:]sensitivity] -r
```

where

- The **-f** flag represents the *Frequency* tunable value.
- The **-s** flag represents the *Sensitivity* tunable value.

The tuning can be done on a network-basis if the **network** operand is specified If **network** it omitted, the changes apply to all the networks.

To verify that the tuning changes have taken effect, issue the **lssrc** command:

```
lssrc -ls subsystem_name
```

approximately one minute after making the changes. The tunable parameters in use are shown in the output in a line similar to the following:

```
HB Interval = 1 secs. Sensitivity = 4 missed beats
```

For each network, `HB Interval` is the *Frequency* parameter, and `Sensitivity` is the *Sensitivity* parameter.

For examples of tuning parameters that can be used in different environments, consult Chapter 5, “The Topology Services subsystem” on page 185 and the **cthatstune** command.

Good results are indicated by the tunable parameters being set to the desired values.

Error results are indicated by the parameters having their original values or incorrect values.

To verify whether the tuning changes were effective in masking the daemon outage, the system has to undergo a similar workload to that which caused the outage.

To remove the tuning changes, follow the same tuning changes outlined previously, but this time restore the previous values of the tunable parameters.

Preventing memory contention problems with the AIX Workload Manager: Memory contention has often caused the Topology Services daemon to be blocked for significant periods of time. This results in “false node downs”, and in the triggering of the Dead Man Switch timer in HACMP/ES. An AIX error log entry with label **TS_LATEHB_PE** may appear when running RSCT 1.2 or higher. The message “Late in sending Heartbeat by ...” will appear in the daemon log file in any release of RSCT, indicating that the Topology Services daemon was blocked. Another error log entry that could be created is **TS_DMS_WARNING_ST**.

In many cases, such as when the system is undergoing very heavy disk I/O, it is possible for the Topology Services daemon to be blocked in paging operations, even though it looks like the system has enough memory. Two possible causes for this phenomenon are:

- In steady state, when there are no node and adapter events on the system, the Topology Services daemon uses a “working set” of pages that is substantially smaller than its entire addressing space. When node or adapter events happen, the daemon faces the situation where additional pages it needs to process the events are not present in memory.

- When heavy file I/O is taking place, the operating system may reserve a larger percentage of memory pages to files, making fewer pages available to processes.
- When heavy file I/O is taking place, paging I/O operations may be slowed down by contention for the disk.

The probability that the Topology Services daemon gets blocked for paging I/O may be reduced by making use of the AIX Workload Manager (WLM). WLM is an operating system feature introduced in AIX Version 4.3.3. It is designed to give the system administrator greater control over how the scheduler and Virtual Memory Manager (VMM) allocate CPU and physical memory resources to processes. WLM gives the system administrator the ability to create different classes of service, and specify attributes for those classes.

The following explains how WLM can be used to allow the Topology Services daemon to obtain favorable treatment from the VMM. There is no need to involve WLM in controlling the daemon's CPU use, because the daemon is already configured to run at a real time fixed scheduling priority. WLM will not assign priority values smaller than 40 to any thread.

These instructions are given using SMIT, but it is also possible to use WLM or AIX commands to achieve the same goals.

Initially, use the sequence:

```
smit wlm
   Add a Class
```

to add a `TopologyServices` class to WLM. Ensure that the class is at `Tier 0` and has `Minimum Memory` of 20%. These values will cause processes in this class to receive favorable treatment from the VMM. Tier 0 means that the requirement from this class will be satisfied before the requirements from other classes with higher tiers. Minimum Memory should prevent the process's pages from being taken by other processes, while the process in this class is using less than 20% of the machine's memory.

Use the sequence:

```
      smit wlm
         Class Assignment Rules
            Create a new Rule
```

to create a rule for classifying the Topology Services daemon into the new class. In this screen, specify **1** as `Order of the Rule`, `TopologyServices` as `Class`, and **/usr/sbin/rsct/bin/hatsd** as `Application`.

To verify the rules that are defined, use the sequence:

```
smit wlm
   Class Assignment Rules
        List all Rules
```

To start WLM, after the new class and rule are already in place, use the sequence:

```
smit wlm
   Start/Stop/Update WLM
       Start Workload Management
```

To verify that the Topology Services daemon is indeed classified in the new class, use command:

```
ps -ef -o pid,class,args | grep hatsd | grep -v grep
```

One sample output of this command is:

```
15200 TopologyServices /usr/sbin/rsct/bin/hatsd -n 5
```

The `TopologyServices` text in this output indicates that the Topology Services daemon is a member of the `TopologyServices` class.

If WLM is already being used, the system administrator must ensure that the new class created for the Topology Services daemon does not conflict with other already defined classes. For example, the sum of all “minimum values” in a tier must be less than 100%. On the other hand, if WLM is already in use, the administrator must ensure that other applications in the system do not cause the Topology Services daemon to be deprived of memory. One way to prevent other applications from being more privileged than the Topology Services daemon in regard to memory allocation is to place other applications in tiers other than tier 0.

If WLM is already active on the system when the new classes and rules are added, WLM needs to be restarted in order to recognize the new classes and rules.

Action 6 - Investigate IP communication problem: Probable causes of this problem are:

1. The Topology Services daemon was temporarily blocked.
2. The Topology Services daemon exited on the node.
3. IP communication problem, such as mbuf shortage or excessive adapter traffic.

Probable cause 1 and probable cause 2 are usually only possible when all the monitored adapters in the node are affected. This is because these are conditions that affect the daemon as a whole, and not just one of the adapters in a node.

Probable cause 3, on the other hand, may result in a single adapter in a node being considered as down. Follow the procedures described to diagnose symptom ″Node appears to go down and then up″, “Action 5 - Investigate hatsd problem” on page 247. If probable cause 1 on page 247 or probable cause 2 on page 247 is identified as the source of the problem, follow the repair procedures described under the same symptom.

If these causes are ruled out, the problem is likely related to IP communication. The instructions in ″Node appears to go down and then up″, “Action 5 - Investigate hatsd problem” on page 247 describe what communication parameters to monitor in order to pinpoint the problem.

To identify the network that is affected by the problem, issue command **errpt -J TS_DEATH_TR | more** and look for the entry **TS_DEATH_TR**. This is the syslog entry created when the local adapter stopped receiving heartbeat messages from its neighbor adapter. The neighbor’s address, which is listed in the error log entry, indicates which network is affected.

Action 7 - Investigate Group Services failure: This is most likely a problem in the Topology Services daemon, or a problem related to the communication between the daemon and the Topology Services library, which is used by the Group Services daemon. This problem may happen during Topology Services refresh in Linux.

When this problem occurs, the Group Services daemon exits and produces an error log entry with a LABEL of **GS_TS_RETCODE_ER**. This entry will have the Topology Services return code in the Detail Data field. Topology Services will produce an

error log entry with a LABEL of **TS_LIBERR_EM**. Follow the instructions in “Information to collect before contacting the IBM Support Center” on page 229 and contact the IBM Support Center.

Action 8 - Investigate problems after a refresh: Probable causes of this problem are:

- A refresh operation fails on the node.
- Adapters are configured with an incorrect address in the cluster configuration.

Verify whether all nodes were able to complete the refresh operation, by running “Operational test 8 - Check if configuration instance and security status are the same across all nodes” on page 240. If this test reveals that nodes are running with different Configuration Instances (from the **lssrc** command output), at least one node was unable to complete the refresh operation successfully.

Issue the command **errpt -J TS_* | more** on all nodes and look for **TS**_ Error Labels. The startup script log provides more details about this problem.

Other error log entries that may be present are:

- TS_REFRESH_ER
- TS_MACHLIST_ER
- TS_LONGLINE_ER
- TS_SPNODEDUP_ER, TS_HANODEDUP_ER, or TS_CTNODEDUP_ER
- TS_SPIPDUP_ER, TS_HAIPDUP_ER, or TS_CTIPDUP_ER
- TS_IPADDR_ER
- TS_KEY_ER

For information about each error log entry and how to correct the problem, see “Error information” on page 199.

If a node does not respond to the command: **lssrc -ls** *subsystem*, (the command hangs), this indicates a problem in the connection between Topology Services and the SRC subsystem. Such problems will also cause in the Topology Services daemon to be unable to receive the refresh request.

If no **TS**_ error log entry is present, and all nodes are responding to the **lssrc** command, and **lssrc** is returning different Configuration Instances for different nodes, contact the IBM Support Center.

If all nodes respond to the **lssrc** command, and the Configuration Instances are the same across all nodes, follow “Configuration verification test” on page 231 to find a possible configuration problem. Error log entry **TS_MISCFG_EM** is present if the adapter configuration collected by the configuration resource manager does not match the actual address configured in the adapter.

For problems caused by loss of connection with the AIX SRC, the Topology Services subsystem may be restarted. Be aware that issuing the **/usr/sbin/rsct/bin/cthatsctrl -k** command **will not work** because the connection with the AIX SRC subsystem was lost. To recover, perform these steps:

1. Issue the command:

   ```
   ps -ef | grep hats | grep -v grep
   ```

   to find the daemon’s *process_ID*:

The output of the command is similar to the following:

```
root 13446 8006 0 May 27 - 26:47 /usr/sbin/rsct/bin/hatsd -n 3
```

In this example, the *process_ID* is 13446.

2. Issue the command:

   ```
   kill process_ID
   ```

   This stops the Topology Services daemon.
3. If the AIX SRC subsystem does not restart the Topology Services subsystem automatically, issue this command:

   ```
   /usr/sbin/rsct/bin/cthatsctrl -s
   ```

   For HACMP, restarting the Topology Services daemon requires shutting down the HACMP cluster on the node, which can be done with the sequence:

   ```
   smit hacmp
                       Cluster Services
                           Stop Cluster Services
   ```

   After HACMP is stopped, find the process id of the Topology Services daemon and stop it, using the command:

   ```
   /usr/sbin/rsct/bin/topsvcsctrl
   ```

   instead of the command:

   ```
   /usr/sbin/rsct/bin/hatsctrl
   ```

   Now restart HACMP on the node using this sequence:

   ```
   smit hacmp
                        Cluster Services
                            Start Cluster Services
   ```

Follow the procedures in “Operational verification tests” on page 233 to ensure that the subsystem is behaving as expected across all nodes.

Note: In the HACMP/ES environment, **DO NOT STOP** the Topology Services daemon by issuing any of these commands.

- kill
- stopsrc
- topsvcsctrl -k

This is because stopping the Topology Services daemon while the cluster is up on the node results in the node being stopped by the HACMP cluster manager.

Action 9 - Investigate a node crash: If a node crashes, perform AIX system dump analysis. Probable causes of this problem are:

1. The Dead Man Switch timer was triggered, probably because the Topology Services daemon was blocked.
2. An AIX-related problem.

When the node restarts, issue the command:

```
errpt -J KERNEL_PANIC
```

to look for any AIX error log entries that were created when the node crashed. If this command produces an output like:

```
IDENTIFIER TIMESTAMP  T C RESOURCE_NAME  DESCRIPTION
225E3B63   0821085101 T S PANIC          SOFTWARE PROGRAM ABNORMALLY TERMINATED
```

then run:

```
errpt -a
```

to get details for the event. The output of the command may be similar to the following:

```
LABEL:          KERNEL_PANIC
IDENTIFIER:     225E3B63

Date/Time:       Tue Aug 21 08:51:29
Sequence Number: 23413
Machine Id:      000086084C00
Node Id:         c47n16
Class:           S
Type:            TEMP
Resource Name:   PANIC

Description
SOFTWARE PROGRAM ABNORMALLY TERMINATED
          Recommended Actions
          PERFORM PROBLEM DETERMINATION PROCEDURES

Detail Data
ASSERT STRING

PANIC STRING
RSCT Dead Man Switch Timeout for PSSP; halting non-responsive node
```

If the “RSCT Dead Man Switch Timeout for PSSP” string appears in the output above then this means that the crash was caused by the Dead Man Switch timer trigger. Otherwise, there is another source for the problem. For problems unrelated to the Dead Man Switch timer, contact the IBM Support Center.

If the dump was produced by the Dead Man Switch timer, it is likely that the problem was caused by the Topology Services daemon being blocked. HACMP/ES uses this mechanism to protect data in multi-tailed disks. When the timer is triggered, other nodes are already in the process of taking over this node’s resources, since Topology Services is blocked in the node. If the node was allowed to continue functioning, both this node and the node taking over this node’s disk would be concurrently accessing the disk, possibly causing data corruption.

The Dead Man Switch (DMS) timer is periodically stopped and reset by the Topology Services daemon. If the daemon gets blocked and does not have a chance to reset the timer, the timer-handling function runs, causing the node to crash. Each time the daemon resets the timer, the remaining amount left in the previous timer is stored. The smaller the remaining time, the closer the system is to triggering the timer. These “time-to-trigger” values can be retrieved with command:

```
/usr/sbin/rsct/bin/hatsdmsinfo
```

The output of this command is similar to:

```
Information for Topology Services -- HACMP/ES
DMS Trigger time: 8.000 seconds.
Last DMS Resets                        Time to Trigger (seconds)
11/11/99 09:21:28.272                   7.500
11/11/99 09:21:28.772                   7.500
11/11/99 09:21:29.272                   7.500
11/11/99 09:21:29.772                   7.500
11/11/99 09:21:30.272                   7.500
```

```
11/11/99 09:21:30.782                    7.490

DMS Resets with small time-to-trigger    Time to Trigger (seconds)
Threshold value: 6.000 seconds.
11/11/99 09:18:44.316                    5.540
```

If small “time-to-trigger” values are seen, the HACMP tunables described in “Action 5 - Investigate hatsd problem” on page 247 need to be changed, and the root cause for the daemon being blocked needs to be investigated. Small “time-to-trigger” values also result in an AIX error log entry with template **TS_DMS_WARNING_ST**. Therefore, when this error log entry appears, it indicates that the system is getting close to triggering the Dead Man Switch timer. Actions should be taken to correct the system condition that leads to the timer trigger.

Chapter 6. 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 **startrpdomain** 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.
- discusses diagnostic procedures and failure responses.

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 5, “The Topology Services subsystem” on page 185.

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*. For example, Event Management is a Group Services client subsystem. The Event Manager daemon on each node is 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 component of AIX.

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:

- **/usr/lib/libha_gs.a** (non-thread-safe version)
- **/usr/lib/libha_gs_r.a** (thread-safe version)

These path names are actually symbolic links to the files **/usr/sbin/rsct/lib/libha_gs.a** and **/usr/sbin/rsct/lib/libha_gs.r.a**, respectively. The symbolic links are placed in **/usr/lib** for ease of use. For serviceability, the actual libraries are placed in the **/usr/sbin/rsct/lib** directory. These libraries are supplied as shared libraries, also for serviceability.

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 11).

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 263.

Files and directories

The Group Services subsystem uses the following directories:

- **/var/ct/***cluster_name***/lck**, for lock files
- **/var/ct/***cluster_name***/log**, for log files
- **/var/ct/***cluster_name***/run**, for Group Services daemon current working directories
- **/var/ct/***cluster_name***/soc**, for socket files.

The /var/ct/*cluster_name*/lck directory (lock files)

In the **/var/ct/lck** directory, the **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.

The /var/ct/*cluster_name*/log directory (log files)

The **/var/ct/log** 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.

The /var/ct/*cluster_name*/run directory (daemon working files)

In the **/var/ct/run** directory, a directory called **cthags**. This directory is the current working directory for the Group Services daemon. If the Group Services daemon abnormally terminates, the core dump file is placed in this 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 5, “The Topology Services subsystem” on page 185.

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. The system partition name is an argument to the **hagsd** program in the SRC subsystem specification.

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

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 from the local peer domain 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 cthats** command. If the response indicates that the Topology Services subsystem is inoperative, try to restart it using the **startrpdomain** or **startrpnode** 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”.

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
                    Number of   Number of local
 Group name         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.

Diagnosing Group Services problems

This section discusses diagnostic procedures and failure responses for the Group Services (GS) component of RSCT. The list of known error symptoms and the associated responses are in the section “Error symptoms, responses, and recoveries” on page 287. A list of the information to collect before contacting the IBM Support Center is in the section “Information to collect before contacting the IBM Support Center” on page 276.

Requisite function

This is a list of the software directly used by the GS component of RSCT. Problems within the requisite software may manifest themselves as error symptoms in Group Services. If you perform all the diagnostic routines and error responses listed in this chapter, and still have problems with the GS component of RSCT, you should consider these components as possible sources of the error. They are listed with the most likely candidate first, least likely candidate last.

- Topology Services subsystem of RSCT
- System Resource Controller (SRC)
- **/var/ct** directory
- FFDC library
- UDP communication
- Unix-Domain sockets

Error information

Error Logs and templates

Table 20 on page 268 shows the error log templates used by Group Services.

- GS_ASSERT_EM
- GS_AUTH_DENIED_ST
- GS_CLNT_SOCK_ER
- GS_DEACT_FAIL_ST
- GS_DOM_MERGE_ER
- GS_DOM_NOT_FORM_WA

- GS_ERROR_ER
- GS_GLSM_ERROR_ER
- GS_GLSM_START_ST
- GS_GLSM_STARTERR_ER
- GS_GLSM_STOP_ST
- GS_INVALID_MSG_ER
- GS_MESSAGE_ST
- GS_START_ST
- GS_STARTERR_ER
- GS_STOP_ST
- GS_TS_RETCODE_ER
- GS_XSTALE_PRCLM_ER

When you retrieve an error log entry, look for the Detail Data section near the bottom of the entry.

Each entry refers to a particular instance of the Group Services daemon on the local node. One entry is logged for each occurrence of the condition, unless otherwise noted in the Detail Data section. The condition is logged on every node where the event occurred.

The Detail Data section of these entries is not translated to other languages. This section is in English.

The error type is:

- A - Alert (failure in a GS client)
- E - Error (failure in GS)
- I - Informational (status information)

Table 20. Error Log templates for Group Services

Label	Type	Diagnostic explanation and details
GS_ASSERT_EM	E	**Explanation:** The GS daemon produced a core dump. **Details:** The GS daemon encountered an irrecoverable assertion failure. This occurs only if the daemon core dumps due to a specific GS assertion failure. GS will be restarted automatically and the situation will be cleared. However, its state is not cleared and the system administrator must determine the cause of the failure. The REFERENCE CODE field in the Detail Data section may refer to the error log entry which caused this event. See “Information to collect before contacting the IBM Support Center” on page 276 and contact the IBM Support Center.
GS_AUTH_DENIED_ST	A	**Explanation:** An unauthorized user tried to access GS. **Details:** An unauthorized user tried to connect to the GS daemon. Standard fields indicate that GS daemon detected an attempt to connect from an unauthorized user. Detailed fields explain the detail information. Possibilities are: the user is not a **root** user, the user is not a member of the **hagsuser** group, or the user is not a supplemental member of the **hagsuser** group.

Table 20. Error Log templates for Group Services (continued)

Label	Type	Diagnostic explanation and details
GS_CLNT_SOCK_ER	E	**Explanation:** Warning or error on the Group Services client socket. **Details:** Group Services has an error on the client socket, or the **hagsuser** group is not defined. Standard fields indicate that Group Services received an error or warning condition on the client socket. Detailed fields explain what error or warning caused this problem.
GS_DEACT_FAIL_ST	I	**Explanation:** Failure of the deactivate script. **Details:** The GS daemon is unable to run the deactivate script. Standard fields indicate that the GS daemon is unable to run the script. Detailed fields give more information. The deactivate script may not exist, or system resources are not sufficient to run the deactivate script.
GS_DOM_MERGE_ER	A, E	**Explanation:** Two Group Services domains were merged. **Details:** Two disjoint Group Services domains are merged because Topology Services has merged two disjoint node groups into a single node group. There may be several nodes with the same entries. Detailed fields contains the merging node numbers. At the time of domain merge, GS daemons on the nodes that generate **GS_DOM_MERGE_ER** entries will exit and be restarted. After the restart, (by **GS_START_ST**) Group Services will clear this situation. The REFERENCE CODE field in the Detail Data section may refer to the error log entry that caused this event. See “Action 2 - Verify Status of Group Services Subsystem” on page 288. See “Information to collect before contacting the IBM Support Center” on page 276 and contact the IBM Support Center.
GS_DOM_NOT_FORM_WA	I	**Explanation:** A Group Services domain was not formed. **Details:** The GS daemon writes this entry periodically until the GS domain is formed. There may be several nodes in the same situation at the same time. The GS domain cannot be formed because: • On some nodes, Topology Services may be running but GS is not. • Nameserver recovery protocol is not complete. This entry is written periodically until the domain is established. The entry is written as follows: every 5, 30, 60, 90 minutes, and then once every two hours as long as the domain is not established. The domain establishment is recorded by a **GS_MESSAGE_ST** template label. The REFERENCE CODE field in the Detail Data section may refer to the error log entry that caused this event.
GS_ERROR_ER	A, E	**Explanation:** Group Services logic failure. **Details:** The GS daemon encountered an irrecoverable logic failure. Detailed fields describes what kind of error is encountered. The GS daemon exits due to the GS logic failure. Group Services will be restarted automatically and the situation will be cleared. However, if the state is not cleared, the administrator must determine what caused the GS daemon to terminate. The REFERENCE CODE field in the Detail Data section may refer to the error log entry that caused this event. See “Information to collect before contacting the IBM Support Center” on page 276 and contact the IBM Support Center.

Table 20. Error Log templates for Group Services (continued)

Label	Type	Diagnostic explanation and details
GS_GLSM_ERROR_ER	A, E	**Explanation:** Group Services GLSM daemon logic failure. **Details:** The Group Services GLSM daemon encountered an irrecoverable logic failure. Standard fields indicate that the daemon stopped. Detailed fields point to the error log entry created when the daemon started. The Group Services GLSM daemon exited due to the logic failure. The Group Services GLSM daemon will be restarted automatically and the situation will be cleared. However, if the state is not cleared, the administrator must determine what caused the problem. The standard fields are self-explanatory. The REFERENCE CODE field in the Detail Data section may refer to the error log entry that caused this event. See “Information to collect before contacting the IBM Support Center” on page 276 and contact the IBM Support Center.
GS_GLSM_START_ST	I	**Explanation:** Group Services GLSM Daemon started (AIX error log entry). **Details:** The Group Services GLSM daemon has started. Standard fields indicate that the daemon started. Detailed fields contain the path name of the log file. The Group Services GLSM subsystem was started by a user or by a process. Issue this command: `lssrc -l -s` *`glsm_subsystem`* If the daemon is started, the output will contain a status of ″active″ for **cthagsglsm**. Otherwise, the output will contain a status of ″inoperative″ for **cthagsglsm**.
GS_GLSM_STARTERR_ER	A, E	**Explanation:** Group Services GLSM daemon cannot be started. **Details:** The Group Services GLSM daemon encountered a problem during startup. Standard fields indicate that the daemon is stopped. Detailed fields point to the error log entry created when the daemon started. The GS daemon cannot be started because **exec** to **hagsglsmd** has failed. The AIX log entry may be the only remaining information about the cause of the problem after it is cleared.

Table 20. Error Log templates for Group Services (continued)

Label	Type	Diagnostic explanation and details
GS_GLSM_STOP_ST	I	**Explanation:** HAGSGLSM (HA Group Services GLobalized Switch Membership) daemon stopped. **Details:** The Group Services GLSM daemon was stopped by a user or by a process. Standard fields indicate that the daemon stopped. Detailed fields point to the error log entry created when the daemon started. If the daemon was stopped by the SRC, the word ″SRC″ will be present in the Detail Data. The REFERENCE CODE field in the Detail Data section may reference the error log entry that caused this event. Issue this command: `lssrc -l -s glsm_subsystem` If the daemon is stopped, the output will contain a status of ″inoperative″ for **cthagsglsm**. Otherwise, the output will contain a status of ″active″ for **cthagsglsm**.
GS_INVALID_MSG_ER	A, E	**Explanation:** The GS daemon received an unknown message. **Details:** The GS daemon received an incorrect or unknown message from another daemon. The transmitted messages may be corrupted on the wire, or a daemon sent a corrupted message. The GS daemon will restart and clear the problem. See “Information to collect before contacting the IBM Support Center” on page 276 and contact the IBM Support Center.
GS_MESSAGE_ST	I	**Explanation:** Group Services informational message **Details:** The GS daemon has an informational message about the Group Services activity, or condition. Detailed fields describes the information. It is one of the following: 1. The GS daemon is not connected to Topology Services. 2. The GS domain has not recovered or been established after a long time. 3. Any other message, which will be in the detailed field. The REFERENCE CODE field in the Detail Data section may refer to the error log entry that caused this event.
GS_START_ST	I	**Explanation:** Group Services daemon started. **Details:** The GS subsystem is started by a user or by a process. Detailed fields contain the log file name.
GS_STARTERR_ER	A, E	**Explanation:** Group Services cannot be started. **Details:** The GS daemon encountered a problem during startup. **Information about the cause of this problem may not be available once the problem is cleared.** The GS daemon cannot start because one of the following conditions occurred: 1. **exec** to **hagsd** failed. 2. The environment variables used by the startup scripts are not set properly. 3. Daemon initialization failed.

Table 20. Error Log templates for Group Services (continued)

Label	Type	Diagnostic explanation and details
GS_STOP_ST	I	**Explanation:** Group Services daemon stopped. **Details:** The GS daemon was stopped by a user or by a process. Detailed fields indicate how the daemon stops. If this was not intended, the system administrator must determine what caused the GS daemon to terminate. If the daemon was stopped by the SRC, ″SRC″ will be present in the Detail Data.
GS_TS_RETCODE_ER	A, E	**Explanation:** The Topology Services library detected an error condition. **Details:** The GS daemon received an incorrect or unknown message from another daemon. This entry refers to a particular instance of the Topology Services library on the local node. Standard fields indicate that Group Services received an error condition from Topology Services. Detailed fields contain the explanation and Topology Services library error number. The GS daemon will restart and clear the problem. The standard fields are self-explanatory. The REFERENCE CODE field in the Detail Data section may contain the Topology Services log entry that causes this event.
GS_XSTALE_PRCLM_ER	A, E	**Explanation:** Non-stale proclaim message was received. This means that inconsistent domain join request messages were received. **Details:** The local node received a valid domain join request (proclaim) message from his Nameserver twice. This should not happen in a normal situation. Detailed fields point to the error log entry of a NodeUp event. Topology Services reports inconsistent node down and up events between nodes. The GS daemon will restart and clear the problem. The REFERENCE CODE field in the Detail Data may reference the error log entry that caused this event. For more information, see the symptom ″Non-stale proclaim message received″ in “Error symptoms, responses, and recoveries” on page 287. See “Information to collect before contacting the IBM Support Center” on page 276 and contact the IBM Support Center.

Dump information

Group Services creates a core dump automatically when certain errors occur, and also provides service information that can be obtained automatically by the **ctsnap** command.

Core dump

A core dump is generated by the Group Services daemon if it encounters an undefined condition. It contains normal information saved in a core dump. The dump is specific to a particular instance of the GS daemon on the local node. Other nodes may have a similar core dump. Each core dump file is approximately 10MB in size.

The core dumps are located in: **/var/ct/***cluster_name***/run/cthags/core***. For a HACMP node, the core dumps are located in: **/var/ha/run/grpsvcs.***cluster***/core*** and **/var/ha/run/grpglsm.***cluster***/core***.

Core dumps are created automatically when:

- One of the GS daemons invokes an **assert()** statement if the daemon state is undefined or encounters an undefined condition by design.
- The daemon attempts an incorrect operation, such as division by zero.
- The daemon receives a segmentation violation signal for accessing its data incorrectly.

A core dump is created manually by issuing the command:

```
kill -6 pid_of_daemon
```

where *pid_of_daemon* is obtained by issuing the command:

```
lssrc -s cthags
```

The core dump is valid as long as the executable file **/usr/sbin/rsct/bin/hagsd** is not replaced. Copy the core dumps and the executable file to a safe place.

To verify the core dump, issue this command:

```
dbx /usr/sbin/rsct/bin/hagsd core_file
```

where *core_file* is one of the **core*** files described previously.

Good results are indicated by output similar to:

```
Type 'help' for help.
 reading symbolic information ...
[using memory image in core]
IOT/Abort trap in evt._pthread_ksleep [/usr/lib/libpthreads.a]
at 0xd02323e0 ($t6) 0xd02323e0 (_pthread_ksleep+0x9c) 80410014
 lwz    r2,0x14(r1)
```

Error results may look like one of the following:

1. This means that the current executable file was not the one that created the core dump.

   ```
   Type 'help' for help.
   Core file program (hagsd) does not match current program (core ignored)
   reading symbolic information ...
   (dbx)
   ```

2. This means that the dump is incomplete due to lack of disk space.

   ```
   Type 'help' for help.
   warning: The core file is truncated.  You may need to increase the ulimit
   for file and coredump, or free some space on the filesystem.
   reading symbolic information ...
   [using memory image in core]

   IOT/Abort trap in evt._pthread_ksleep [/usr/lib/libpthreads.a]
   at 0xd02323e0
   0xd02323e0 (_pthread_ksleep+0x9c) 80410014
           lwz   r2,0x14(r1)
   (dbx)
   ```

ctsnap dump

This dump contains diagnostic data used for RSCT problem determination. It is a collection of configuration data, log files and other trace information for the RSCT components.

Trace information

ATTENTION - READ THIS FIRST

Do *NOT* activate this trace facility until you have read this section completely, and understand this material. If you are not certain how to properly use this facility, or if you are not under the guidance of IBM Service, do *NOT* activate this facility.

Activating this facility may result in degraded performance of your system. Activating this facility may also result in longer response times, higher processor loads, and the consumption of system disk resources. Activating this facility may also obscure or modify the symptoms of timing-related problems.

The log files, including the Group Services Trace logs and startup logs, are preserved as long as their total size does not exceed the default value of 5MB. If the total size is greater than 5MB, the oldest log file is removed at Group Services startup time. The total log size can be changed by issuing the **cthagstune** command.

GS service log trace

The GS service log contains a trace of the GS daemon. It is intended for IBM Support Center use only, and written in English. It refers to a particular instance of the GS daemon running on the local node. When a problem occurs, logs from multiple nodes are often needed.

If obtaining logs from all nodes is not feasible, collect logs from these nodes:

- The node where the problem was detected
- The Group Services Nameserver (NS) node. To find the NS node, see “How to find the GS nameserver (NS) node” on page 276.
- If the problem is related to a particular GS group, the Group Leader node of the group that is experiencing the problem. To find a Group Leader node for a specific group, see “How to find the Group Leader (GL) node for a specific group” on page 277.

Service log short tracing is always in effect. Service log long tracing is activated by this command:

```
traceson -l -s cthags
```

The trace is deactivated, (reverts to short tracing) by issuing this command:

```
tracesoff -s cthags
```

The trace may produce 20MB or more of data, depending on GS activity level and length of time that the trace is running. Ensure that there is adequate space in the directory **/var/ct**.

The trace is located in:
/var/ct/*cluster_name***log/cthags/cthags***_nodenum_incarnation***.***cluster_name*. where *incarnation* is an increasing integer set by the GS daemon. This value can be obtained from the **NodeId** field of the command:

```
hagsns -l -s cthags
```

The long trace contains this information:

1. Each Group Services protocol message sent or received
2. Each significant processing action as it is started or finished
3. Details of protocols being run

For many of the cases, log files from multiple nodes must be collected. The other nodes' log files must be collected before they wrap or are removed. By default, during the long tracing, log files will expand to a maximum of 5 times the configured log size value.

To change the configured value of the log size on a node, issue this command:

```
cthagstune -l new_length
```

where *new_length* is the number of lines in the trace log file. Then, restart the GS daemon.

To change the configured value on a HACMP node, perform these steps:

1. Issue this command: **smit hacmp**.
2. Select **Cluster Configuration.**
3. Select **Cluster Topology**.
4. Select **Configure Topology Services and Group Services**.
5. Select **Change/Show Topology and Group Services Configuration**.
6. Select **Group Services log length** (number of lines).
7. Enter the number of lines for each Group Services log file.

When the log file reaches the line number limit, the current log is saved into a file with a suffix of **.bak**. The original file is then truncated. With the ″long″ trace option, the default of 5000 lines should be enough for only 30 minutes or less of tracing.

Each time the daemon is restarted, a new log file is created. Only the last 5 log files are kept.

Long tracing should be activated on request from IBM Service. It can be activated (for about one minute, to avoid overwriting other data in the log file) when the error condition is still present.

Each entry is in the format: *date message*.

The ″short″ form of the service log trace is always running. It contains this information:

1. Each Group Services protocol message sent or received.
2. Brief information for significant protocols being run.
3. Significant information for possible debugging.

GS service log trace - summary log

The GS service log - summary log contains a trace of the GS daemon, but records only important highlights of daemon activity. This log does not record as much information as the GS service log, and therefore it will not wrap as quickly as the GS service log. This log is more useful in diagnosing problems whose origin occurred a while ago. All information in this log is also recorded in the GS service log, provided that the log has not yet wrapped. The GS service log - summary log is intended for IBM Support Center use only, and written in English. It refers to a

particular instance of the GS daemon running on the local node. When a problem occurs, both logs from multiple nodes are often needed.

The trace is located in:

- **/var/ct/***cluster_name***log/cthags**_*node_incarnation***.***cluster_name***.long**
- **/var/ha/log/grpsvcs**_*node_incarnation***.***domain***.long** on HACMP nodes

where *incarnation* is an increasing integer set by the GS daemon. This value can be obtained from the **NodeId** field of the command:

```
hagsns -l -s gssubsys
```

Group Services startup script log

This log contains the GS daemon's environment variables and error messages where the startup script cannot start the daemon. The trace refers to a particular instance of the GS startup script running on the local node. This trace is always running. One file is created each time the startup script runs. The size of the file varies from 5KB to 10KB.

It is located in: **/var/ct/***cluster_name***log/cthags.default.***node_incarnation*.

The data in this file is in English. This information is for use by the IBM Support Center. The format of the data is the same as that of the GS Service Log Trace, ″long″ option.

Information to collect before contacting the IBM Support Center

Collect information from these nodes:

1. Nodes that exhibit the problem
2. GS nameserver (NS) node. See "How to find the GS nameserver (NS) node".
3. Group Leader (GL) node, if the problem is related to a particular group. See "How to find the Group Leader (GL) node for a specific group" on page 277.

Issue the **ctsnap** command to collect the necessary information.

See Chapter 7, "How to contact the IBM Support Center" on page 295.

How to find the GS nameserver (NS) node

Perform these steps to find out which node is the GS nameserver node.

1. Issue the **lssrc** command:

   ```
   lssrc -ls cthags
   ```

   If the output is similar to:

   ```
    Subsystem         Group            PID     Status
     cthags           cthags           14460   active
   0 locally-connected clients.
   HA Group Services domain information:
   Domain established by node 6
   Number of groups known locally: 1
                       Number of   Number of local
   Group name          providers   providers/subscribers
   cssMembership             9           1           0
   ```

you can obtain the node number of the nameserver. In this case, it is node 6, from the line `Domain established by node 6`. Do not perform any of the remaining steps.

2. If the output indicates `Domain not established`, wait to see if the problem is resolved in a few minutes, and if not, proceed to “Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered” on page 281.
3. There is another command that is designed for the NS status display. Issue the **hagsns** command:

   ```
   /usr/sbin/rsct/bin/hagsns -s cthags
   ```

   Output is similar to:

```
HA GS NameServer Status
NodeId=1.16, pid=14460, domainId=6.14, NS established, CodeLevel=GSlevel(DRL=8)
NS state=kCertain, protocolInProgress=kNoProtocol, outstandingBroadcast=kNoBcast
Process started on Jun 19 18:34:20, (10d 20:19:22) ago, HB connection took (19:14:9).
Initial NS certainty on Jun 20 13:48:45, (10d 1:4:57) ago, taking (0:0:15).
Our current epoch of Jun 23 13:05:19 started on (7d 1:48:23), ago.
Number of UP nodes: 12
List of UP nodes:  0 1 5 6 7 8 9 11 17 19 23 26
```

   In this example, `domainId=6.14` means that node 6 is the NS node. Note that the `domainId` consists of a node number and an incarnation number. The incarnation number is an integer, incremented whenever the GS daemon is started.

4. The **hagsns** command output on the NS also displays the list of groups:

```
We are: 6.14 pid: 10094 domaindId = 6.14 noNS = 0 inRecovery = 0, CodeLevel=GSlevel(DRL=8)
NS state=kBecomeNS, protocolInProgress = kNoProtocol, outstandingBroadcast = kNoBcast
Process started on Jun 19 18:35:55, (10d 20:22:39) ago, HB connection took (0:0:0).
Initial NS certainty on Jun 19 18:36:12, (10d 20:22:22) ago, taking (0:0:16).
Our current epoch of certainty started on Jun 23 13:05:18, (7d 1:53:16) ago.
Number of UP nodes: 12
List of UP nodes: 0 1 5 6 7 8 9 11 17 19 23 26
List of known groups:
2.1 ha_gpfs: GL: 6 seqNum: 30 theIPS: 6 0 8 7 5 11 lookupQ:
```

   In this example, the group is **ha_gpfs**.

How to find the Group Leader (GL) node for a specific group

There are two ways of finding the Group Leader node of a specific group:

1. The **hagsns** command on the NS displays the list of membership for groups, including their Group Leader nodes. To use this method:
   a. Find the NS node from “How to find the GS nameserver (NS) node” on page 276.
   b. Issue the following command on the NS node:

      ```
      /usr/sbin/rsct/bin/hagsns -s cthags
      ```

      The output is similar to:

```
HA GS NameServer Status
NodeId=6.14, pid=10094, domainId=6.14, NS established, CodeLevel=GSlevel(DRL=8)
NS state=kBecomeNS, protocolInProgress=kNoProtocol, outstandingBroadcast=kNoBcast
Process started on Jun 19 18:35:55, (10d 20:22:39) ago, HB connection took (0:0:0).
Initial NS certainty on Jun 19 18:36:12, (10d 20:22:22) ago, taking (0:0:16).
```

```
Our current epoch of certainty started on Jun 23 13:05:18, (7d 1:53:16) ago.
Number of UP nodes: 12
List of UP nodes:  0 1 5 6 7 8 9 11 17 19 23 26
List of known groups:
2.1 ha_gpfs: GL: 6 seqNum: 30 theIPS: 6 0 8 7 5 11 lookupQ:
```

The bottom few lines display the group membership information. For example, the GL node of the group **ha_gpfs** is node 6, and its participating nodes are ″6 0 8 7 5 11″.

2. If you need only the GL node of a specific group, the **hagsvote** command gives the answer. Issue the command:

   ```
   hagsvote -s cthags
   ```

   The output is similar to:

   ```
   Number of groups: 3
   Group slot #[0] Group name [HostMembership] GL node [Unknown] voting data:
   No protocol is currently executing in the group.
   ------------------------------------------------

   Group slot #[1] Group name [enRawMembership] GL node [Unknown] voting data:
   No protocol is currently executing in the group.
   ------------------------------------------------

   Group slot #[2] Group name [enMembership] GL node [6] voting data:
   No protocol is currently executing in the group.
   ```

   In this output, node 6 is the GL node of the group **enMembership**. If the GL node is `Unknown`, this indicates that no client applications tried to use the group on this node, or the group is one of the adapter groups.

Diagnostic procedures

Installation verification test

This test determines whether RSCT has been successfully installed. Group Services is a part of RSCT. Perform the following steps:

1. Issue the command:

   ```
   lslpp -l | grep rsct
   ```

   Good results are indicated by output similar to:

   ```
   rsct.basic.hacmp   1.2.0.0 COMMITTED RS/6000 Cluster Technology (HACMP domains)
   rsct.basic.rte     1.2.0.0 COMMITTED RS/6000 Cluster Technology (all domains)
   rsct.basic.sp      1.2.0.0 COMMITTED RS/6000 Cluster Technology (SP domains)
   rsct.clients.hacmp 1.2.0.0 COMMITTED RS/6000 Cluster Technology (HACMP domains)
   rsct.clients.rte   1.2.0.0 COMMITTED RS/6000 Cluster Technology (all domains)
   rsct.clients.sp    1.2.0.0 COMMITTED RS/6000 Cluster Technology (SP domains)
   rsct.core.utils    1.2.0.  COMMITTED RS/6000 Cluster Technology (all domains)
   ```

   Error results are indicated by no output from the command.

2. Issue the command:

   ```
   lppchk -c "rsct*"
   ```

   Good results are indicated by the absence of error messages and the return of a zero exit status from this command. The command produces no output if it succeeds.

Error results are indicated by a non-zero exit code and by error messages similar to these:

```
lppchk: 0504-206  File /usr/lib/nls/msg/en_US/hats.cat could not be located.
lppchk: 0504-206  File /usr/sbin/rsct/bin/hatsoptions could not be located.
lppchk: 0504-208  Size of /usr/sbin/rsct/bin/phoenix.snap is 29356,
                      expected value was 29355.
```

Some error messages may appear if an EFIX is applied to a file set. An EFIX is an emergency fix, supplied by IBM, to correct a specific problem.

If the test fails, the following file sets need to be installed:

1. **rsct.basic.rte**
2. **rsct.core.utils**
3. **rsct.clients.rte**
4. **rsct.basic.sp**
5. **rsct.clients.sp**
6. **rsct.basic.hacmp**
7. **rsct.clients.hacmp**

If this test is successful, proceed to “Configuration verification test”.

Configuration verification test

This test verifies that Group Services on a node has the configuration data that it needs. Perform the following steps:

1. Perform the Topology Services Configuration verification diagnosis. See “Diagnosing Topology Services problems” on page 199.
2. Verify that the **cthats** and **cthags** subsystems are added, by issuing the **lssrc -a** command. If **lssrc -a** does not contain **cthats** or **cthags**, or **lssrc -s cthats** and **lssrc -s cthags** cause an error, the above setup may not be correct.
3. Verify the cluster status by issuing the command: **/usr/sbin/rsct/bin/lsclcfg**. The output of this command must contain:

   ```
   cluster_name cluster_name
   node_number local-node-number
   ```

   If anything is missing or incorrect, the setup procedure may not be correct.

If this test is successful, proceed to “Operational verification tests”.

Operational verification tests

The following information applies to the diagnostic procedures that follow:

- Subsystem Name: **cthags**
- Service and User log files: **/var/ct/***cluster_name***log/cthags/cthags_***
- Startup Script log: **/var/ct/***cluster_name***log/cthags/cthags.default***

Operational test 1 - Verify that Group Services is working properly: Issue the **lssrc** command:

```
lssrc -ls cthags
```

Good results are indicated by an output similar to:

```
Subsystem         Group            PID     Status
 cthags           cthags           22962   active
```

```
1 locally-connected clients.  Their PIDs:
25028(haemd)
HA Group Services domain information:
Domain established by node 21
Number of groups known locally: 2
                   Number of   Number of local
Group name         providers   providers/subscribers
ha_gpfs             6            1           0
```

Error results are indicated by one of the following:

1. A message similar to:

   ```
   0513-036 The request could not be passed to the cthags subsystem.
       Start the subsystem and try your command again.
   ```

   This means that the GS daemon is not running. The GS subsystem is down. Proceed to “Operational test 2 - Determine why the Group Services subsystem is not active” on page 281.

2. A message similar to:

   ```
   0513-085 The cthags Subsystem is not on file.
   ```

   This means that the GS subsystem is not defined to the SRC.

   Use the **lsrpnode** command to determine whether or not the node is online in the cluster. For complete syntax information on the **lsrpnode** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*.

3. Output similar to:

   ```
   Subsystem         Group            PID     Status
   cthags            cthags           7350     active
    Subsystem cthags trying to connect to Topology Services.
   ```

   This means that Group Services is not connected to Topology Services. Check the Topology Services subsystem. See “Diagnosing Topology Services problems” on page 199.

4. Output similar to:

   ```
   Subsystem         Group            PID     Status
   cthags            cthags          35746    active
    No locally-connected clients.
    HA Group Services domain information:
    Domain not established.
    Number of groups known locally: 0
   ```

   This means that the GS domain is not established. This is normal during the Group Services startup period. Retry this test after about three minutes. If this situation continues, perform “Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered” on page 281.

5. Output similar to:

   ```
   Subsystem         Group            PID     Status
   cthags            cthags          35746    active
    No locally-connected clients.
    HA Group Services domain information:
    Domain is recovering.
    Number of groups known locally: 0
   ```

This means that the GS domain is recovering. It is normal during Group Services domain recovery. Retry this test after waiting three to five minutes. If this situation continues, perform "Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered".

6. An output similar to the **Good results**, but no **cssMembership** group is shown on nodes that have the SP switch. Proceed to "Operational test 7 - Verify the HAGSGLSM (Group Services GLobalized Switch Membership) subsystem" on page 285.

Operational test 2 - Determine why the Group Services subsystem is not active: Issue the command:

```
errpt -N cthags
```

and look for an entry for the *cthags*. It appears under the RESOURCE_NAME heading.

If an entry is found, issue the command:

```
errpt -a -N cthags
```

to get details about error log entries. The entries related to Group Services are those with LABEL beginning with **GS**_.

The error log entry, together with its description in "Error Logs and templates" on page 267, explains why the subsystem is inactive.

If there is no **GS**_ error log entry explaining why the subsystem went down or could not start, it is possible that the daemon may have exited abnormally. Look for an error entry with LABEL of CORE_DUMP and PROGRAM NAME of **hagsd**, by issuing the command:

```
errpt -J CORE_DUMP
```

If this entry is found, see "Information to collect before contacting the IBM Support Center" on page 276 and contact the IBM Support Center.

Another possibility when there is no **GS**_ error log entry is that the Group Services daemon could not be loaded. In this case, a message similar to the following may be present in the Group Services startup log:

```
0509-036 Cannot load program hagsd because of the following errors:
0509-026 System error: Cannot run a file that does not have a valid format.
```

The message may refer to the Group Services daemon, or to some other program invoked by the startup script **cthags**. If this error is found, see "Information to collect before contacting the IBM Support Center" on page 276 and contact the IBM Support Center.

For errors where the daemon did start up but then exited during initialization, detailed information about the problem is in the Group Services error log.

Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered: The **hagsns** command is used to determine the nameserver (NS) state and characteristics. Issue the command:

```
hagsns -s cthags
```

The output is similar to:

```
HA GS NameServer Status
NodeId=0.32, pid=18256, domainId=0.Nil, NS not established, CodeLevel=GSlevel(DRL=8)
The death of the node is being simulated.
NS state=kUncertain, protocolInProgress=kNoProtocol, outstandingBroadcast=kNoBcast
Process started on Jun 21 10:33:08, (0:0:16) ago, HB connection took (0:0:0).
Our current epoch of uncertainty started on Jun 21 10:33:08, (0:0:16) ago.
Number of UP nodes: 1
List of UP nodes:  0
```

Error results are indicated by output of `NS state is kUncertain`, with the following considerations:

1. `kUncertain` is normal for a while after Group Services startup.
2. Group Services may have instructed Topology Services to simulate a node death. This is so that every other node will see the node down event for this local node. This simulating node death state will last approximately two or three minutes.

If this state does not change or takes longer than two or three minutes, proceed to check Topology Services. See "Diagnosing Topology Services problems" on page 199.

If the Group Services daemon is not in `kCertain` or `kBecomeNS` state, and is waiting for the other nodes, the **hagsns** command output is similar to:

```
HA GS NameServer Status
NodeId=11.42, pid=21088, domainId=0.Nil, NS not established, CodeLevel=GSlevel(DRL=8)
NS state=kGrovel, protocolInProgress=kNoProtocol, outstandingBroadcast=kNoBcast
Process started on Jun 21 10:52:13, (0:0:22) ago, HB connection took (0:0:0).
Our current epoch of uncertainty started on Jun 21 10:52:13, (0:0:22) ago.
Number of UP nodes: 2
List of UP nodes:  0 11
Domain not established for (0:0:22).
  Currently waiting for node 0
```

In the preceding output, this node is waiting for an event or message from node 0 or for node 0. The expected event or message differs depending on the NS state which is shown in the second line of the **hagsns** command output.

Analyze the `NSstate` as follows:

1. `kGrovel` means that this node believes that the waiting node (node 0 in this example) will become his NS. This node is waiting for node 0 to acknowledge it (issue a Proclaim message).
2. `kPendingInsert` or `kInserting` means that the last line of the **hagsns** command output is similar to:

   ```
   Domain not established for (0:0:22).  Currently waiting for node 0.1
   ```

   This node received the acknowledge (Proclaim or InsertPhase1 message) and is waiting for the next message (InsertPhase1 or Commit message) from the NS (node 0).

   If this state does not change to `kCertain` in a two or three minutes, proceed to "Operational test 1 - Verify that Group Services is working properly" on page 279, for Topology Services and Group Services on the waiting node (node 0 in this example).
3. `kAscend`, `kAscending`, `kRecoverAscend`, or `kRecoverAscending` means that the last line of the **hagsns** command output is similar to:

```
Domain not established for (0:0:22).  Waiting for 3 nodes: 1 7 6
```

If there are many waiting nodes, the output is similar to:

```
Domain not established for(0:0:22).Waiting for 43 nodes: 1 7 6 9 4 ....
```

This node is trying to become a nameserver, and the node is waiting for responses from the nodes that are listed in the **hagsns** command output. If this state remains for between three and five minutes, proceed to “Operational test 1 - Verify that Group Services is working properly” on page 279, for Topology Services and Group Services on the nodes that are on the waiting list.

4. `kKowtow` or `kTakeOver` means that the last line of the **hagsns** command output is similar to:

```
Domain not recovered for (0:0:22).  Currently waiting for node 0.1
```

After the current NS failure, this node is waiting for a candidate node that is becoming the NS. If this state stays too long, proceed to “Operational test 1 - Verify that Group Services is working properly” on page 279, for the Topology Services and Group Services on the node that is in the waiting list.

In this output, the value `0.1` means the following:

- The first number (″0″) indicates the node number that this local node is waiting for.
- The second number(″1″) is called the incarnation number, which is increased by one whenever the GS daemon starts.

Therefore, this local node is waiting for a response from the GS daemon of node 0, and the incarnation is 1.

Operational test 4 - Verify whether a specific group is found on a node: Issue the **lssrc** command:

```
lssrc -ls  cthags
```

Error results are indicated by outputs similar to the **error results** of “Operational test 1 - Verify that Group Services is working properly” on page 279 through “Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered” on page 281.

Good results are indicated by an output similar to:

```
Subsystem         Group             PID     Status
 cthags           cthags           22962    active
1 locally-connected clients.  Their PIDs:
25028(haemd)
HA Group Services domain information:
Domain established by node 21
Number of groups known locally: 1
                   Number of   Number of local
Group name         providers   providers/subscribers
ha_gpfs             6            1           0
```

In this output, examine the `Group name` field to see whether the requested group name exists. For example, the group **ha_gpfs** has 1 local provider, 0 local subscribers, and 6 total providers.

For more information about the given group, issue the **hagsns** command:

```
hagsns -s cthags
```

on the NS node. The output is similar to:

```
HA GS NameServer Status
NodeId=6.14, pid=10094, domainId=6.14, NS established, CodeLevel=GSlevel(DRL=8)
NS state=kBecomeNS, protocolInProgress=kNoProtocol, outstandingBroadcast=kNoBcast
Process started on Jun 19 18:35:55, (10d 20:22:39) ago, HB connection took (0:0:0).
Initial NS certainty on Jun 19 18:36:12, (10d 20:22:22) ago, taking (0:0:16).
Our current epoch of certainty started on Jun 23 13:05:18, (7d 1:53:16) ago.
Number of UP nodes: 12
List of UP nodes:  0 1 5 6 7 8 9 11 17 19 23 26
List of known groups: 2.1 ha_gpfs: GL: 6 seqNum: 30 theIPS: 6 0 8 7 5 11 lookupQ:
```

In the last line, the nodes that have the providers of the group **ha_gpfs** are 6 0 8 7 5 11.

Operational test 5 - Verify whether the cssMembership or css1Membership groups are found on a node: If "Operational test 1 - Verify that Group Services is working properly" on page 279 through "Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered" on page 281 succeeded, issue the following command:

```
lssrc -ls subsystem_name
```

The output is similar to:

```
Subsystem         Group            PID     Status
cthags            cthags           22962   active
2 locally-connected clients.  Their PIDs:
20898(hagsglsmd) 25028(haemd)
HA Group Services domain information:
Domain established by node 21
Number of groups known locally: 2
                   Number of   Number of local
Group name         providers   providers/subscribers
cssMembership           10           1           0
ha_em_peers             6            1           0
```

In the preceding output, the **cssMembership** group has 1 local provider. Otherwise, the following conditions apply:

1. No **cssMembership** or **css1Membership** exists in the output.

   There are several possible causes:

   a. **/dev/css0** or **/dev/css1** devices are down.

      Perform switch diagnosis.

   b. Topology Services reports that the switch is not stable.

      Issue the following command:

      ```
      lssrc -ls hats_subsystem
      ```

      where *hats_subsystem* is **cthats**, or, on HACMP nodes, **topsvcs**.

      The output is similar to:

      ```
      Subsystem       Group           PID     Status
      cthats          cthats          17058   active
      Network Name   Indx Defd Mbrs St Adapter ID      Group ID
      ```

```
SPether        [0]   15    2  S 9.114.61.65     9.114.61.125
SPether        [0] en0          0x37821d69      0x3784f3a9
HB Interval = 1 secs. Sensitivity = 4 missed beats
SPswitch       [1]   14    0  D 9.114.61.129
SPswitch       [1] css0
HB Interval = 1 secs. Sensitivity = 4 missed beats
  1 locally connected Client with PID:
hagsd( 26366)
  Configuration Instance = 926456205
  Default: HB Interval = 1 secs. Sensitivity = 4 missed beats
  Control Workstation IP address = 9.114.61.125
  Daemon employs no security
  Data segment size 7044 KB
```

Find the first `SPswitch` row in the `Network Name` column. Find the `St` (state) column in the output. At the intersection of the first `SPswitch` row and state column is a letter. If it is not **S**, wait for few minutes longer since the Topology Services SPswitch group is not stable. If the state stays too long as **D** or **U**, proceed to Topology Services diagnosis. See “Diagnosing Topology Services problems” on page 199. If the state is **S**, proceed to Step 1c. In this example, the state is **D**.

The state has the following values:

- **S** - stable or working correctly
- **D** - dead, or not working
- **U** - unstable (not yet incorporated)

c. **HAGSGLSM** is not running or waiting for Group Services protocols.

Proceed to “Operational test 7 - Verify the HAGSGLSM (Group Services GLobalized Switch Membership) subsystem”.

2. **cssMembership** or **css1Membership** exist in the output, but the number of local providers is zero.

Proceed to “Operational test 7 - Verify the HAGSGLSM (Group Services GLobalized Switch Membership) subsystem”.

Operational test 7 - Verify the HAGSGLSM (Group Services GLobalized Switch Membership) subsystem: Issue the following command:

```
lssrc -ls glsm_subsystem
```

where *glsm_subsystem* is **cthagsglsm**, or, on HACMP nodes, **grpglsm**.

Good results are indicated by output similar to:

- On the control workstation,

```
Subsystem    Group          PID     Status
cthagsglsm   cthags         22192   active
Status information for subsystem hagsglsm.c47s:
Connected to Group Services.
 Adapter  Group                 Mbrs   Joined  Subs'd  Aliases
 css0     (device does not exist)
          cssMembership         0       No      Yes     -
 css1     (device does not exist)
          css1Membership        0       No      Yes     -
 ml0      ml0Membership         -       No      -
Aggregate Adapter Configuration
 The current configuration id is 0x1482933.
 ml0[css0] Nodes: 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
 ml0[css1] Nodes: 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
```

- On other nodes,

```
Subsystem         Group            PID     Status
cthagsglsm        cthags           16788   active
Status information for subsystem cthagsglsm:
Connected to Group Services.
 Adapter  Group                  Mbrs   Joined  Subs'd  Aliases
 css0     cssRawMembership      16      -       Yes     1
          cssMembership         16      Yes     Yes     -
 css1     css1RawMembership     16      -       Yes     1
          css1Membership        16      Yes     Yes     -
 ml0      ml0Membership         16      Yes     -       cssMembership
Aggregate Adapter Configuration
 The current configuration id is 0x23784582.
 ml0[css0] Nodes: 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
 ml0[css1] Nodes: 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
```

Error results are indicated by one of the following outputs:

1. A message similar to:

   ```
   0513-036 The request could not be passed to the cthags subsystem.
            Start the subsystem and try your command again.
   ```

   This means that the HAGSGLSM daemon is not running. The subsystem is down. Issue the **errpt** command and look for an entry for the subsystem name. Proceed to “Operational test 2 - Determine why the Group Services subsystem is not active” on page 281.

2. A message similar to:

   ```
   0513-085 The cthagsglsm Subsystem is not on file.
   ```

   This means that the HAGSGLSM subsystem is not defined to the AIX SRC.

   In HACMP/ES, HACMP may have not been installed on the node. Check the HACMP subsystem.

3. Output similar to:

   ```
   Subsystem      Group            PID     Status
   cthagsglsm     cthags           26578   active
   Status information for subsystem cthagsglsm:
   Not yet connected to Group Services after 4 connect tries
   ```

   HAGSGLSM is not connected to Group Services. The Group Services daemon is not running. If the state is **S**, proceed to “Operational test 1 - Verify that Group Services is working properly” on page 279 for Group Services subsystem verification.

4. Output similar to:

   ```
   Subsystem         Group            PID     Status
   cthagsglsm        cthags            16048   active
   Status information for subsystem bhagsglsm:
   Waiting for Group Services response.
   ```

   HAGSGLSM is being connected to Group Services. Wait for a few seconds. If this condition does not change after several seconds, proceed to “Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered” on page 281.

5. Output similar to:

```
Subsystem         Group            PID     Status
cthagsglsm        cthags             26788   active
Status information for subsystem hagsglsm:
Connected to Group Services.
 Adapter  Group                  Mbrs   Joined  Subs'd  Aliases
 css0     cssRawMembership      -       -       No      -
          cssMembership         16       No     No      -
 css1     css1RawMembership     15      -       Yes     1
          css1Membership        15      Yes     Yes     -
 ml0      ml0Membership          -      -      -        -
Aggregate Adapter Configuration
 The current configuration id is 0x23784582.
 ml0[css0] Nodes: 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
 ml0[css1] Nodes: 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
```

On nodes that have the switch, the line "cssRawMembership" has No in the Subs'd column.

Check Topology Services to see whether the switch is working. Issue the command:

```
lssrc -ls hats_subsystem
```

The output is similar to:

```
Subsystem         Group            PID     Status
cthats            cthats             25074   active
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
SPether        [0]   15   11  S 9.114.61.65     9.114.61.193
SPether        [0] en0          0x376d296c      0x3784fdc5
HB Interval = 1 secs. Sensitivity = 4 missed beats
SPswitch       [1]   14    8  S 9.114.61.129    9.114.61.154
SPswitch       [1] css0         0x376d296d      0x3784fc48
HB Interval = 1 secs. Sensitivity = 4 missed beats
  1 locally connected Client with PID:
hagsd( 14460)
  Configuration Instance = 925928580
  Default: HB Interval = 1 secs. Sensitivity = 4 missed beats
  Control Workstation IP address = 9.114.61.125
  Daemon employs no security
  Data segment size 7052 KB
```

Find the first row under Network Name with SPswitch. Find the column with heading St (state). Intersect this row and column. If the value at the intersection is not **S**, see **TS_LOC_DOWN_ST** on page 209 and proceed to "Action 3 - Investigate local adapter problems" on page 245.

If the state is **S**, proceed to "Operational test 1 - Verify that Group Services is working properly" on page 279 to see whether the Group Services domain is established or not.

Error symptoms, responses, and recoveries

Use the following table to diagnose problems with Group Services. Locate the symptom and perform the action described in the following table:

Table 21. Group Services symptoms

Symptom	Error label	Recovery
GS daemon cannot start.	GS_STARTERR_ER	See "Action 1 - Start Group Services daemon" on page 288.

Table 21. Group Services symptoms (continued)

Symptom	Error label	Recovery
GS domains merged.	GS_DOM_MERGE_ER	See “Action 2 - Verify Status of Group Services Subsystem”.
GS clients cannot connect or join the GS daemon.	The following errors may be present: GS_AUTH_DENIED_ST GS_CLNT_SOCK_ER GS_DOM_NOT_FORM_WA	See “Action 3 - Correct Group Services access problem” on page 289.
GS daemon died unexpectedly.	The following errors may be present: GS_ERROR_ER GS_DOM_MERGE_ER GS_TS_RETCODE_ER GS_STOP_ST GS_XSTALE_PRCLM_ER	See “Action 4 - Correct Group Services daemon problem” on page 291.
GS domain cannot be established or recovered.	The following errors may be present: GS_STARTERR_ER GS_DOM_NOT_FORM_WA	See “Action 5 - Correct domain problem” on page 291.
GS protocol has not been completed for a long time.	None	See “Action 6 - Correct protocol problem” on page 291.
Non-stale proclaim message received.	GS_XSTALE_PRCLM_ER	See “Action 7- Investigate non-stale proclaim message” on page 292.
HAGSGLSM cannot start.	GS_GLSM_STARTERR_ER	See “Action 8- Correct hagsglsm startup problem” on page 292.
HAGSGLSM has stopped.	GS_GLSM_ERROR_ER or None	See “Action 9 - hagsglsm daemon has stopped” on page 293.

Actions

Action 1 - Start Group Services daemon: Some of the possible causes are:

- Configuration-related problems that prevent the startup script from obtaining configuration data from the configuration resource manager.
- Operating system-related problems such as a shortage of space in the **/var** directory or a port number already in use.
- SRC-related problems that prevent the daemon from setting the appropriate SRC environment.

Run the diagnostics in “Operational test 2 - Determine why the Group Services subsystem is not active” on page 281 to determine the cause of the problem.

Action 2 - Verify Status of Group Services Subsystem: If the AIX error log, has an entry of **GS_DOM_MERGE_ER**, this indicates that the Group Services daemon has restarted. The most common cause of this situation is for the Group Services

daemon to receive a **NODE_UP** event from Topology Services after the Group Services daemon formed more than one domain.

If the Group Services daemon has been restarted and a domain has been formed, no action is needed. However, if the Group Services daemon is not restarted, perform “Operational test 1 - Verify that Group Services is working properly” on page 279 to verify the status of the GS subsystem.

Perform these steps:

1. Find a node with the **GS_DOM_MERGE_ER** in the AIX error log.
2. Find the **GS_START_ST** entry before the **GS_DOM_MERGE_ER** in the log.
3. If there is a **GS_START_ST** entry, issue the **lssrc** command:

   ```
   lssrc -l -s subsystem_name
   ```

   Where *subsystem_name* is **cthags**.
4. The **lssrc** output contains the node number that established the GS domain.
5. Otherwise, proceed to “Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered” on page 281.

After the merge, the Group Services daemon must be restarted. See **TS_NODEUP_ST** on page 216. Check it with “Operational test 2 - Determine why the Group Services subsystem is not active” on page 281.

Action 3 - Correct Group Services access problem: For the nodes that cannot join, some of the possible causes are:

1. Group Services may not be running.
2. Group Services domain may not be established.
3. The clients may not have permission to connect to the Group Services daemon.
4. Group Services is currently doing a protocol for the group that is trying to join or subscribe.

Analyze and correct this problem as follows:

1. Issue the **lssrc** command:

   ```
   lssrc -s cthags
   ```

   The output is similar to:

   ```
   Subsystem         Group            PID     Status
    cthags           cthags           23482   active
   ```

   If `Status` is not `active`, this indicates that the node cannot join the GS daemon. Perform “Operational test 2 - Determine why the Group Services subsystem is not active” on page 281. Start the Group Services subsystem by issuing this command:

   ```
   /usr/sbin/rsct/bin/cthagsctrl -s
   ```

   If `Status` is `active`, proceed to Step 2.
2. Perform “Operational test 1 - Verify that Group Services is working properly” on page 279 to check whether the Group Services domain is established or not.
3. Issue the command:

```
errpt -a -N subsystem_name | more
```

where *subsystem_name* is **cthags**, or, on HACMP nodes, **grpsvsc**.

Check the AIX error log for this entry:

```
Resource Name:   hags

-------------------------------------------------
LABEL:          GS_AUTH_DENIED_ST
IDENTIFIER:     23628CC2

Date/Time:       Tue Jul 13 13:29:52
Sequence Number: 213946
Machine Id:      000032124C00
Node Id:         c47n09
Class:           O
Type:            INFO
Description
User is not allowed to use Group Services daemon

Probable Causes
The user is not the root user
The user is not a member of hagsuser group

Failure Causes
Group Services does not allow the user

        Recommended Actions
        Check whether the user is the root
Check whether the user is a member of hagsuser group

Detail Data
DETECTING MODULE
RSCT,SSuppConnSocket.C,           1.17, 421
ERROR ID
.0ncMX.ESrWr.Oin//rXQ7.....................
REFERENCE CODE

DIAGNOSTIC EXPLANATION
User myuser1 is not a supplementary user of group 111. Connection refused.
```

This explains that the user of the client program does not have correct permission to use Group Services.

The following users can access Group Services:

- The **root** user.
- A user who is a primary or supplementary member of the **hagsuser** group, which is defined in the **/etc/group** file.

Change the ownership of the client program to a user who can access Group Services.

4. Issue the **hagsvote** command:

   ```
   hagsvote -ls cthags
   ```

   to determine whether the group is busy, and to find the Group Leader node for the specific group.
5. Issue the same command on the Group Leader Node to determine the global status of the group. Resolve the problem by the client programs.

Action 4 - Correct Group Services daemon problem: Some of the possible causes are:

1. Domain merged.
2. Group Services daemon received a non-stale proclaim message from its NS.

   If the Topology Services daemon is alive when the current NS restarts and tries to become a NS, the newly started NS sends a proclaim message to the other nodes. These nodes consider the newly started node as their NS. The receiver nodes consider the proclaim message current (that is, ″non-stale″) but undefined by design. Therefore, the received Group Services daemon will be core dumped.
3. The Topology Services daemon has died.
4. The Group Services daemon has stopped.
5. Group Services has an internal error that caused a core dump.

Examine the AIX error log by issuing the command:

```
errpt -J GS_DOM_MERGE_ER,GS_XSTALE_PRCLM_ER,GS_ERROR_ER,GS_STOP_ST,\
GS_TS_RETCODE_ER | more
```

and search for **GS**_ labels or a RESOURCE NAME of any of the GS subsystems. If an entry is found, the cause is explained in the DIAGNOSTIC EXPLANATION field.

If Group Services has taken a core dump, the core file is in: **/var/ct/***cluster_name***/run/cthags**. Save this file for error analysis.

Action 5 - Correct domain problem: Some of the possible causes are:

1. Topology Services is running, but the Group Services daemon is not running on some of the nodes.
2. Group Services internal NS protocol is currently running.

Proceed to “Operational test 3 - Determine why the Group Services domain is not established or why it is not recovered” on page 281.

Action 6 - Correct protocol problem: This is because the related client failed to vote for a specific protocol. Issue the **hagsvote** command on any node that has target groups:

```
hagsvote -ls cthags
```

If this node did not vote for the protocol, the output is similar to:

```
Number of groups: 1
Group slot #[3] Group name [theSourceGroup] GL node [0] voting data:
Not GL in phase [1] of n-phase protocol of type [Join].
Local voting data:
Number of providers: 1
Number of providers not yet voted: 1 (vote not submitted).
Given vote:[No vote value] Default vote:[No vote value]
ProviderId      Voted?  Failed? Conditional?
[101/11]        No      No      Yes
```

As the preceding text explains, one of local providers did not submit a vote. If this node has already voted but the overall protocol is still running, the output is similar to:

```
Number of groups: 1
Group slot #[3] Group name [theSourceGroup] GL node [0] voting data:
```

```
Not GL in phase [1] of n-phase protocol of type [Join].
Local voting data:
Number of providers: 1
Number of providers not yet voted: 0 (vote submitted).
Given vote:[Approve vote] Default vote:[No vote value]
ProviderId      Voted?  Failed? Conditional?
[101/11]        Yes     No      Yes
```

In this case, issue the same command on the Group Leader node. The output is similar to:

```
Number of groups: 1
Group slot #[2] Group name [theSourceGroup] GL node [0] voting data:
GL in phase [1] of n-phase protocol of type [Join].
Local voting data:
Number of providers: 1
Number of providers not yet voted: 1 (vote not submitted).
Given vote:[Approve vote] Default vote:[No vote value]
ProviderId      Voted?  Failed? Conditional?
[101/0] No      No      No

Global voting data:
Number of providers not yet voted: 1
Given vote:[Approve vote] Default vote:[No vote value]
Nodes that have voted: [11]
Nodes that have not voted: [0]
```

The GL's output contains the information about the nodes that did not vote. Investigate the reason for their failure to do so. Debug the GS client application.

Action 7- Investigate non-stale proclaim message: The local Group Services daemon receives a valid domain join request (proclaim) message from its NameServer (NS) more than once. This typically happens when Topology Services notifies Group Services of inconsistent node events. This problem should be resolved automatically if a **GS_START_ST** syslog entry is seen after the problem occurs.

Perform these actions:

1. Find the **GS_START_ST** AIX error log entry after this one.
2. If there is a **GS_START_ST** entry, issue the **lssrc** command:

   ```
   lssrc -l -s cthags
   ```
3. The **lssrc** output contains the node number that established the GS domain.
4. Otherwise, proceed to "Action 4 - Correct Group Services daemon problem" on page 291.

If this problem continues, contact the IBM Support Center (see "Information to collect before contacting the IBM Support Center" on page 276)

Action 8- Correct hagsglsm startup problem: Some of the possible causes are:

- AIX-related problems such as a shortage of space in the **/var** directory or a port number already in use.
- SRC-related problems that prevent the daemon from setting the appropriate SRC environment.

Proceed to "Operational test 7 - Verify the HAGSGLSM (Group Services GLobalized Switch Membership) subsystem" on page 285.

Action 9 - hagsglsm daemon has stopped: Issue this command:

```
lssrc -l -s cthagsglsm
```

If the daemon is stopped, the output will contain a status of ″inoperative″ for **hagsglsm**. Otherwise, the output will contain a status of ″active″ for **hagsglsm**. If stopping the daemon was not intended, see “Information to collect before contacting the IBM Support Center” on page 276 and contact the IBM Support Center.

Chapter 7. How to contact the IBM Support Center

IBM support is available for:

1. Customers without a SupportLine contract.
2. Customers with a SupportLine contract.

Service for non-SupportLine customers

If you do not have an IBM SupportLine service contract, please go to the on-line support at www.ibm.com/support/

Service for SupportLine customers

If you have an IBM SupportLine service contract, you may phone IBM at:

1. In the United States:
   The number for IBM software support is **1-800-237-5511**.
   The number for IBM hardware support is **1-800-IBM-SERV**.
2. Outside the United States, contact your local IBM Service Center.

Contact the IBM Support Center, for these problems:

- Node halt or crash not related to a hardware failure
- Node hang or response problems
- Failure in specific RSCT software subsystems
- Failure in other software supplied by IBM

You will be asked for the information you collected from “Information to collect before contacting the IBM Support Center” on page 276 and “Information to collect before contacting the IBM Support Center” on page 229.

You will be given a time period during which an IBM representative will return your call.

For failures in non-IBM software, follow the problem reporting procedures documented for that product.

For IBM hardware failures, contact IBM Hardware Support at the number above.

For any problems reported to the IBM Support Center, a Problem Management Record (PMR) is created. A PMR is an online software record used to keep track of software problems reported by customers.

- The IBM Support Center representative will create the PMR and give you its number.
- Have the information you collected available as it will need to be included in the PMR.
- Record the PMR number. You will need it to send data to the IBM Support Center. You will also need it on subsequent phone calls to the IBM Support Center to discuss this problem.

Be sure that the person you identified as your contact can be reached at the phone number you provided in the PMR.

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:

IBM Corporation
Department LRAS, Building 003
11400 Burnet Road
Austin, Texas 78758-3498
U.S.A.

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

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

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

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

COPYRIGHT LICENSE:

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

AIX
AIX 5L
IBM
IBM(logo)
IBMLink

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. The RS/6000 system, among others, runs the AIX 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.

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.

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.

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.

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.

node. Operating system image.

PAC. See *privileged attribute certificate.*

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.

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. Reliable Scalable Cluster Technology.

RSCT peer domain. See *peer domain.*

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.

Bibliography

This bibliography helps you find documentation related to Reliable Scalable Cluster Technology (RSCT).

Reliable Scalable Cluster Technology (RSCT) publications

- *RSCT for AIX 5L: Guide and Reference*, SA22-7889
- *RSCT for AIX 5L: Messages*, GA22-7891
- *RSCT for AIX 5L: Technical Reference*, SA22-7890
- *RSCT Group Services Programming Guide and Reference*, SA22-7888
- *RSCT Event Management Programming Guide and Reference*, SA22-7354
- *RSCT First Failure Data Capture Programming Guide and Reference*, SA22-7454

Finding RSCT documentation on the World Wide Web

You can download Portable Document Format (PDF) versions of the RSCT books from the IBM Publications Center Web site:

http://www.ibm.com/shop/publications/order/

The RSCT books are also available at:

http://www.ibm.com/servers/eserver/clusters/library

To view the RSCT PDF publications, you need access to the Adobe Acrobat Reader. The Acrobat Reader is shipped with the AIX Bonus Pack and is also freely available for downloading from the Adobe Web site at:

http://www.adobe.com

AIX publications

You can find links to the latest AIX publications on the IBM Web site at:

http://www.ibm.com/servers/aix/library/techpubs.html

Cluster Systems Management (CSM) publications

- *CSM for AIX 5L Administration Guide*, SA22-7918
- *CSM for AIX 5L Hardware Planning and Control Guide*, SA22-7920
- *CSM for AIX 5L Software Planning and Installation Guide*, SA22-7853

Red books

IBM's International Technical Support Organization (ITSO) has published a number of red books related to RSCT. For a current list, see the IBM Web site at:

http://www.ibm.com/redbooks

Non-IBM publications

Here are some non-IBM publications that you might find helpful:

- Almasi, G. and A. Gottlieb. *Highly Parallel Computing*, Benjamin-Cummings Publishing Company, Inc., 1989.
- Foster, I. *Designing and Building Parallel Programs*, Addison-Wesley, 1995.
- Pfister, Gregory, F. *In Search of Clusters*, Prentice Hall, 1998.

Index

Special characters

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

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

IBM Reliable Scalable Cluster Technology for AIX 5L
RSCT Guide and Reference

Publication No. SA22-7889-01

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

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

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

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

Please tell us how we can improve this book:

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

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

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SA22-7889-01

Cut or Fold
Along Line

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

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

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

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

SA22-7889-01

Cut or Fold
Along Line

Program Number: 5765-E61, 5765-E62

SA22-7889-01