

# PRIMECLUSTER™

Concepts Guide 4.3 (Oracle Solaris, Linux)

## **Copyright and Trademarks**

Linux is a trademark or registered trademark of Mr. Linus Torvalds in the United States and other countries.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions.

EMC is a registered trademark of EMC Corporation in the United States or other countries.

PRIMECLUSTER is a registered trademark of Fujitsu Limited.

All other hardware and software names used are trademarks of their respective companies.

## **Export Controls**

Exportation/release of this document may require necessary procedures in accordance with the regulations of the Foreign Exchange and Foreign Trade Control Law of Japan and/or US export control laws.

### Requests

No part of this documentation may be reproduced or copied without permission of FUJITSU LIMITED.

The contents of this documentation may be revised without prior notice.

All Rights Reserved, Copyright (C) FUJITSU LIMITED 2012-2014.

---

Preface

---

Clustering technology overview

---

PRIMECLUSTER architecture

---

Cluster interconnect details

---

Reliant Monitor Services (RMS)

---

RMS configuration tools

---

SIS

---

Glossary

---

Abbreviations

---

Figures

**Continued** ►

---

Index



---

# Contents

<b>1</b>	<b>Preface</b>	<b>1</b>
1.1	About this manual	1
1.2	Related documentation	2
1.3	Conventions	3
1.3.1	Notation	3
1.3.1.1	Prompts	3
1.3.1.2	Manual page section numbers	4
1.3.1.3	The keyboard	4
1.3.1.4	Typefaces	4
1.3.1.5	Example 1	5
1.3.1.6	Example 2	5
1.3.2	Command syntax	5
1.4	Important notes and cautions	6
1.5	Abbreviations	7
1.6	Revision history	7
<b>2</b>	<b>Clustering technology overview</b>	<b>9</b>
2.1	Introduction	9
2.2	High availability	11
2.2.1	Cluster interconnects	12
2.2.2	HA manager (RMS)	12
2.2.2.1	Protecting data integrity	12
2.2.2.2	RMS configuration tools	19
2.3	Scalability	19
2.4	Single-node cluster	20
2.5	Virtualization support	20
<b>3</b>	<b>PRIMECLUSTER architecture</b>	<b>25</b>
3.1	Architectural overview	25
3.2	PRIMECLUSTER key design features	28
3.2.1	Modularity	28
3.2.2	Platform independence	28
3.2.3	Scalability	29
3.2.4	Availability	29
3.2.5	Guaranteed data integrity	29
3.3	PRIMECLUSTER components	30
3.3.1	CF	31
3.3.1.1	OSD	31
3.3.1.2	ICF	31
3.3.1.3	JOIN	32
3.3.1.4	ENS	32

# Contents

---

3.3.1.5	Cluster Admin . . . . .	33
3.3.1.6	Web-Based Admin View . . . . .	33
3.3.1.7	PRIMECLUSTER SF . . . . .	34
3.3.2	RMS . . . . .	44
3.3.2.1	RMS configuration tools . . . . .	45
3.3.3	SIS . . . . .	46
3.3.4	PAS . . . . .	47
3.3.5	GDS . . . . .	48
3.3.6	GFS . . . . .	51
3.3.6.1	GFS shared file system . . . . .	51
3.3.6.2	Benefits . . . . .	54
3.3.7	GLS . . . . .	54
<b>4</b>	<b>Cluster interconnect details . . . . .</b>	<b>57</b>
4.1	Overview . . . . .	57
4.1.1	A cluster interconnect is different from a network . . . . .	57
4.1.2	Network availability . . . . .	58
4.1.3	Interconnect protocol . . . . .	59
4.2	Cluster interconnect requirements . . . . .	59
4.2.1	Redundancy . . . . .	60
4.2.2	Routes . . . . .	61
4.2.2.1	Heartbeats . . . . .	61
4.2.3	Properties . . . . .	63
4.2.3.1	Bandwidth . . . . .	63
4.2.3.2	Latency . . . . .	65
4.2.3.3	Reliability . . . . .	66
4.2.3.4	Device interface . . . . .	66
4.2.3.5	Security . . . . .	67
<b>5</b>	<b>Reliant Monitor Services (RMS) . . . . .</b>	<b>69</b>
5.1	RMS overview . . . . .	69
5.1.1	Redundancy . . . . .	70
5.1.2	Application switchover . . . . .	71
5.1.2.1	Automatic switchover . . . . .	72
5.1.2.2	Manual switchover . . . . .	72
5.1.2.3	IP aliasing . . . . .	72
5.1.2.4	Data integrity . . . . .	72
5.2	RMS monitoring and switchover . . . . .	73
5.2.1	Base monitor . . . . .	73
5.2.2	Configuration file . . . . .	74
5.2.2.1	Interdependencies . . . . .	74
5.2.2.2	Object types . . . . .	75
5.2.2.3	Object definitions . . . . .	75
5.2.3	Scripts . . . . .	76

5.2.4	Detectors . . . . .	.77
5.2.5	RMS environment variables . . . . .	.78
5.3	RMS administration . . . . .	.79
5.4	Customization options . . . . .	.79
5.4.1	Generic types and detectors . . . . .	.79
<b>6</b>	<b>RMS configuration tools . . . . .</b>	<b>.81</b>
6.1	RMS configuration overview . . . . .	.81
6.2	RMS Wizard Tools . . . . .	.82
6.3	RMS Wizard Kit . . . . .	.83
<b>7</b>	<b>SIS . . . . .</b>	<b>.85</b>
7.1	SIS overview . . . . .	.85
7.1.1	Features . . . . .	.86
7.1.2	Service nodes . . . . .	.87
7.1.3	Gateway nodes . . . . .	.87
7.1.4	Primary database node . . . . .	.87
7.1.5	Backup database nodes . . . . .	.87
7.1.6	Satellite nodes . . . . .	.88
7.2	SIS design . . . . .	.89
7.3	VIP load-balancing algorithms . . . . .	.90
7.4	Proxy addresses . . . . .	.90
7.5	Private addresses . . . . .	.91
7.6	Failover . . . . .	.91
<b>8</b>	<b>Appendix-Release information . . . . .</b>	<b>.93</b>
	<b>Glossary . . . . .</b>	<b>.95</b>
	<b>Abbreviations . . . . .</b>	<b>.111</b>
	<b>Figures . . . . .</b>	<b>.115</b>
	<b>Index . . . . .</b>	<b>.117</b>

# Contents


---



---

# 1 Preface

This manual is a conceptual overview of the PRIMECLUSTER suite of products. PRIMECLUSTER is a fourth-generation clustering solution that provides high availability and scalability, independent of the operating system and hardware platform. Its modular software architecture consists of a basic set of modules deployed on all computers (nodes) in a cluster, plus optional modules that support specific types of applications. The modular architecture allows flexible clustering solutions for a wide range of customers—solutions that can be adapted to virtually any current or future platform.

 This guide describes all the components of PRIMECLUSTER. All of these components might not be available for all releases. The *Software Release Guide PRIMECLUSTER* and the *PRIMECLUSTER Installation Guide* on the CD should be checked to verify which features are available for a specific platform.

This manual is intended for end users, system administrators, and support personnel. The purpose of this manual is to provide conceptual information only. It is not designed as a guide for administration, configuration, or installation (for more information, refer to the section “Related documentation”).

## 1.1 About this manual

This manual is organized as follows:

- The chapter “Clustering technology overview” describes the concepts and benefits of clustering, including the main components of PRIMECLUSTER.
- The chapter “PRIMECLUSTER architecture” explains the PRIMECLUSTER architecture and discusses the key features that PRIMECLUSTER provides.
- The chapter “Cluster interconnect details” discusses the concepts, requirements, and design considerations of the cluster interconnect.
- The chapter “Reliant Monitor Services (RMS)” introduces the basic concepts, components, and benefits of RMS.
- The chapter “RMS configuration tools” explains the concepts of the RMS Wizard Tools and the RMS Wizard Kit.
- The chapter “SIS” introduces the basic concepts and components of the Scalable Internet Services (SIS).

- The chapter “Appendix-Release information” lists changes in this manual.

## 1.2 Related documentation

The documents listed below provide details about related PRIMECLUSTER products. Please contact your sales representative for ordering information.

- *Software Release Guide (Oracle Solaris) and Installation Guide (Oracle Solaris)*—Provides instructions for installing and upgrading PRIMECLUSTER products.
- *Software Release Guide (Linux) and Installation Guide (Linux)*—Provides instructions for installing and upgrading PRIMECLUSTER products.
- *Web-Based Admin View (Oracle Solaris) Operation Guide*—Provides information on using the Web-Based Admin View management GUI.
- *Web-Based Admin View (Linux) Operation Guide*—Provides information on using the Web-Based Admin View management GUI.
- *Cluster Foundation (CF) (Oracle Solaris) Configuration and Administration Guide*—Provides instructions for configuring and administering the PRIMECLUSTER Cluster Foundation.
- *Cluster Foundation (CF) Configuration and Administration Guide (Linux)*—Provides instructions for configuring and administering the PRIMECLUSTER Cluster Foundation.
- *Reliant Monitor Services (RMS) with Wizard Tools (Oracle Solaris, Linux) Configuration and Administration Guide*—Provides instructions for configuring and administering PRIMECLUSTER Reliant Monitor Services using the Wizard Tools interface.
- *Reliant Monitor Services (RMS) (Oracle Solaris, Linux) Troubleshooting Guide*—Describes diagnostic procedures to solve RMS configuration problems, including how to view and interpret RMS log files. Provides a list of all RMS error messages with a probable cause and suggested action for each condition.
- *Scalable Internet Services (SIS) (Oracle Solaris, Linux) Configuration and Administration Guide*—Provides information on configuring and administering Scalable Internet Services (SIS).

- *Global Disk Services (Oracle Solaris) Configuration and Administration Guide*—Provides information on configuring and administering Global Disk Services (GDS).
- *Global File Services (Oracle Solaris) Configuration and Administration Guide*—Provides information on configuring and administering Global File Services (GFS).
- *Global Link Services (Oracle Solaris) Configuration and Administration Guide: Redundant Line Control Function*—Provides information on configuring and administering the redundant line control function for Global Link Services (GLS).
- *Global Link Services (Oracle Solaris) Configuration and Administration Guide: Multipath Function*—Provides information on configuring and administering the multipath function for Global Link Services (GLS).
- *Software Release Guide PRIMECLUSTER and PRIMECLUSTER Installation Guide* for all products—These documentation files are included as HTML files on the PRIMECLUSTER Framework CD. Release notices provide late-breaking information about installation, configuration, and operations for PRIMECLUSTER. Read this information first.

## 1.3 Conventions

To standardize the presentation of material, this manual uses a number of notational, typographical, and syntactical conventions.

### 1.3.1 Notation

This manual uses the following notational conventions.

#### 1.3.1.1 Prompts

Command line examples that require system administrator (or root) rights to execute are preceded by the system administrator prompt, the hash sign (#). Entries that do not require system administrator rights are preceded by a dollar sign (\$).

In some examples, the notation *node#* indicates a root prompt on the specified node. For example, a command preceded by *fuji2#* would mean that the command was run as user *root* on the node named *fuji2*.

### 1.3.1.2 Manual page section numbers

References to operating system commands are followed by their manual page section numbers in parentheses—for example, *cp(1)*.

### 1.3.1.3 The keyboard

Keystrokes that represent nonprintable characters are displayed as key icons such as **Enter** or **F1**. For example, **Enter** means press the key labeled *Enter*; **Ctrl-B** means hold down the key labeled *Ctrl* or *Control* and then press the **B** key.

### 1.3.1.4 Typefaces

The following typefaces highlight specific elements in this manual.

Typeface	Usage
Constant Width	Computer output and program listings; commands, file names, manual page names and other literal programming elements in the main body of text.
<i>Italic</i>	Variables in a command line that you must replace with an actual value. May be enclosed in angle brackets to emphasize the difference from adjacent text, e.g., <i>&lt;nodename&gt;RMS</i> ; unless directed otherwise, you should not enter the angle brackets.  The name of an item in a character-based or graphical user interface. This may refer to a menu item, a radio button, a checkbox, a text input box, a panel, or a window title.
<b>Bo1d</b>	Items in a command line that you must type exactly as shown.

Typeface conventions are shown in the following examples.

### 1.3.1.5 Example 1

Several entries from an `/etc/passwd` file are shown below:

```
root:x:0:1:0000-Admin(0000):/:/sbin/ksh
sysadm:x:0:0:System Admin.:/usr/admin:/usr/sbin/sysadm
setup:x:0:0:System Setup:/usr/admin:/usr/sbin/setup
daemon:x:1:1:0000-Admin(0000):/:
```

### 1.3.1.6 Example 2

To use the `cat(1)` command to display the contents of a file, enter the following command line:

```
$ cat file
```

## 1.3.2 Command syntax

The command syntax observes the following conventions.

Symbol	Name	Meaning
[ ]	Brackets	Enclose an optional item.
{ }	Braces	Enclose two or more items of which only one is used. The items are separated from each other by a vertical bar ( ).
	Vertical bar	When enclosed in braces, it separates items of which only one is used. When not enclosed in braces, it is a literal element indicating that the output of one program is piped to the input of another.
( )	Parentheses	Enclose items that must be grouped together when repeated.
...	Ellipsis	Signifies an item that may be repeated. If a group of items can be repeated, the group is enclosed in parentheses.

## 1.4 Important notes and cautions



### **Important**

Indicates important information.



### **Caution**

Indicates a situation that can cause harm to data.



### **Note**

Indicates information that needs special attention.

## 1.5 Abbreviations

Oracle Solaris might be described as Solaris, Solaris Operating System, or Solaris OS.

If "Solaris X" is indicated in the reference manual name of the Oracle Solaris manual, replace "Solaris X" with "Oracle Solaris 10 (Solaris 10)" or "Oracle Solaris 11 (Solaris 11)."

## 1.6 Revision history

Revision	Location	Manual code
Added the descriptions of supported virtualization environments.	2.1	J2UZ-5303-04ENZ0(01)
Added the descriptions of a single-node cluster.	2.4	
Changed the term "Oracle Solaris Containers environment" to "Oracle Solaris Zones environment."	2.5	
Added the descriptions to ICMP(SA_icmp).	3.3.1.7	
Added the descriptions when making cluster interconnects redundant.	4.2 4.2.1	





---

## 2 Clustering technology overview

This chapter introduces the basic concepts and benefits of clustering, including the main components of PRIMECLUSTER.

This chapter discusses the following:

- The section “Introduction” introduces the concepts of clustering technology.
- The section “High availability” describes the concept of high availability (HA) and specifies how PRIMECLUSTER makes applications highly available.
- The section “Scalability” describes the scalable benefits of PRIMECLUSTER.
- The section “Single-node cluster” describes the operation of a cluster system with one node.
- The section “Virtualization support” describes the support for virtualization in PRIMECLUSTER.

### 2.1 Introduction

Clustering is imprecisely defined in distributed computing. In general, a cluster is a combination of computers or partitions of a computer (nodes) that act cooperatively to provide one or more of the following:

- High availability (HA)—Increasing service availability by using redundant components
- Scalability—Supplied by replicating application resources

This document focuses on cluster servers that provide HA and scalability as provided by the PRIMECLUSTER software suite. It does not discuss other kinds of clustering such as administrative clusters or scientific computing clusters.

Figure 1 illustrates a typical two-node cluster.

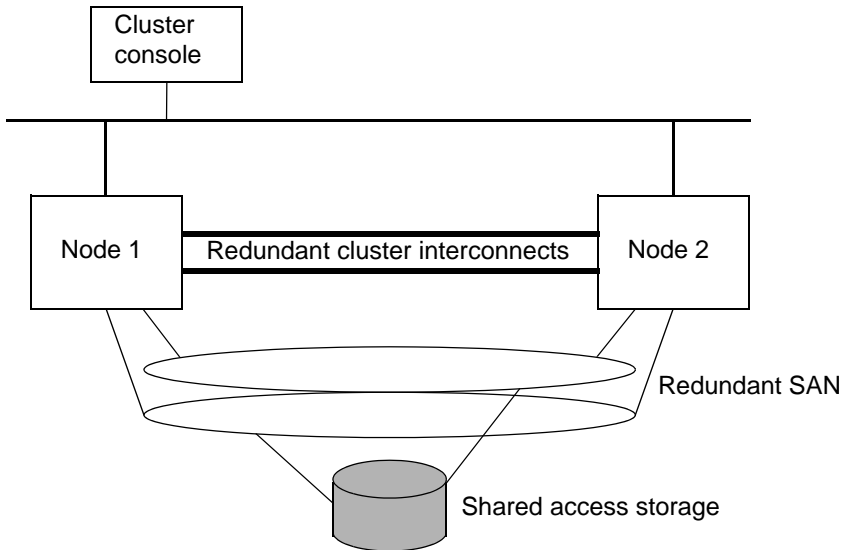


Figure 1: Typical two-node cluster

PRIMECLUSTER also supports the single-node cluster configuration with one node. Application status is monitored in a single-node cluster. If an error is detected, availability can be improved by automatically restarting the application and trying recovery.

Furthermore, PRIMECLUSTER supports the following virtualization environments:

- KVM environment
- Xen environment
- VMware environment
- Oracle VM Server for SPARC environment
- Oracle Solaris Zones environment

## 2.2 High availability

HA clusters use redundant components to compensate for a failure. Most HA clusters have a shared-storage environment; therefore, PRIMECLUSTER assumes, but does not require, that the cluster nodes connect to storage through a shared-access Storage Area Network (SAN). To provide redundancy, access to the SAN should include multiple host adaptors in each node. Additionally, each node in the cluster must know if the other nodes in the cluster are operational. They do this by sending heartbeats over the interconnects.

## 2.2.1 Cluster interconnects

The cluster interconnect is the basic building block of a cluster. The following rules apply to the cluster interconnects:

- A node must be connected to a cluster interconnect to participate in a cluster.
- Multiple cluster interconnects provide redundancy, so that a failure in a component of one interconnect does not cause complete loss of contact within the cluster.

In addition to heartbeat requests, the cluster interconnects carry messages between nodes such as notification of events, communications between processes, and cluster file access. Additional details are discussed in the Chapter “Cluster interconnect details” later in this document.

## 2.2.2 HA manager (RMS)

Heartbeats between nodes help determine if a node is functional and if the node is able to communicate with other nodes in the cluster; however, heartbeats cannot determine if applications are running or if host adapters for the SAN are functional. PRIMECLUSTER provides Monitoring Agents (MAs), called detectors, that monitor the state of components for applications and the resources used by those applications. If a resource or application fails, the event is noted by the detector and reported to the HA manager, Reliant Monitor Services (RMS).

### 2.2.2.1 Protecting data integrity

RMS protects data integrity by performing the following tasks:

- Monitoring applications and their resources
- Ensuring that one application is only allowed to run on one node at a time (except parallel applications like Oracle RAC)
- Starting applications automatically only when all cluster nodes are in a known state (except when otherwise controlled due to settings of the `HV_AUTOSTARTUP_IGNORE` or `PARTIAL_CLUSTER` environment variables)

## Monitoring applications

RMS is configured with rules specific to the applications and the configuration of the cluster. When a detector reports a failure, RMS takes the appropriate actions to recover the resources that are needed to provide continued availability of the application. The recovery actions are defined for every application and resource.

RMS recovery actions are as follows:

- Local recovery—The application's resources are recovered, and the application is restarted on the same cluster node.
- Remote recovery—The application's resources are recovered, and the application is restarted on another cluster node.

## Cluster partition

A cluster partition is the result of multiple failures in the cluster interconnects such that some or all of the nodes continue to operate, but subsets of nodes cannot communicate with each other (this is sometimes called *split-brain syndrome*). Redundant cluster interconnects prevent partitions from occurring when only a single component has failed.

Figure 2 shows an example of two breaks in the cluster interconnects in which Node 1 and Node 2 can no longer communicate with each other. However, both nodes still have access to the SAN. Therefore, if recovery actions were taken independently on each node, two instances of an application could be running unaware on the two nodes of the cluster. If these instances were to make uncoordinated updates to their data, then data corruption would occur. Clearly, this condition cannot be allowed.

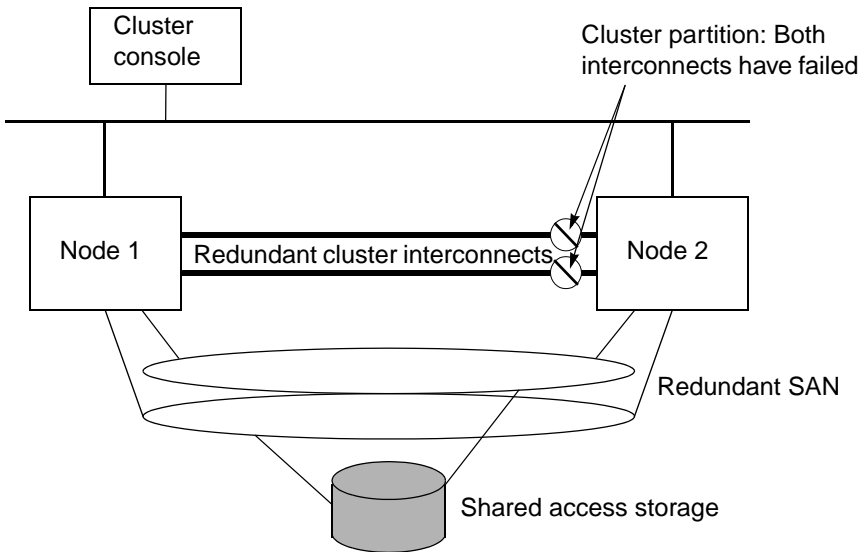


Figure 2: Cluster partition in two-node cluster

When a heartbeat failure occurs, each node sets the nodes from which they do not receive a heartbeat into a state called `LEFTCLUSTER`. The `LEFTCLUSTER` state is a middle state between `UP` and `DOWN` and indicates that the node is no longer part of the cluster, but its actual status is unknown. Before the nodes can take any recovery action, they must ensure that the cluster has a consistent state. The consistent state is sometimes called a quorum.

The terms consistent state and quorum are used interchangeably in `PRIME-CLUSTER` documents. The cluster is in a consistent state when every node of the cluster is in a known state (`UP` or `DOWN`) and each node that is `UP` can communicate with every other node that is `UP`. The applications in the cluster should ensure that the cluster is in a consistent state before starting any operations that will alter shared data. For example, `RMS` ensures that the cluster is in a consistent state before it will start any applications in the cluster.

`PRIMECLUSTER` performs an elimination through a variety of methods, depending on the architecture of the nodes. When `PRIMECLUSTER` determines that a node is entering the `LEFTCLUSTER` state, it eliminates that node; thereby, recovering the application and guaranteeing data integrity.

**i** The term *quorum* has been used in various ways in the literature to describe the handling of cluster partitions. Usually, this implies that when  $(n + 1)/2$  nodes can see each other, they have a quorum and the other nodes will not perform any I/O operations. Because PRIMECLUSTER methods differ from the conventional meaning of quorum, the term *cluster integrity* was adopted. Some PRIMECLUSTER commands use the *quorum* term heritage, but in these cases *cluster integrity* is implied.

### Cluster Integrity Monitor

The purpose of the Cluster Integrity Monitor (CIM) is to allow applications to determine when it is safe to perform operations on shared resources. It is safe to perform operations on shared resources when a node is a member of a cluster that is in a consistent state.

A consistent state is when all the nodes of a cluster that are members of the CIM set are in a known and safe state. The nodes that are members of the CIM set are specified in the CIM configuration. Only these nodes are considered when the CIM determines the state of the cluster.

When a node first joins or forms a cluster, the CIM indicates that the cluster is consistent only if it can determine the following:

- The status of the other nodes that make up the CIM set
- Whether the nodes of the CIM set are in a safe state

The method used to determine the state of the members of the cluster is sometimes called the CIM method. The CIM can use several different CIM methods; however, the following are available by default and are discussed here:

- NSM—The Node State Monitor (NSM) monitors the node states at fixed cycles, and it tracks the state of the nodes that are currently, or have been, members of the cluster. This is also known as the NULL or default CIM method. NSM is an integrated part of the PRIMECLUSTER CF.
- RCI—The RCI (Remote Cabinet Interface) is a special SPARC Enterprise M series environmental control and state network that can asynchronously both report on the state of the systems and control the systems on Solaris systems. (For more information, refer to the PRIMECLUSTER *Cluster Foundation (CF) Configuration and Administration Guide (Solaris).*)

- **MMB**—The MMB (Management Board) is a special PRIMEQUEST environmental control and state network that can asynchronously both report the status of the systems and control the systems on Linux systems. (For more information, refer to the PRIMECLUSTER *Cluster Foundation (CF) Configuration and Administration Guide (Linux)*.)

PRIMECLUSTER allows you to register and use multiple CIM methods. When multiple CIM methods are registered, CIM uses the lower priority method to check the state of a node only if the higher priority method cannot determine the node state. For example, if RCI and NSM are registered as CIM methods and RCI has the higher priority, then CIM uses the CIM method that uses RCI to check the node status.

If the target is a node or a partition, the RCI CIM method returns UP or DOWN, and then processing ends. However, if the node being checked by the RCI method is not connected to the RCI or if the RCI is not operating properly, then the RCI method will fail. CIM then uses the NSM-based CIM method to check the node state.

Similarly if MMB and NSM are registered as CIM methods and MMB has the higher priority, then CIM uses the CIM method that uses MMB to check the node status. In this case, if the target is a PRIMEQUEST node, the MMB CIM method returns UP or DOWN, and then processing ends. However, if the node being checked by the MMB method is not connected to the MMB or if the MMB is not operating properly, then the MMB method will fail. CIM then uses the NSM-based CIM method to check the node state.

The CIM reports on whether a node state in a cluster is consistent (true), or a node state is not consistent (false) for the cluster. True and false are defined as follows:

- **TRUE**—A known state for all CIM nodes
- **FALSE**—An unknown state for any cluster CIM node



## Shutdown Facility


The CIM allows applications to determine when a cluster is in a consistent state, but it does not take action to resolve inconsistent clusters. Many different methods to ensure consistent clusters have been used in the high-availability field, but there is only one method that has proven completely effective and does not require cooperation between the nodes. PRIMECLUSTER uses this method known as the Shutdown Facility (SF) to return to a consistent cluster state when something occurs to disrupt that state. In the cluster partition example shown in Figure 2, both nodes will report the other node as having the state LEFTCLUSTER. The CIM will return a FALSE status. To get the cluster into a consistent state, SF forces one of the nodes into a safe state by either forcing a panic or shutting off the power.

The SF can be configured to eliminate nodes through a variety of methods. When the SF receives a request to eliminate a node, it tries to shut down the node by using the methods in the order that were specified. Once a method has successfully eliminated the node, the node's state is changed to DOWN by the SF.

The transition from LEFTCLUSTER to DOWN is the signal used by the various cluster services to start recovery actions. In the two-node cluster configuration in Figure 2, the first elimination method could be to panic the node through the cluster console. If this were to fail, the next method might be to use a method to turn off the power to the node. Note that different systems will support different shutdown methods. For example, the cluster console is available for Solaris, but is not available for Linux.

If all of the configured SF methods fail to return a positive acknowledgement that the requested node has been eliminated, then no further action is taken. This leaves the cluster in an inconsistent state and requires operator intervention to proceed.

This fail-safe approach ensures that damage to user data could not occur by inadvertently allowing an application to run in two parts of a partitioned cluster. This also protects from the situation where a node fails to respond to heartbeats (for example, an extreme system load) and then comes back to life later. In this case, the application on the node that returns to life may continue to run even though the other node has taken action to start that application.

 PRIMECLUSTER allows you to use various hardware-specific methods to set definition that reset nodes on which Solaris or Linux operate. For example, PRIMECLUSTER allows you to forcibly stop a node by using the RCI, sending a BREAK signal to the console unit, using the IPMI, using the Blade, or using the MMB.

## Monitoring Agents (MA)

Many hardware platforms provide an environmental monitoring and control network that can report on the state of a system. For example, the SPARC Enterprise M series hardware provides the Remote Cabinet Interface (RCI) and the PRIMEQUEST hardware provides the Management Board (MMB), which were discussed previously in the Shutdown Facility as means to shutdown a system. The RCI and the MMB can also be used to monitor the state of the system and quickly detect when a node has panicked or been shutdown. Other hardware platforms use different interfaces but the functionality is similar.

PRIMECLUSTER provides a mechanism where these hardware monitors can be used to quickly detect a system state change and inform the cluster membership functions. Without this monitoring capability, only the cluster heartbeat timeout will detect that a node has panicked; this will take up to 10 seconds with the default heartbeat interval. When a Monitoring Agent (MA) is used, it can detect a node panic very quickly. For example, with PRIMEPOWER hardware and the RCI, the MA takes less than 1 second to detect a system panic. MAs are implemented as plug-ins that interfaces with the Shutdown Facility.

The MA technology allows PRIMECLUSTER to recover from monitored node failures very quickly. For non-cluster aware applications the time from when a node panic occurs to the time that the application recovery begins can be as short as 2.5 seconds under optimal conditions. The time the application takes to start up and become ready to operate varies from one application to another. For cluster-aware applications, such as Oracle RAC, the time from a system panic to the time Oracle has started recovery and is processing queries on the surviving nodes can be as short as 6.5 seconds. At this point, Oracle may still be performing some recovery actions that might impact performance, but it is able to respond to user queries.

If a node fails, PRIMECLUSTER does the following:

- Detects a node failure
- Notifies of the failure
- Confirms the node state
- Eliminates the node

The MA notifies SF of a node failure on detecting it. SF seeks a redundant confirmation regarding the node state to assess the reliability of the failure notification. This verification procedure is required to prevent the node that is normally running from being shut down.

SF confirms the node state as follows:

- Collects the node state information from all registered MAs again.
- Checks if the response to the CF heartbeat request is returned.

SF prompts the MA to eliminate the failed node when all the MAs notify SF of the node failure, and CF notifies SF of the failure in responding to the heartbeat request. When the node elimination is done, this brings the other node **DOWN**.

### 2.2.2.2 RMS configuration tools

To properly recover an application, RMS must know about the resources an application requires for proper operation. The configuration of the resources and the relationship between the resources can be very complex. RMS configuration tools assist in obtaining this information for RMS, and simplify the configuration process.

The configuration tool foundations (RMS Wizard Tools) capture generic information about the cluster and common application services. The RMS Wizard Kit is an expert system for configuring specific, complex applications such as SAP R/3 or Oracle Parallel Server (OPS) when they are to be monitored by RMS.



For information on the availability of RMS Wizard Kit, contact your local customer support service.

## 2.3 Scalability

Scalability is another benefit of PRIMECLUSTER. Scalability is provided by the cluster's ability to grow in computing capacity when demand increases. There are two basic types of applications relative to scalability. These types of applications can be divided as follows:

- Applications that interact closely with the cluster software and are designed for a distributed environment
- Applications that are not aware of the cluster

### **Applications that interact with cluster software**

An example of a scalable application that interacts with cluster software is Oracle 9iRAC. Oracle 9iRAC starts an instance of the database server on some or all the nodes of the cluster. In addition, Oracle 9iRAC interacts with the cluster software to send messages between the instances on different nodes and must know about the state of the nodes in the cluster.

### **Applications that are not aware of the cluster**

Applications that do not have special provisions for distributing their work in the cluster can be replicated on different nodes. If the application is designed so that different users can run separate copies of the application on a single computer, then this effect can be extended by using Scalable Internet Services (SIS) (refer to the Chapter “SIS”) to distribute the workload to the nodes of the cluster and Global File Services (GFS) to allow an application to run anywhere in the cluster (refer to the section “PRIMECLUSTER components” and the manuals for PRIMECLUSTER GFS for more details).

## **2.4 Single-node cluster**

Single-node cluster is a cluster system with one node, and it is possible to monitor and control services on the node.

If an error is detected, availability can be improved by automatically restarting the application and trying recovery.

You can also use this mode as a development environment for creating and testing cluster applications.

However, services will be stopped if a hardware error occurs. Moreover, no failover occurs in the single-node cluster operation.

## **2.5 Virtualization support**

PRIMECLUSTER allows a redundant configuration in a virtualization environment and also enables high reliability of the integrated system.

- KVM environment
- Xen environment
- VMware environment

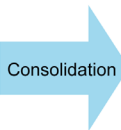
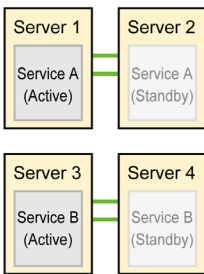
- Oracle VM Server for SPARC environment
- Oracle Solaris Zones environment

Previously, the cluster system has been built and operated with different servers for each service.

However, it has become possible to integrate multiple services by supporting virtual environments.

Within a server, CPU resources can be allocated and operated for each service.

■ Existing cluster system



■ Cluster system in a virtual environment (For example, KVM environment)

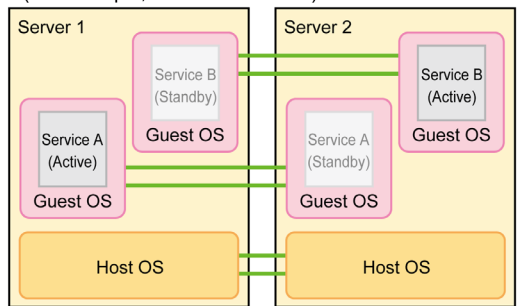


Figure 3: Existing cluster system and cluster system in a virtual environment (For example, KVM environment)

**KVM/Xen environment**

By installing PRIMECLUSTER into each host OS and guest OS, you can immediately switch servers in the event of an operating system error as well as an application error.

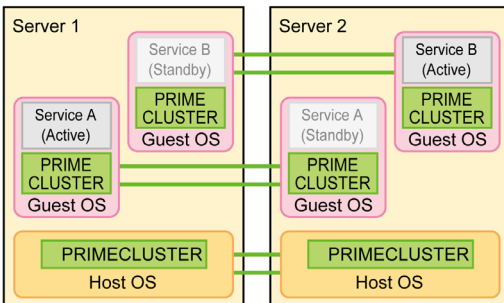


Figure 4: Cluster system in a virtual environment (For example, KVM/Xen environment)

**VMware environment**

Install PRIMECLUSTER into guest OSes only. When an error of applications or operating systems occurs, you can safely and securely switch servers with I/O fencing using shared disks.

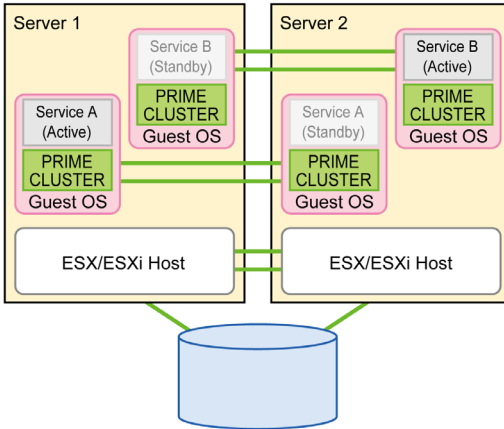


Figure 5: Cluster system in a virtual environment (For example, VMware environment)

**Oracle VM Server for SPARC environment**

By installing PRIMECLUSTER into control domains, you can switch each server when an error occurs in control domains or guest domains. Applications are not monitored.

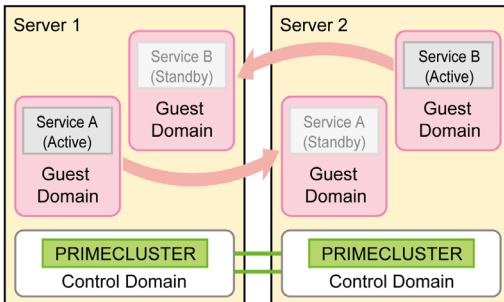


Figure 6: Cluster system in a virtual environment (For example, Oracle VM Server for SPARC environment)

### Oracle Solaris Zones environment

By installing PRIMECLUSTER into the global zone, you can switch each server when an error occurs in the global zone.

Moreover, by installing PRIMECLUSTER into non-global zones, you can switch each non-global zone when an error occurs in applications.

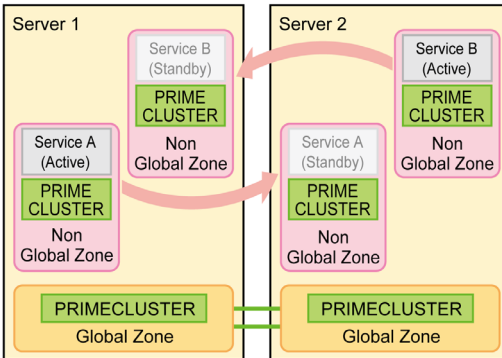


Figure 7: Cluster system in a virtual environment (For example, Oracle Solaris Zones environment)





---

## 3 PRIMECLUSTER architecture

This chapter explains the PRIMECLUSTER architecture and discusses the key features that PRIMECLUSTER provides.

This chapter discusses the following:

- The section “Architectural overview” introduces the basic PRIMECLUSTER components and how the components function within the clustering environment.
- The section “PRIMECLUSTER key design features” details the key design features of PRIMECLUSTER, including operating system and platform independence, scalability, availability, and guaranteed data integrity.
- The section “PRIMECLUSTER components” describes the Cluster Foundation (CF), Reliant Monitor Services (RMS), Web-Based Admin View graphical user interface (GUI), and optional PRIMECLUSTER services.

### 3.1 Architectural overview

The PRIMECLUSTER design is based on a long history of clustering and high availability (HA) software and hardware experience. Figure 8 is a conceptual model of a typical cluster design that illustrates PRIMECLUSTER's position as a middleware solution. Features of the PRIMECLUSTER solution include:

- PRIMECLUSTER is easily ported to new hardware platforms, operating systems, and cluster interconnects.
- PRIMECLUSTER provides services only for the management or use of the cluster.
- PRIMECLUSTER supplies interfaces so that other applications, such as enterprise management software, can interact with or call on services provided by PRIMECLUSTER.

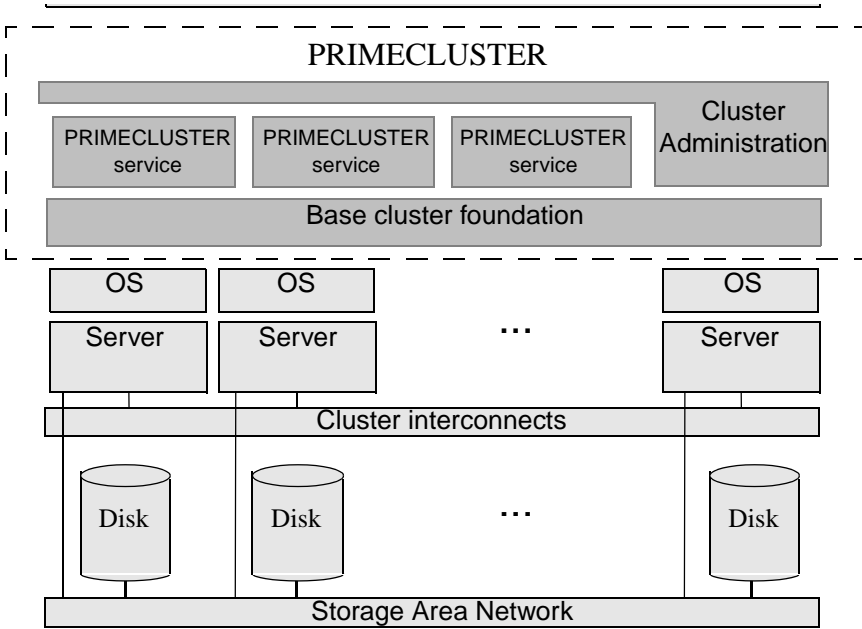


Figure 8: Diagram of a typical PRIMECLUSTER setup

Figure 9 shows a conceptual overview of the PRIMECLUSTER software architecture and how it interfaces with a server's native operating system. All of the PRIMECLUSTER software modules use operating-system-independent interfaces with an operating-system-dependant (OSD) layer to communicate between themselves and to access the base operating system services. Some examples of the operations that the OSD provides are as follows:

- Memory allocations
- Synchronizations
- Device and network access

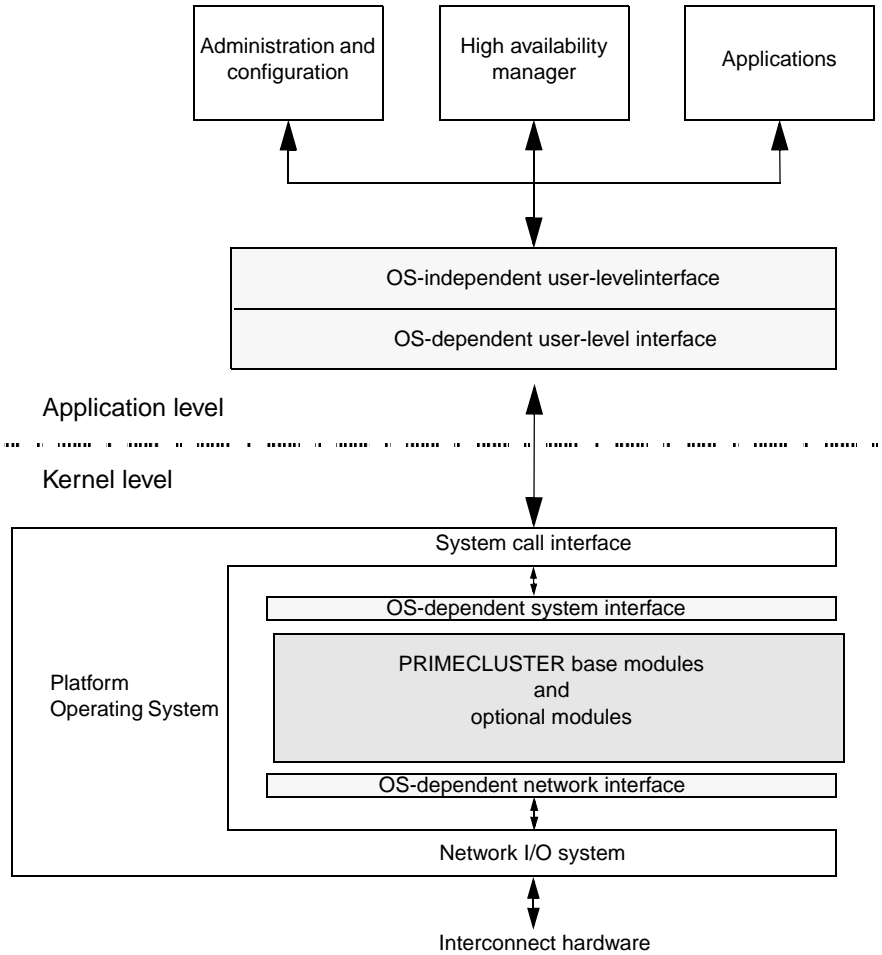


Figure 9: PRIMECLUSTER framework overview

## 3.2 PRIMECLUSTER key design features

The PRIMECLUSTER clustering software has been designed to satisfy the following goals:

- Modularity
- Platform independence
- Scalability
- Availability
- Guaranteed data integrity

### 3.2.1 Modularity

PRIMECLUSTER is composed of a core set of modules called the Cluster Foundation (CF) that provide the basic clustering functions. In addition, PRIMECLUSTER has optional modules such as the Parallel Application Services (PAS) module, the Scalable Internet Services (SIS) module, and the Reliant Monitor Services (RMS) module.

### 3.2.2 Platform independence

PRIMECLUSTER is independent of the operating system or hardware platform. Its modules are designed and coded based on an abstraction of an operating system's kernel functions. This is done with an OSD adaptation layer specific to different operating systems and network interconnects. This approach allows PRIMECLUSTER to plug seamlessly into the supported operating system without any modification to that operating system.

### 3.2.3 Scalability

PRIMECLUSTER provides not only high availability, but scalability as well. PRIMECLUSTER allows multiple nodes to act together to provide a common service. For example, with SIS, you can distribute web hits across all nodes in a cluster. With PAS, you can run a parallel database across multiple nodes. GFS provides you with a clusterwide file system so cooperating processes on many nodes can access the same data.

PRIMECLUSTER's scalability is important for customers who find that an application's demand for resources (especially CPU resources) has exceeded the capacity of a single machine. By clustering the nodes, more capacity is available for these types of applications.

### 3.2.4 Availability

PRIMECLUSTER implements a symmetric software architecture: All cluster information is fully replicated across nodes, thus avoiding single software points of failure. PRIMECLUSTER also supports multiple redundant interconnects, avoiding single hardware points of failure. RMS, the PRIMECLUSTER high availability manager, ensures the availability of applications by restarting or recovering applications on other nodes if there is a node failure. PRIMECLUSTER also includes an optional network load-balancing module (SIS, GLS) that can improve network availability.

If two or more points fail at the same time, a failover of the user application may not occur to protect data integrity.

### 3.2.5 Guaranteed data integrity

PRIMECLUSTER's algorithms guarantee that network partitions (or split-brain scenarios)—even during multiple hardware interconnect failures—do not result in data inconsistency problems. The quorum algorithms ensure that only one partial cluster can operate during a network partition.

### 3.3 PRIMECLUSTER components

The PRIMECLUSTER core component, Cluster Foundation (CF), provides the basic cluster services on which all other components are built. CF includes:

- Cluster Admin—Supplies the interface for cluster administration, configuration, monitoring, and diagnostics services.
- Web-Based Admin View—Provides a framework under which all the PRIMECLUSTER GUIs, including Cluster Admin, run.

The optional components and products are as follows:

- Reliant Monitor Services (RMS)—Manages high-availability switchover (failover) of application processes and their resources.
- RMS Wizard Tools —RMS configuration tool foundations.  
RMS Wizard Tools is mutually exclusive with respect to installation and operation.
- Parallel Application Services (PAS)—Provides a high-performance, low-latency communication facility that can be used by applications.
- Scalable Internet Services (SIS)—Performs network load balancing and provides a highly available interface to the cluster.
- Global Disk Services (GDS)—Provides the volume management component that improves the availability and manageability of disk-stored data.
- Global File Services (GFS)—Provides a file system that can be accessed from two or more nodes to which a shared disk unit is connected. (Only in Oracle Solaris 10 environment)
- Global Link Services (GLS)—Enables high reliability communications through the use of multiple network interface cards to create multiple redundant paths to a local system.

### 3.3.1 CF

The Cluster Foundation is the base on which all the other modules are built. It provides the fundamental services, such as the OSD, that all other PRIMECLUSTER components use as well as the base cluster services.

CF has the following features:

- Contains a loadable pseudo device driver that automatically loads when the system starts
- Supplies the CF driver that contains the CF kernel-level OSD and generic modules

Some of the functions that CF provides are detailed in the sections that follow.

#### 3.3.1.1 OSD

The CF operating-system-dependant (OSD) module provides an interface between the native OS and the abstract, OS-independent interface upon which all PRIMECLUSTER modules depend. This allows PRIMECLUSTER to use the same source files for all supported operating systems and architectures. The two main advantages of this design are as follows:

- Only need to maintain one version of the source
- Simplifies the porting of CF to a new operating system or architecture

#### 3.3.1.2 ICF

The Internode Communication Facility (ICF) module is the network transport layer for all PRIMECLUSTER inter-node communications. It provides the following functions:

- Ordered, guaranteed, node-to-node datagram communication services
- Guarantees to deliver messages to the destination node in the queued for transmission order, unless the destination node fails

- Interfaces via OS-dependent code to the Network I/O sub-system

To avoid a single point of hardware failure, ICF supports multiple interconnects. When multiple interconnects are available, ICF spreads messages across all available interconnects to improve performance. ICF automatically switches between interconnects when a failure occurs. ICF has a route recovery mechanism for transient interconnect failures such as a network switch being powered off and on again.

ICF is only usable by the CF internal components and is not available to user-level resources. To provide applications with an access to the cluster interconnect, Cluster Interconnect Protocol (CIP) is used. CIP provides a standard TCP/IP protocol suite over ICF.

### 3.3.1.3 JOIN

The cluster join services module (JOIN) dynamically joins a node to a cluster. If the cluster does not exist, then CF creates it. It is possible for several nodes to simultaneously try to form a cluster. The mastering algorithm solves this problem by using a distributed-election algorithm to determine which node should become master and form the cluster. Each node has equal rank, and each node can form the initial one-node cluster.

After the initial cluster is formed, all other nodes can join it. JOIN has built-in support for rolling upgrades by providing versioning information during the initial mastering phase. A new node joining the cluster automatically selects the protocol version in use by the current cluster.

### 3.3.1.4 ENS

The Event Notification Services (ENS) module provides an atomic-broadcast facility for events. Messages queued to ENS are guaranteed to either be delivered to all of the nodes or to none of the nodes. PRIMECLUSTER modules and application programs can both use ENS. Applications can register with ENS to receive notification of cluster events such as nodes joining or leaving the cluster. Applications can also define and broadcast application-specific events to other applications that register for it.



### 3.3.1.5 Cluster Admin

The Cluster Admin manager is an administrative interface for the following cluster features:

- Configuration
- Administration
- Operations and diagnostics services

Administration can be done from any node in the cluster, remotely from the Internet, or from both. A Java-enabled Web browser serves as the administrative interface; a conventional command-line interface is also available on a node. Diverse, clear-reporting metrics and event logs provide concise and timely information on the state of the cluster.

### 3.3.1.6 Web-Based Admin View

Web-Based Admin View is a GUI framework used by the PRIMECLUSTER products. The features of Web-Based Admin View are as follows:

- Common framework for multiple GUIs  
In addition to the Cluster Admin GUI, which controls CF, SIS, RMS, and SF, PRIMECLUSTER contains GUIs for other services such as GDS and GFS. In the Web-Based Admin View, all of these GUIs operate as a common framework.
- A single login for multiple GUIs.
- Password encryption. Passwords sent from the client browser to the management server are encrypted.
- Logging of all GUI commands dealing with configuration or administration.
- The ability to off load the management overhead onto the management servers outside the cluster.



For additional information about Web-Based Admin View features, see your *Web-Based Admin View Operation Guide*.

### 3.3.1.7 PRIMECLUSTER SF

The PRIMECLUSTER Shutdown Facility (SF) provides an interface to guarantee machine shutdown during error processing within the cluster. PRIMECLUSTER SF is made up of the following major components:

- Shutdown Daemon (SD)—The SD monitors the state of cluster machines and provides an interface for gathering status and requesting manual machine shutdown.
- One or more Shutdown Agents (SA)—The SA's role is to guarantee the shutdown of a remote cluster node.
- MA (asynchronous monitoring)—In addition to the SA, the MA monitors the state of remote cluster nodes and immediately detects failures in those nodes.

The advantages of PRIMECLUSTER Shutdown Facility are as follows:

- Ability to shut down a cluster node with or without running RMS
- Ability to shut down a cluster node from any PRIMECLUSTER service-layer product
- Redundant shutdown methods are available

#### Monitoring Agent

The Monitoring Agent (MA) has the capability to monitor the state of a system and promptly detect a failure such as system panic and shutdown. This function is provided by taking advantage of the hardware features that detect the state transition and inform the upper-level modules.

Without the MA, the cluster heartbeat time-out detects only a communication failure during periodic intervals. The MA allows the PRIMECLUSTER system to quickly detect a node failure.

The MA provides the following functions:

- Monitors the remote node state—The MA monitors the state of the remote node that uses the hardware features. It also notifies the Shutdown Facility (SF) of a failure in the event of an unexpected system panic and shutoff. Even when a request of responding to heartbeat is temporarily disconnected between cluster nodes because of an overloaded system, the MA recognizes the correct node state.
- Eliminates the remote node—The MA provides a function to forcibly shut down the node as Shutdown Agent (SA).

The hardware products supported by the MA are as follows:

- RCI—The MA monitors the node state and detects a node failure by using the SCF/RCI mounted on SPARC Enterprise M-series. The System Control Facility (SCF), which is implemented on a hardware platform, monitors the hardware state and notifies the upper-level modules. The MA assures node elimination and prevents access to the shared disk.

**i** Node state monitoring of the RCI asynchronous monitoring function operates from when message (a) shown below is output until message (b) is output. The messages for the console asynchronous monitoring function are messages (c) and (d). When node state monitoring is disabled, the function that forcibly stops nodes may not operate normally. The messages for the MMB asynchronous monitoring function are messages (e) and (f).

(a) FJSVcluster:INFO:DEV:3042: The RCI monitoring agent has been started.

(b) FJSVcluster:INFO:DEV:3043: The RCI monitoring agent has been stopped.

(c) FJSVcluster:INFO:DEV:3040: The console monitoring agent has been started (node:monitored node name).

(d) FJSVcluster:INFO:DEV:3041: The console monitoring agent has been stopped (node:monitored node name).

(e) FJSVcluster:INFO:DEV:3080: The MMB monitoring agent has been started.

(f) FJSVcluster:INFO:DEV:3081: The MMB monitoring agent has been stopped.

- XSCF/ILOM—(Available server models are limited to SPARC Enterprise M-series and most of SPARC Enterprise T-series.) The console monitoring agent monitors message output to the console of each node. If an error message of a node failure is output to one node, the other node detects the message and notifies SF of a node failure. Normally, the console monitoring agent creates a loop, monitoring another node, for example, A controls B, B controls C, and C controls A. If one node goes down because of a failure, another node takes over the monitoring role instead of this failed node.

The console monitoring agent also ensures node elimination by sending a break signal to the failed node.

Figure 10 shows how the monitoring feature is taken over in a cluster system with three nodes if one node goes down. The arrow indicates that a node monitors another node.

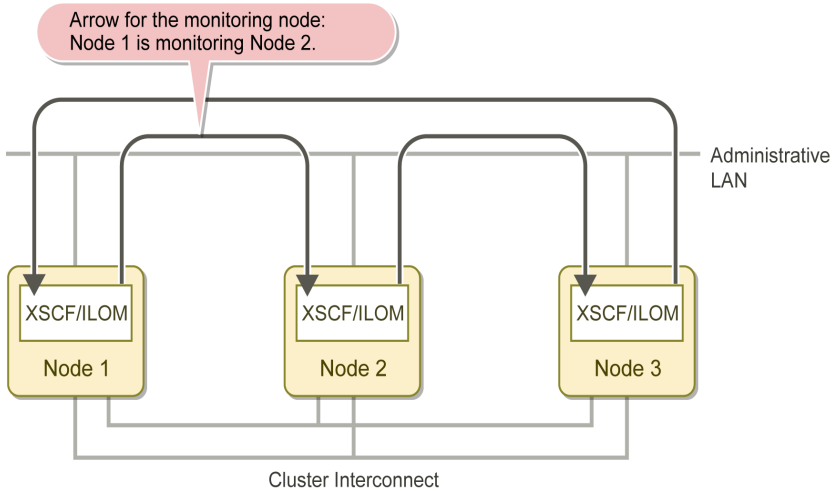


Figure 10: MA normal operation

When a failure occurs, and Node 2 is DOWN, the following actions occur:

- Node 1 begins to monitor Node 3.
- The following message is output to the `/var/adm/messages` file of Node 1:

```
FJSVcluster:Information:DEV:3044: The console monitoring agent took over monitoring (node: targetnode)
```

Figure 11 shows how Node 1 added Node 3 as the monitored node when Node 2 went down.

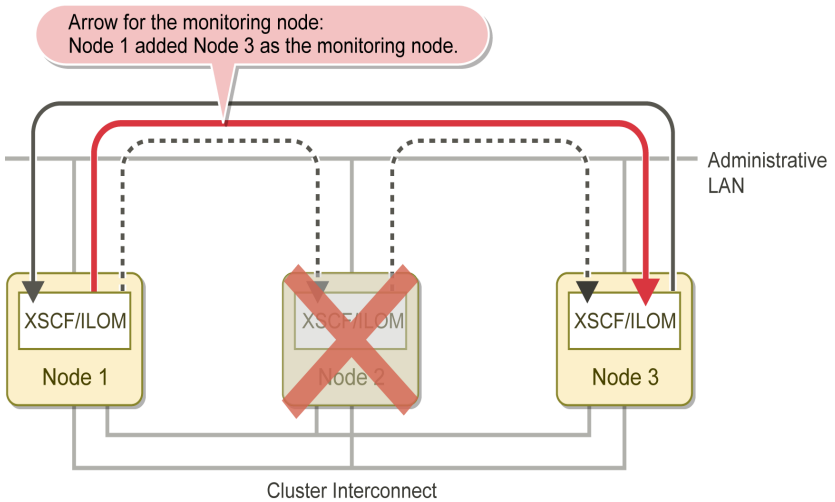


Figure 11: MA operation in the event of node failure

**i** If monitoring function is taken over while the console monitoring agent is stopped, the stopped console monitoring agent is resumed.

When Node 2 recovers from the failure and starts, the following actions occur:

- The original monitoring mode is restored.
- The following message is output to the `/var/adm/messages` file of Node 1:

```
FJSVcluster:Information:DEV:3045: The console monitoring agent cancelled to monitor (node: targetnode)
```

Figure 12 shows how Node 2 returns to monitoring Node 3 once it has been restored to the cluster.

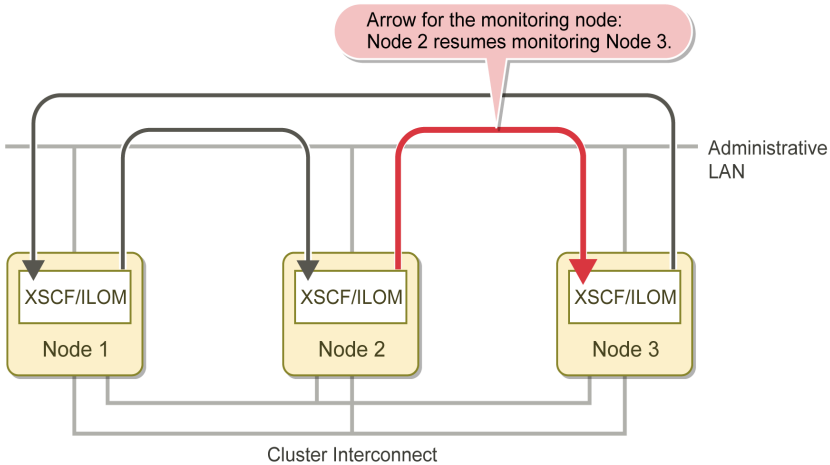


Figure 12: Node recovery

The following are possible messages that might be found in the `/var/adm/messages` file:

- `FJsvcluster:Information:DEV:3042: The RCI monitoring agent has been started`

Indicates that the RCI monitoring agent is enabled.

- `FJsvcluster:Information:DEV:3043: The RCI monitoring agent has been stopped`

Indicates that the monitoring feature is disabled.

- `FJsvcluster:Information:DEV:3040: The console monitoring agent has been started (node:monitored node name)`

Indicates that the monitoring feature of the console monitoring agent is enabled.

- `FJsvcluster:Information:DEV:3041: The console monitoring agent has been stopped (node:monitored node name)`

Indicates that the monitoring feature of the console monitoring agent is disabled. When the monitoring feature is not enabled, the other feature that forcibly brings the node DOWN might not work.



The console monitoring agent monitors the console message of the remote node. So it cannot recognize the node state in the event of an unexpected shutdown. In such a case, the node goes into the LEFTCLUSTER state, and you need to mark the remote node DOWN.

- MMB asynchronous monitoring (MMB) (PRIMEQUEST)

This function uses the MMB, which is one of the hardware units installed in PRIMEQUEST, to monitor nodes. The function can ascertain node failures by having the MMB, which is one of the standard units installed in the hardware, report the node state to the software.

This function can intentionally trigger a panic or a reset in other nodes to forcibly stop those nodes with certainty and prevent contention over user resources.



- If a node error occurs while an MMB error has occurred, it may take longer than usual for the node failure to be ascertained (up to 6 seconds).
- If a node error occurs while an error has occurred in one of the MMB management LAN systems in all nodes, it may take longer than usual for the node failure to be ascertained (up to 6 seconds).
- It may take up to 20 minutes for an MMB error to be detected.
- If the MMB is recovered from an error, it takes up to 10 minutes for that recovery to be detected. However, if a node error occurs before the recovery is detected, the recovery is recognized at that point, and the MMB asynchronous function actually operates without any problems. To have an MMB recovery detected immediately after the MMB is recovered from an error, restart the Shutdown Facility (SF).
- If an error (snmptrapd is stopped) occurs in the local node, the following message is displayed:

```
FJSVcluster:INFO:DEV:3084:Monitoring another node has been stopped.
```

In this case, node state monitoring is disabled. Therefore if a node error occurs under this condition, it may take longer than usual for the node failure to be ascertained.

Also after a node is started or after the Shutdown Facility is restarted, node state monitoring is disabled even if the following message is not displayed. Therefore if a node error occurs under this condition, it may take longer than usual for the node failure to be ascertained.

```
FJSVcluster:INFO:DEV:3083:Monitoring another node has been started.
```

- The following message may be displayed while a node is being started:

```
FJSVcluster: INFO: DEV: 3084: Monitoring another node has been stopped.
```

This condition occurs because `snmptrapd` is being started, and there is not problem as long as the following message is output within about 10 minutes after `snmptrapd` is started:

```
FJSVcluster: INFO: DEV: 3083: Monitoring another node has been started.
```

- If you execute `sdtool -s` immediately after the node is started, Test Failed may be displayed as the test state of the local node. To check the test state of the local node immediately after the node is started, execute `sdtool -r` and then execute `sdtool -s`.



The MMB asynchronous monitoring function displays the following messages if the MMB management LAN is disconnected:

```
FJSVcluster:WARN:DEV:5021:An error has been detected in part of the transmissionroute to MMB. (node:nodename mmb_ipaddress1:mmb_ipaddress1 mmb_ipaddress2:mmb_ipaddress2 node_ipaddress1:node_ipaddress1 node_ipaddress2:node_ipaddress2)
```

```
FJSVcluster:ERROR:DEV:7213:An error has been detected in the transmission route to MMB. (node:nodename mmb_ipaddress1:mmb_ipaddress1 mmb_ipaddress2:mmb_ipaddress2 node_ipaddress1:node_ipaddress1 node_ipaddress2:node_ipaddress2)
```

Respond according to the action for each message. If error message 7213 is displayed, node state monitoring is disabled. Therefore the function that forcibly stops other nodes may not operate normally. In addition, if an error (for example, `snmptrapd` terminates abnormally) occurs in the local node, the following message is displayed:



```
FJSVcluster:ERROR:DEV:7210:An error was detected in MMB.  
(node:nodename mmb_ipaddress1:mmb_ipaddress1  
mmb_ipaddress2:mmb_ipaddress2  
node_ipaddress1:node_ipaddress1  
node_ipaddress2:node_ipaddress2 status:status detail:detail)
```

In this case as well, node state monitoring is disabled. Therefore the function that forcibly stops other nodes may not operate normally. For details about this message, see "Monitoring Agent messages" in the PRIMECLUSTER *Cluster Foundation Configuration and Administration Guide*.

### Shutdown Agents (SA)

The PRIMECLUSTER Shutdown Facility provides the following Shutdown Agents:

- RCI (SA\_pprcip, SA\_pprcir): Remote Cabinet Interface  
This SA uses the RCI, which is one of the hardware units installed in SPARC Enterprise M-series, to stop other nodes with certainty by intentionally triggering a panic or reset in those nodes.
- XSCF (SA\_xscfp, SA\_xscfr, SA\_rccu, SA\_rccux): eXtended System Control Facility  
The SA uses the XSCF, which is one of the hardware units installed in SPARC Enterprise M-series, to stop other nodes with certainty by intentionally triggering a panic or reset in those nodes.  
If the XSCF is being used in the console, the Shutdown Facility stops other nodes with certainty by sending the break signal to those nodes.
- ALOM (SA\_sunF): Advanced Lights Out Management  
The SA uses ALOM of SPARC Enterprise T1000, T2000 to stop other nodes with certainty by sending the break signal to those nodes.
- ILOM (SA\_ilomp, SA\_ilomr): Integrated Lights Out Manager  
The SA uses ILOM of SPARC Enterprise T5120, T5220, T5140, T5240, T5440, SPARC T3 series to stop other nodes with certainty by intentionally triggering a panic or reset in those nodes.

- **NPS (SA\_wtinps): Network Power Switch (not supported)**

The SA uses the Western Telematic Inc.'s Network Power Switch (WTINPS) to stop other nodes with certainty by shutting them down.
- **BLADE (SA\_blade)**

This SA, which can be used in the PRIMERGY blade server, uses the SNMP command to stop other nodes with certainty by shutting them down.
- **IPMI (SA\_ipmi): Intelligent Platform Management Interface**

This SA uses the IPMI, which is one of the hardware units installed in PRIMERGY, to stop other nodes with certainty by shutting them down.
- **kdump (SA\_kcd)**

The SA uses kdump in PRIMERGY or the PRIMERGY blade server to stop other nodes with certainty by intentionally triggering a panic.
- **MMB (SA\_mmbp, SA\_mnbr): Management Board**

This SA uses the MMB, which is one of the hardware units installed in PRIMEQUEST, to forcibly stop other nodes with certainty by intentionally triggering a panic or reset in those nodes.
- **vmSP (SA\_vmSPgp, SA\_vmSPgr)**

When using the Xen virtual machine in PRIMEQUEST, the SA stops other nodes (guest OSes) with certainty by intentionally triggering a panic or reset in those nodes.
- **ICMP (SA\_icmp)**

The SA uses the network path to check other nodes. If no response is received from other nodes, the SA determines that nodes are shut down. Other nodes are not forcibly shut down.

Figure 13 shows an example of state confirmation by SA\_icmp if one node (Node 2) goes down in a cluster system with two nodes.

If no response is received from Node 2 through all specified network paths, SA\_icmp determines that Node 2 is shut down.

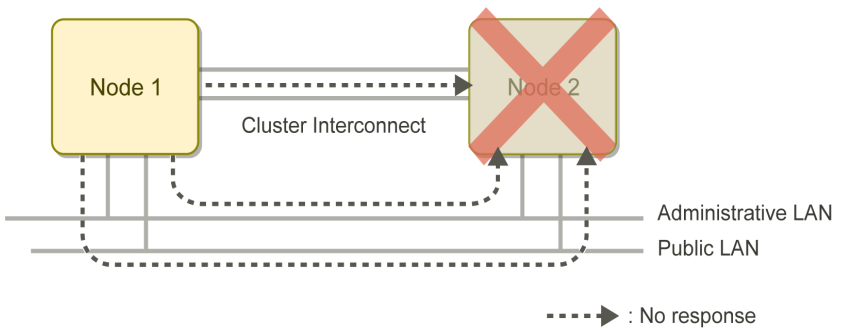


Figure 13: State confirmation by SA\_icmp if the other node goes down

Figure 14 shows an example of state confirmation by SA\_icmp if the cluster interconnect fails in a cluster system with two nodes.

If Node 1 receives a response from Node 2 on any of specified network path, SA\_icmp determines that Node 2 is running.

In this case, Node 2 is not forcibly shut down by SA\_icmp.

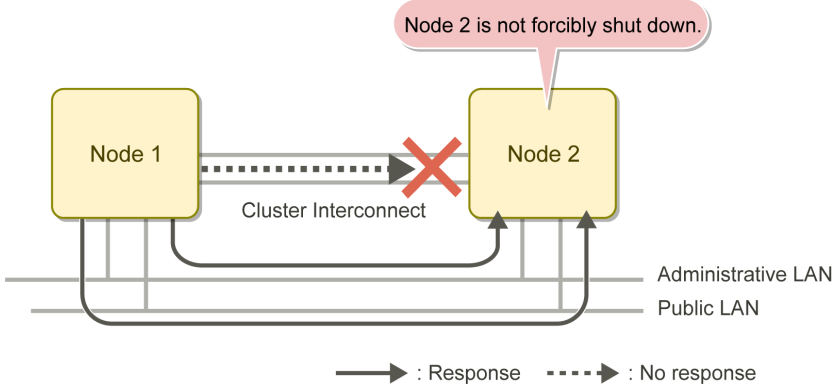


Figure 14: State confirmation by SA\_icmp if the cluster interconnect fails

- VMCHKHOST (SA\_vmchkhost)

When the cluster system is installed in the host OS with the Xen/KVM machine, the SA checks the status of guest OSES together with the cluster system of the host OS.

Other nodes are not forcibly shut down.

- libvirt (SA\_libvirtgp, SA\_libvirtgr)

When using a KVM virtual machine in PRIMERGY, the PRIMERGY blade server, and PRIMEQUEST 1000 series, the SA stops other nodes with certainty by intentionally triggering a panic or reset in those nodes.

### 3.3.2 RMS

RMS is an application availability manager that ensures the availability of both hardware and software resources in a cluster. This is accomplished through redundancy and through the ability to fail over monitored resources to surviving nodes.

Monitored resources can be almost any system component, such as the following:

- File system
- Volume (disk)
- Application
- Network interface
- Entire node

For redundancy, RMS uses multiple nodes in the cluster. Each node is configured to assume the resource load from any other node. In addition, RAID hardware and/or RAID software replicate data stored on secondary storage devices.

For application availability, RMS monitors resources with detector programs. When a resource fails, RMS triggers a user-defined response. Normally, the response is to make the resource available on other nodes.

Resources that are mutually dependent can be combined into logical groups such that the failure of any single resource in the group triggers a response for the entire group. During switchover, RMS ensures that all of a group's resources

on the original node (before the failure) are brought offline prior to any resources being brought online on the new node. This prevents any possibility of data corruption by two or more nodes attempting to access a resource simultaneously.

Figure 15 shows how RMS uses detectors to monitor resources. A detector reports any changes in the state of a resource to the RMS base monitor, which then determines if any action is required.

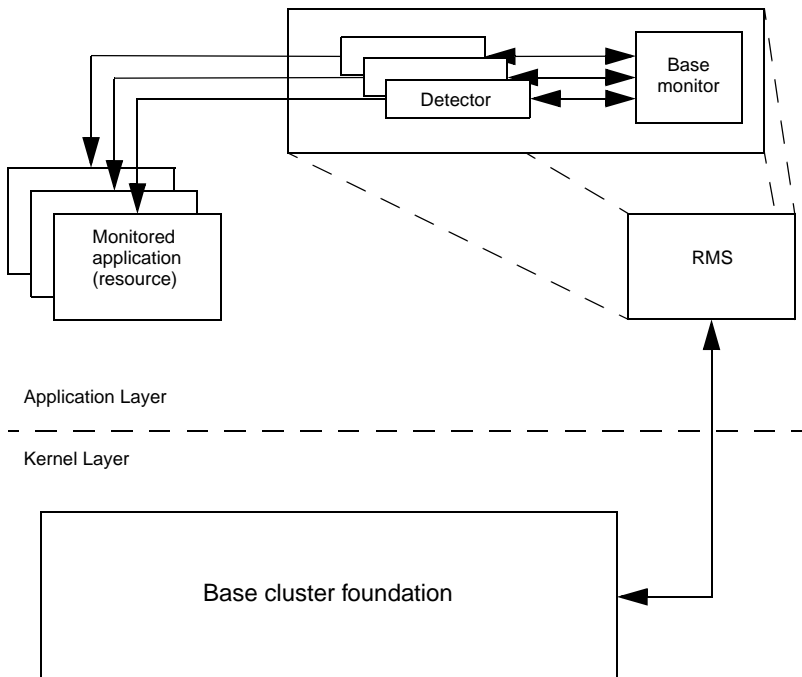


Figure 15: RMS resource monitoring

### 3.3.2.1 RMS configuration tools

RMS configuration tools facilitate the creation of RMS configurations and are divided into the following product areas:

- **RMS Wizard Tools**—This contains configuration tool foundations and interface with the RMS base. They simplify the configuration setup and work with RMS in the HA (high availability) cluster environment.

- RMS Wizard Kit—RMS Wizard Kit creates optimized HA environments for individual enterprise user applications.

### 3.3.3 SIS

SIS enables PRIMECLUSTER to act as a scalable, reliable, and easily managed network system by providing the following:

- Access through one or more Virtual Interface Provider (VIP) addresses, which appear to clients as a single network server
- Makes the cluster appear to clients as a single network server
- Provides transparent communication with the cluster by disguising the fact that the network request is serviced by one of the nodes in the cluster

*Example:*

Figure 16 is an example of a four-node SIS cluster. In this case, the SIS cluster is serving Web pages. The SIS cluster has one virtual address, `www.mycluster.com`, and when a client connects to this address, the Web pages are served by any of the available nodes.

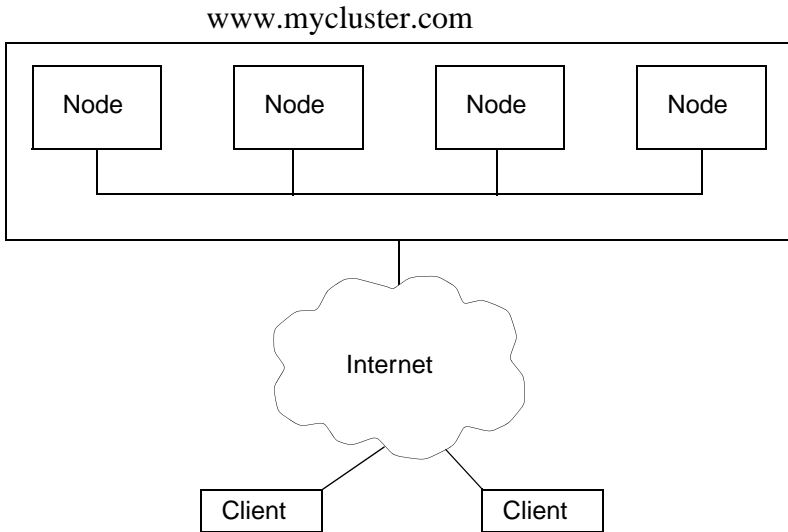


Figure 16: Example of a SIS cluster

### 3.3.4 PAS

Parallel database applications were one of the first commercial applications to use cluster technology. By partitioning the workload and data across multiple nodes in a cluster, parallel database applications can achieve performance beyond the limits of a single, large multiprocessor server.

The OPS (Oracle Parallel Server) has been the leader in parallel database applications since it was first deployed on commercial UNIX platforms in the early 90s. The Oracle 9iRAC and Oracle 10g *Cache Fusion* technology, which take advantage of the high-speed, interprocess communication API that PRIMECLUSTER provides, now overcomes the performance bottleneck of previous OPS implementations (that is, where database instances communicated through a secondary storage system).

### 3.3.5 GDS

Global Disk Services (GDS) is a volume management software that improves the availability and manageability of disk-stored data. GDS protects data from hardware failures and operational mistakes, and supports the management of disk units.

GDS has following functions, which are closely related:

- To improve availability of disk data
- To improve manageability of disk data

GDS's mirroring function protects data from hardware failures by maintaining replicas of disk data on multiple disks. This allows users to continue to access the disk data without stopping the application in the event of an unexpected trouble.



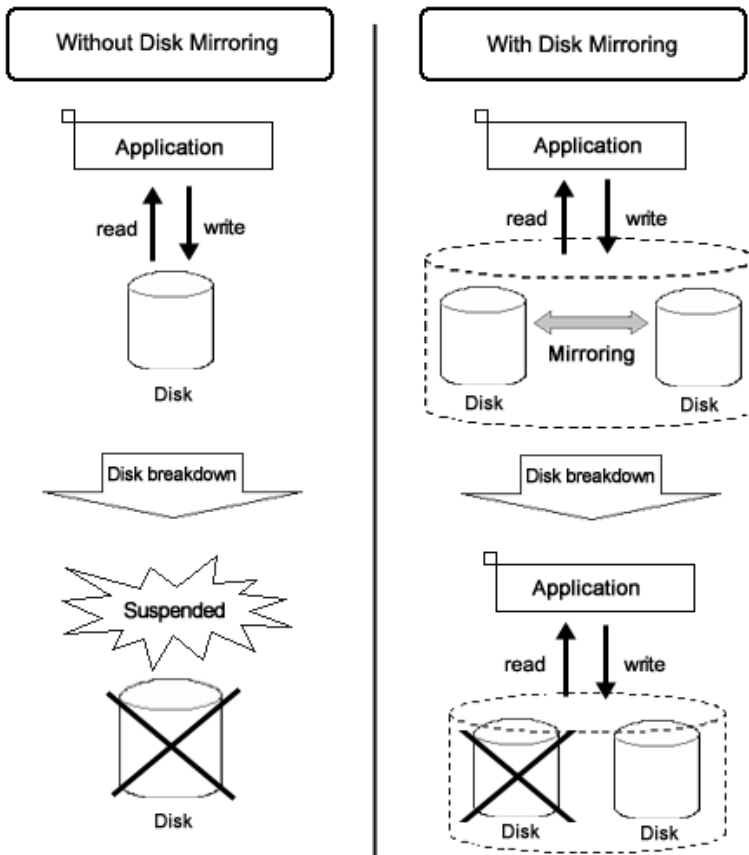


Figure 17: Disk Mirroring

The GDS management functions reduce the system administrator’s workloads of disk management. The user-friendly functions simplify management, and at the same time, prevent data corruption by operational mistakes.

In a SAN (Storage Area Network) environment, multiple servers can be connected to multiple disk units (see Figure 18). Disk-stored data can be accessed from those servers. This allows simultaneous access to a file system or database and improves the efficiency of data duplication between the servers and backup procedures. On the other hand, it also carries the risk of data damage, as multiple servers will compete to access the shared disk. Therefore, volume management functions suitable for the SAN environment are essential.

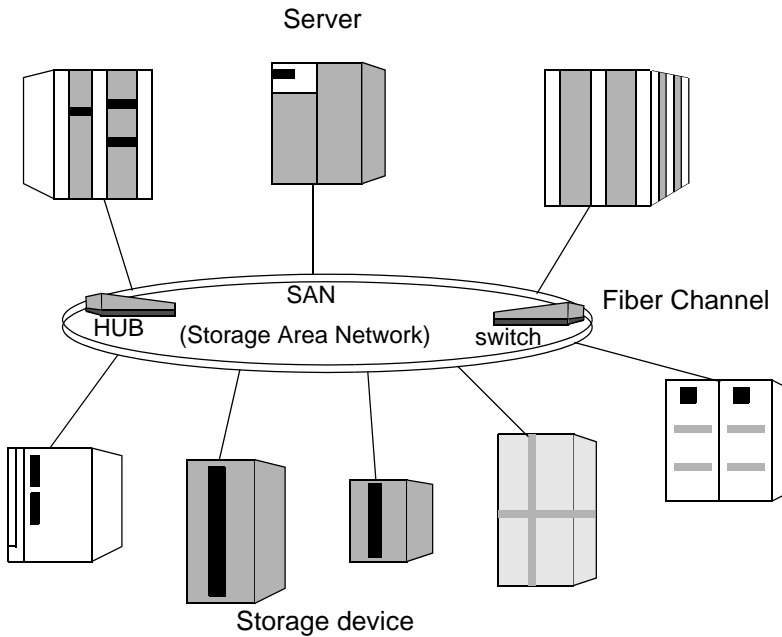


Figure 18: SAN (Storage Area Network)

GDS provides volume management functions suitable for SAN environments. GDS allows users to integrate management of all disk units connected to all servers, including local disk units that are connected to specific servers as well as disk units that are shared by multiple servers in a SAN environment.



PRIMERGY cannot manage system disks. In addition, if the Linux kernel is Version 2.4, PRIMERGY cannot manage locally connected disks.

GDS's main functions include:

- Mirroring of system disks (Solaris server and PRIMEQUEST)
- Mirroring of shared disks
- Mirroring of disk array units
- Hot spare that automatically recovers mirroring in the event of a disk failure
- Hot swap that replaces a failed disk without stopping applications

- JRM (Just Resynchronization Mechanism) that pursues high-speed recovery of mirroring in the event of an unexpected system down or cluster failover
- Integrated management and access control of disks in a SAN environment
- Automatic configuration that recognizes physical connections to servers and disk units, registers the configuration information, and checks the connections (Solaris server only)
- Concatenation that enables creation of large volumes of data
- Striping that distributes load of access to disks
- Logical partitioning that enables flexible use of disks
- Snap-shot that supports backup minimizing the effect on core services

See PRIMECLUSTER *Global Disk Services Configuration and Administration Guide* for further details.

### 3.3.6 GFS

The GFS file system provides the GFS shared file system which can be simultaneously shared with multiple nodes.

#### 3.3.6.1 GFS shared file system

Global File Services (GFS) is a highly reliable file system, assuring simultaneous access from multiple nodes to which a shared disk unit is connected (up to 2 nodes for Linux).

GFS provides the following features:

- Simultaneous shared access to a file or file system from two or more nodes
- Data consistency and integrity when file data is referenced or updated by two or more nodes
- When a node is down, file operation can be continued on the other node while maintaining the consistency of the file system
- File system high-speed recovery function
- High-speed input/output (I/O) processing with sequential block assignment of file area

- File access using file cache of each node
- Multi-volume supports the distribution of input/output processing load and the use of large-scale file systems
- Addition of online volume (area extension) is possible without disrupting or reconfiguring the file system
- Creation, deletion, expansion, and operation of the file system can be performed using a GUI based Web browser interface

### **Simultaneous shared access while maintaining consistency**

The GFS shared file system assures integrity of data when data is updated from multiple nodes. In the Solaris version, the file-lock function is enabled on multiple nodes using a conventional UFS API. When a distributed application is executed on multiple nodes, the conventional APIs are used ensuring application data transfer.

### **High availability**

When the GFS file system is used from multiple nodes, file access can be continued from the other node even if a node is down. The file system information retained by the downed node automatically restores the consistency within the GFS on the other nodes. That is, the application program running on the other nodes can continue processing without causing an error in the file system operation.

The operations required to change file system structure (such as file creation or deletion) are recorded in the area called the update log with the GFS file system. By using the information stored in this area, a system failure can be recovered in seconds without having to make a full check of the file system structure.

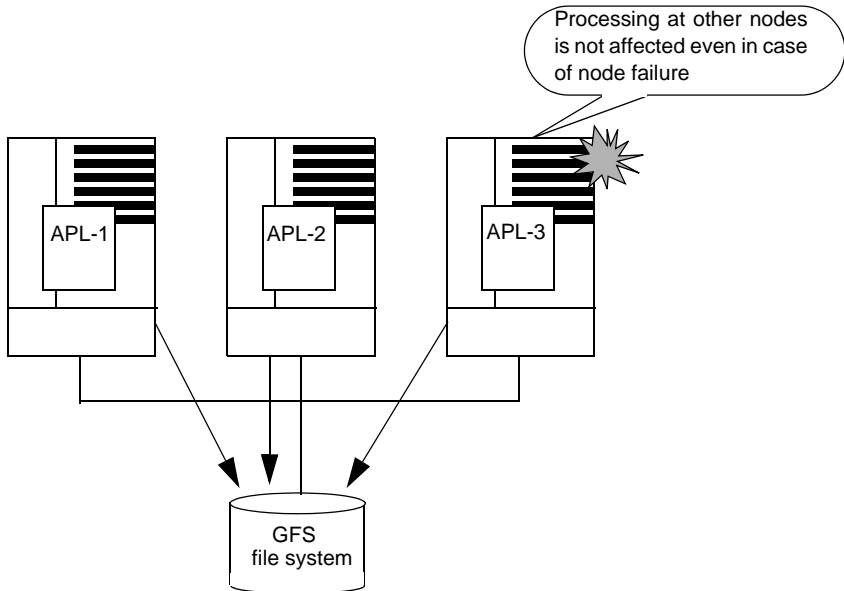


Figure 19: Operation continuation when a node is down

### Data accessibility performance

GFS enables a file system on a shared disk unit to be accessed from two or more nodes. With a conventional distributed file system, data is transferred to the client that requested access from the server on which file system data is managed by means of network communication over a LAN. The requested node directly accesses the disk unit. This reduces the network load from NFS and speeds up the response time required to read or write the request.

By allocating sequential blocks to the file data, the GFS file system enables collective I/O processing to improve the file system performance.

The GFS provides a feature that integrates multiple partitions into one file system. In the case a configuration with multiple partitions, the round-robin allocation method is used, so the file data area can be used from different partitions for each file. Therefore, the I/O load can be distributed into multiple disk units and the file system performance is improved. This function makes it easy to add the data partition described afterwards.

## Scalability

With the GFS file system, the file system can be easily expanded by specifying an empty disk partition. Thus, a shortage of free space in a file system can be solved in a short time. A data partition can be added even while mounting.

### 3.3.6.2 Benefits

GFS has the following benefits:

- The use of file cache on each system and high speed data access not using the LAN improve access performance.
- CPU load distribution of application is enabled via files on two or more systems assuring data integrity.
- A high availability system is provided by continuing file access on other node when the current node goes down.
- Area management on extent base and multi-volume support provides high-speed file access.
- Multi-volume supports the use of large-scale file systems and addition of online volume (area extension) are enabled without disrupting or reconfiguring the file system, and it makes the resource management easy.
- Operation of the file system using a GUI based Web browser interface eases environment configuration and management.

### 3.3.7 GLS

Global Link Services (GLS) is a software product that enables high reliability communications through the use of multiple network interface cards (NICs) to create redundant multiple transmission paths to a local system. Global Link Services provides network solutions that are suitable for systems in which communications continuity is important.

The benefits of Global Link Services are as follows:

- Multiple NICs can provide redundant transmission paths to offer high availability and path failure protection.
- The use of GLS means that applications can continue to run even in the event of changes in the LAN configuration, the redundant transfer route, or any network fault that may occur in a transfer route.

When using PRIMECLUSTER, you can choose the following mode for dual transfer routes: NIC switching mode - Creates a transfer route between a Switch/HUB and servers on the same network.

For details on the features of Global Link Services, see the PRIMECLUSTER *Global Link Services Configuration and Administration Guide: Redundant Line Control Function*, and *PRIMECLUSTER Global Link Services Configuration and Administration Guide: Multipath Function*.

### NIC switching mode

In NIC switching mode, duplexed NICs (LAN cards) are connected on the same network and switching of the transfer route is controlled using the NIC. There are no restrictions on the devices that can be connected, and it is possible to communicate with hosts on another network. To duplex the entire communication route, it is also necessary to duplicate the network equipment, such as routers, on the transfer routes and on all hosts because the duplexing of the paths should extend up to the direct connections to switches and HUBs.

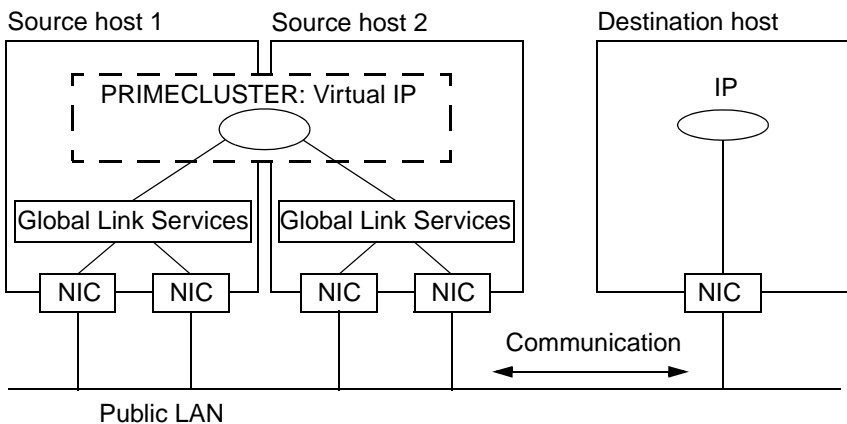


Figure 20: NIC switching mode





---

## 4 Cluster interconnect details

This chapter discusses the differences between a cluster interconnect and a network as well as the requirements of the PRIMECLUSTER cluster interconnects. It ends with a discussion of design considerations for the cluster interconnects.

This chapter discusses the following:

- The section “Overview” provides an overview of the cluster interconnect.
- The section “Cluster interconnect requirements” details the requirements of PRIMECLUSTER cluster interconnects.

### 4.1 Overview

The cluster interconnect is the most fundamental part of the cluster. All of the cluster's services rely on the cluster interconnects to transport messages between nodes and to determine the state of a node by its ability to respond to heartbeat requests.

#### 4.1.1 A cluster interconnect is different from a network

The function of a cluster interconnect is different than the traditional use of a network in several ways; therefore, it is useful to think of them as separate technologies. The highest priority use of the cluster interconnects is to carry heartbeat requests and responses.

Heartbeat messages are used to determine the state of the nodes and the cluster interconnects. When a message fails to get through, the cluster software assumes that a failure has occurred and takes action to recover. However, no network is 100 percent reliable, and the PRIMECLUSTER ICF protocol tolerates errors such as lost packets or out of order delivery.

In the cluster interconnects, the physical connections are redundant so that if one fails, one or more remain to carry the messages. However, a sustained outage of all cluster interconnects results in the cluster management software taking action to recover as if a node had failed.

## 4.1.2 Network availability

Most users assume that their networks are always available. However, most networks have periods of downtime quite regularly. Network administrators need to reprogram switches and routers and perform other maintenance operations; they do this at times of low network usage and the outages last for only a few seconds. A cluster uses its interconnect continuously, 24 hours a day, 7 days a week. As a result, a routine network maintenance operation that would not even be noticed by a typical network user or application can result in a cluster recovery action that shuts down systems and moves applications.

Temporary outages can result from other causes as well. One of the most convenient improvements to Ethernet has been the speed-agile ports on hubs, switches, and routers. These allow the user to plug an Ethernet device into a port and not have to worry about speed matches between the network interface card (NIC) and the port. However, there is a cost to this technology. If a hub or switch gets too busy, it can lose its state and will have to renegotiate with the NIC. This renegotiation takes several seconds, during which time no messages are passed. As a result, a cluster interconnect on this type of device could disconnect long enough so that the cluster takes a recovery action such as shutting down a node.



To avoid renegotiation outages, it is recommended that users configure the network interface card, router, and hub at a fixed speed. If all of the redundant interconnects are unavailable for five seconds, then the software assumes that a node failure has occurred.

Another temporary outage can occur when an Ethernet device is first opened. This starts the chain of events that triggers the switch port to which the Ethernet device is attached to begin its configuration process. Some switches take over one minute for this configuration to complete. PRIMECLUSTER may interpret the lack of responses on this port incorrectly and assume no other cluster nodes are configured on this port.


Because different network configurations and devices have different requirements, PRIMECLUSTER allows the system administrator to tune the parameters associated with the cluster interconnect. Refer to the *Cluster Foundation Configuration and Administration Guide* for more details.

### 4.1.3 Interconnect protocol

PRIMECLUSTER supports either the Ethernet protocol or IP as the transport layer for the cluster interconnects. This does not mean that TCP/IP is the underlying protocol used by the cluster services.

TCP/IP is designed to reliably deliver messages through a variety of obstacles such as different network types, multiple routers or switches, and failures in the network. TCP/IP works well and has been almost universally adopted as the protocol stack of choice for a wide variety of services. But TCP/IP is not the right choice for cluster interconnects.

The ICF (Internode Communication Facility) protocol used by PRIMECLUSTER is designed specifically for cluster communications. It is a low-latency protocol that guarantees ordered delivery of messages. ICF has a lower overhead than TCP/IP, which is important for parallel applications. ICF uses the Ethernet protocol or is a service over IP (CF/IP).

 Devices that only accept TCP/IP cannot route the ICF protocol when it is configured directly on the Ethernet. In this instance, if you need to use a router, it must be a level-two router.

ICF is only usable by the CF internal components and is not available to user-level resources. To provide applications with an access to the cluster interconnects, the Cluster Interconnect Protocol (CIP) is used. CIP provides a standard TCP/IP protocol suite over ICF.

## 4.2 Cluster interconnect requirements

PRIMECLUSTER uses all significant forms of Ethernet and most devices that support TCP/IP, including mixtures of different technologies. The interconnects must be redundant to assure reliable operation of the cluster; that is, there must be two or more independent connections (interconnects) between all of the nodes in the cluster.

When connecting multiple switches and routers by using interconnects, use VLAN to detach each switch or router logically.

Example:

When connecting multiple switches as the following Figure 21, use VLAN to detach each switch logically.

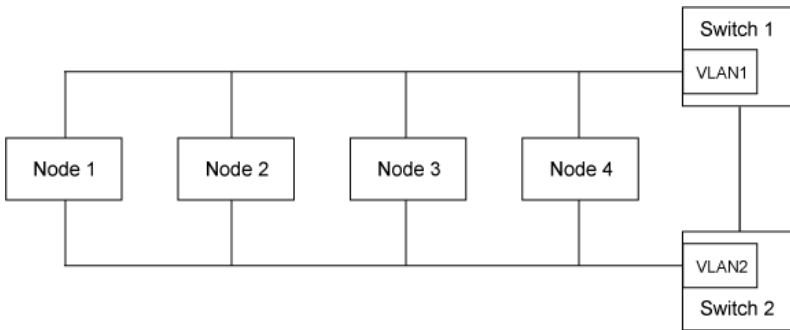


Figure 21: Connecting multiple switches

Each interconnect must support all of the nodes in the cluster. Refer to the Software Release Guide PRIMECLUSTER and the PRIMECLUSTER Installation Guide for a list of all the Ethernet devices that were known to work at the time of release.

## 4.2.1 Redundancy

To be redundant, the cluster interconnect must use two or more independent connections and data paths. An independent connection means that the Ethernets do not share any common components. An independent data path means that the individual interconnects do not share any part of their route between nodes. An example of a redundant cluster interconnect is as follows:

- Only one port is used for each Ethernet board. If more than one port of the same board is used as a cluster interconnect, these ports would be a common point of failure.
- Connections to nodes must be independent.
- No two interconnects can share the same hub, switch, or routers.

Figure 22 shows a typical four-node cluster. In this diagram, there are two switches. Each of these switches has their own power connection on different circuits (when connecting two switches, it is necessary to use VLAN to detach each switch logically). If the switches were connected to the same power strip in a rack, for example, the power strip would be a single point of failure for the cluster interconnects.

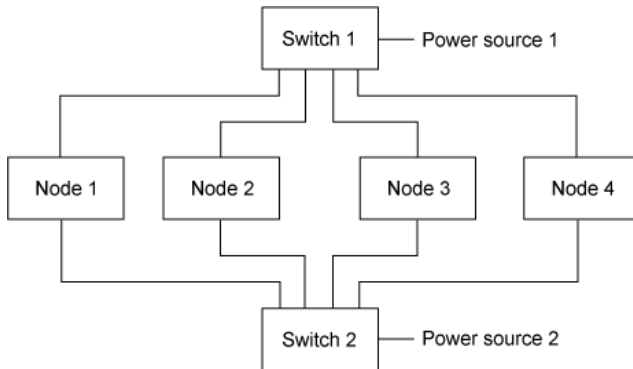


Figure 22: Typical four-node cluster

## 4.2.2 Routes

Since redundant interconnects are required for reliability, ICF is designed to use all of the available bandwidth. Each connection between two nodes in the cluster is called a route. When ICF has a message to send to another node, it chooses a route so that the message traffic is balanced between all the available routes. In the four-node cluster in Figure 22, each node has two routes to each of the other nodes.

### 4.2.2.1 Heartbeats

When everything is functioning properly in the cluster, all of the routes are in the UP state (available for use). Every UP route participates in carrying message traffic if it functions at the same speed (mixed speed interconnects are discussed below.) In addition, every route is always used for heartbeat requests. (A heartbeat is a message indicating that a node is functional.) If a certain number of heartbeat requests fail on a route, then that route is marked as DOWN. If a link down event in NIC is detected, also that route is marked as DOWN. A DOWN route is never used for message traffic. However, heartbeat requests are still attempted on the DOWN routes, and if one succeeds, the route is returned to the UP state. This last behavior is sometimes called self-healing routes. The use of all the UP routes for message traffic is called port aggregation or trunking.

Failures of heartbeat requests can come from several causes. When a network component fails, that route is not usable and the behavior is the same as described in the previous paragraph. When a node is `DOWN` to the only remaining route to another node, `PRIMECLUSTER` does not mark this route as `DOWN`, rather it marks the node as `LEFTCLUSTER`.

`LEFTCLUSTER` is a node state that indicates that the node cannot communicate with other nodes in the cluster. That is, the node has left the cluster. Once a node is marked as `LEFTCLUSTER` or `DOWN`, no more heartbeat attempts are made on any of the routes to the node until the node rejoins the cluster.



The last route to a node is never marked `DOWN`, even if the node is in the `LEFTCLUSTER` or `DOWN` state.

Sometimes, events happen on a node that prevent the node from responding to heartbeat requests for some reason other than a failure. For example, a Fibre Channel controller can prevent other network device drivers from executing while attempting a link recovery. This can result in a false detection of node failure. To allow for some flexibility in responding to this condition, the number of missed heartbeat requests on the final route to a node before declaring the node as `LEFTCLUSTER` is tunable.

You can only adjust the number of failed-heartbeat requests that occur before declaring the node `LEFTCLUSTER`. The other parameters for ICF are not tunable. It is important that the route-detection algorithm marks routes as down as soon as possible, so that messages are switched to alternate routes without a noticeable delay. If a route is momentarily not available, for whatever reason, the self-healing mechanism will quickly reactivate the route after it becomes functional. The *Cluster Foundation Configuration and Administration Guide* has details on tuning the ICF parameters.

`PRIMECLUSTER` also supports the use of non-symmetric interconnects, for example, one interconnect could use gigabit Ethernet and a second fast-Ethernet. When making decisions about routing, the interconnect speed is also considered. In the previous example, the gigabit Ethernet would always be used in preference to the fast-Ethernet whenever it is available. If there are multiple interconnects at the same speed, the port aggregation is done across all the devices with the same speed. Heartbeat requests are always sent on every interconnect, independent of the interconnect speed.

## 4.2.3 Properties

When designing cluster interconnects, it is important for users to consider the following:

- Bandwidth
- Latency
- Reliability
- Device interface
- Security

### 4.2.3.1 Bandwidth

PRIMECLUSTER does not require much bandwidth for its own use. This is significant because the end-user's application requirements for bandwidth are much more important. PRIMECLUSTER requires less than 2 Kbits/s on each of the cluster interconnects. Since the cluster interconnects share bandwidth, the total bandwidth available to the user's application is nearly the sum of all of the bandwidth for each interconnect.

Refer to table 1 as an example of bandwidth use. Suppose that in the configuration in Figure 22 that there are two 100 Mbits/s Ethernets configured for the cluster interconnect. Assume that the available bandwidth for each cluster interconnect is 10 Mbytes/s, and assume that the end-user application needs 4.5 Mbytes/s on each node for the cluster file system and other activities. (This is an example—the actual bandwidth used by an application varies, depending on the application.)

Item	Bandwidth	Total bandwidth
Two interconnects: 100 Mbits/s Ethernet	10 MBytes/s	20 MBytes/s
PRIMECLUSTER requirements per inter- connect and per node	2 Kbits/s (4 x 2 x 2 = 16 Kbits/s)	16 Kbits/s
User application per node	4.5 MBytes/s (4.5 x 4 = 18 MBytes/s)	18 MBytes/s

Table 1: Example of interconnects with two 100 Mbits/s Ethernet boards

Item	Bandwidth	Total bandwidth
Total		18.02 MBytes/s 90% total use

Table 1: Example of interconnects with two 100 Mbits/s Ethernet boards

For this example, two fast-Ethernet interconnects use over 90 percent of the bandwidth.

**i** It is recommended that an initial installation has at least 30 percent available bandwidth capacity because the latency of the interconnect is adversely affected when total use nears 100 percent.

For this example, workload and configuration, one additional fast-Ethernet interconnect should be added to provide the excess capacity. table 2 shows the same calculation with this addition.



Item	Bandwidth	Total bandwidth
Three interconnects: 100 Mbits/s Ethernet	10 MBytes/s	30 MBytes/s
PRIMECLUSTER requirements per inter- connect and per node	2 Kbits/s (4 x 3 x 2 = 24 Kbits/s)	24 Kbits/s
User application per node	4.5 MBytes/s (4.5 x 4 = 18 MBytes/s)	18 MBytes/s
Total		18.02 MBytes/s 60% total use

Table 2: Example of interconnects with three 100 Mbits/s Ethernet boards

This new configuration gives a comfortable 40 percent available bandwidth margin. PRIMECLUSTER supports a maximum of eight interconnect devices.

#### 4.2.3.2 Latency

There are no absolute latency requirements in PRIMECLUSTER. As previously stated, PRIMECLUSTER relies on heartbeat requests and responses to determine that nodes or other resources are functional. When a heartbeat is not received in a preset interval, PRIMECLUSTER starts recovery actions. The intervals depend upon what PRIMECLUSTER is monitoring. The Cluster Foundation (CF) software on each node sends a heartbeat request every 200 ms on each interconnect to every other node in the cluster. By default, CF tries 50 times to get a response before a node is marked as LEFTCLUSTER.

The 200 ms interval is a reasonable choice for a maximum latency in the cluster interconnects. This interval is long enough so that a small message and response can span transcontinental distances. This interval also allows for recovery from lost packets or other errors that can cause a few requests or responses to be lost in the interconnect.

For interconnect latency, an end-user application can have a different set of requirements. For example, applications like OPS will find that 200 ms is far too long an interval for latency. This shows how users must consider the needs of the applications during the design of their cluster solution.

All commercially available Ethernet hubs and switches meet the latency requirements of PRIMECLUSTER and those of most applications. When latency is measured with hubs and fast Ethernet, typical results are between 15 and 60 microseconds for user-level applications.

**i** Some routers, especially in heavily used networks, can introduce excessively high latencies. For this reason, it is recommended that you do not use routers with the cluster interconnect. If routers cannot be avoided, it is essential that you address the latency concerns.

### 4.2.3.3 Reliability

Ethernet as an interconnect technology has not shown any problems with PRIMECLUSTER. The communications protocol used by PRIMECLUSTER is ICF. ICF guarantees that messages are delivered correctly and in order to its clients. However, ICF was designed with fairly reliable communications in mind. When the cluster interconnect is reliable, ICF has very low overhead, but when it is unreliable, the overhead of ICF increases. This is similar to other protocols like TCP/IP; errors in the interconnect will result in messages being resent.

**i** Resending messages consumes bandwidth and increases latency and should be avoided at all times.

**i** An Ethernet error rate greater than 1 error per 1,000,000 bytes indicates that there is some problem with the Ethernet layer that should be investigated. (Use the command `netstat(1)` to find the error rate.)

### 4.2.3.4 Device interface

**i** This section is for Solaris only.

PRIMECLUSTER depends on the DLPI (Data Link Provider Interface) for devices in Solaris. If a device does not support a DLPI interface, PRIMECLUSTER does not recognize the device as eligible for use as a cluster interconnect. In addition, the device must appear to be an Ethernet device. Some devices support TCP/IP, but are not Ethernet-type devices. Keep in mind that PRIMECLUSTER does not use TCP/IP for its cluster interconnects; rather it uses the Ethernet protocols. It is advisable to choose an interconnect device from the list of supported devices (refer to the the *Software Release Guide PRIMECLUSTER* and the *PRIMECLUSTER Installation Guide* as detailed in the section "Related documentation").

#### 4.2.3.5 Security

With the PRIMECLUSTER family of products, it is assumed that the cluster interconnects are private networks; however, it is possible to use public networks as cluster interconnects because ICF does not interfere with other protocols running on the physical media. The security model for running PRIMECLUSTER depends on physical separation of the cluster interconnect networks.



For reasons of security, it is strongly recommended not to use public networks for the cluster interconnects.

The use of public networks for the cluster interconnects allows any machine on that public network to join the cluster (assuming that it is installed with the PRIMECLUSTER products). Once joined, an unauthorized user, through the node, would have full access to all cluster services.



---

## 5 Reliant Monitor Services (RMS)

This chapter introduces the basic concepts of RMS. It begins with an overview of the basic terms and concepts used to provide high availability by means of RMS, and then it goes into more detail on how RMS works.

This chapter discusses the following:

- The section “RMS overview” describes the concepts of high availability, redundancy, and switchover.
- The section “RMS monitoring and switchover” describes the components of RMS and explains how they work together to achieve high availability.
- The section “RMS administration” introduces administering RMS by means of the GUI (graphical user interface) and the CLI (command-line interface).
- The section “Customization options” discusses customizing RMS for a particular site.

### 5.1 RMS overview

RMS is a software monitor designed to provide system-level High Availability (HA) to applications. A minimum of two nodes with shared access to common storage are administered from a cluster console. As the PRIMECLUSTER HA monitor, RMS uses detectors to monitor the state of components and resources for applications. If a resource or application fails, the event is notified by the detector and reported to RMS. RMS then takes the appropriate action as described in the sections that follow.

Figure 23 shows the components of a basic RMS cluster on Solaris.

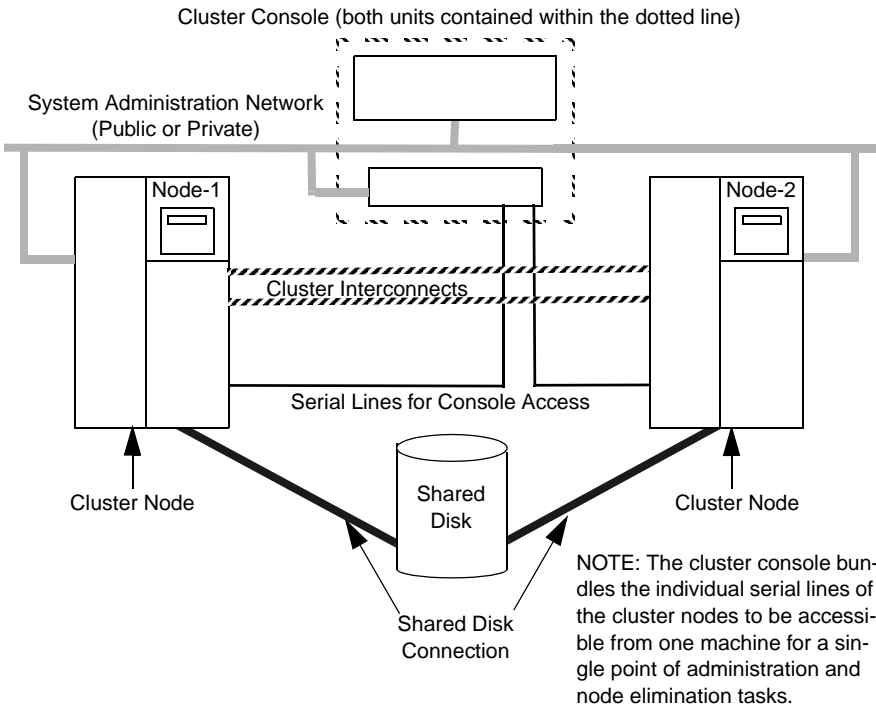


Figure 23: RMS cluster on Solaris

For RMS, high availability means *maximum availability of applications* rather than *uninterrupted availability of individual nodes*.

RMS accomplishes high availability in two ways: redundancy and switchover.

### 5.1.1 Redundancy

To provide high availability of RMS resources, RMS takes advantage of the following redundancy scheme:

- Multiple nodes, each configured to assume the resource load of any other RMS node
- Duplication of stored data using mirror disks, hardware RAID, and remote mirroring

- Multiple-path access to storage media
- Multiple communications channels dedicated to internode communications

### Multiple nodes

An RMS configuration is made up of multiple nodes, each containing identical operating systems and RMS software. The maximum number of nodes in an RMS configuration is theoretically unlimited. Refer to the *Software Release Guide PRIMECLUSTER* and the *PRIMECLUSTER Installation Guide* as detailed in the section “Related documentation” for the qualified number of nodes.

### Shared storage

All nodes in the RMS configuration must have access to whatever data is shared. Typically, all nodes have the ability to access shared disks over a SAN. However, other access methods, such as Network Attached Storage are possible.

### RMS network

RMS uses standard TCP/IP protocols for communication between the nodes in the configuration. Because this communication is to monitor the status of other nodes and because any communication failures would be likely to endanger consistency, RMS uses the Internode Communication Facility (ICF) interface provided by PRIMECLUSTER (refer to section “Interconnect protocol” for more information). The RMS network is layered over the PRIMECLUSTER interconnects, and they both share the same physical media.

## 5.1.2 Application switchover

RMS operates on an object-oriented basis. The objects can be almost any system component, such as virtual disks, file system mount points, processes, and so on. These objects are defined as *resources*. A resource is categorized into a grouping called a *object type*. An object type has specific properties, or attributes, which limit and define what monitoring or action can occur in relation to that resource. Resources are monitored by programs called *detectors*. This very general object-oriented design permits a high degree of flexibility in the type and level of monitoring.

Resources which are dependent on each other can be grouped together to form logical applications. Failure of any resource in such a group generally triggers a reaction by the entire application.

### 5.1.2.1 Automatic switchover

Any failure of a resource triggers a user-defined reaction. In most cases, the most significant reaction to failures is switchover.

A *switchover*, sometimes known as failover, consists of first bringing an application into a well-defined `Offline` state and then restarting the application under the control of RMS on another node. RMS supports symmetrical switchover, which means that every RMS node is able to take on resources from any other RMS node. For example, if the node that is running an application fails, RMS automatically shuts down the node where the failure occurred and redistributes its application load to another operational RMS node.

The details of performing automatic switchover are defined in user-specified scripts and configuration files that RMS accesses when a failure is recognized. The RMS wizards are generally used to create these files.

### 5.1.2.2 Manual switchover

Resources can be switched manually for such purposes as hardware maintenance. For example, in a two-node RMS configuration, all applications can be temporarily moved to one node while maintenance is performed on the other. Then all applications can be switched back to the operational node while maintenance is performed on the second node. The only impact to users might be a momentary interruption in service while the applications are being switched, and perhaps a slowdown in response time while all applications are operating on a single node.

### 5.1.2.3 IP aliasing

RMS uses IP aliasing to allow switchover of the communication link. It is possible for several IP addresses (aliases) to be allocated to one physical network interface. With IP aliasing, the user is able to continue communicating with the same IP address, even though the application is now running on another node.

### 5.1.2.4 Data integrity

RMS ensures that all resources of an application are offline on the current node before initiating a switchover. These resources are then switched online on another operational node. This behavior ensures that multiple nodes do not access the same resource, thus protecting against data loss.



In the rare event of simultaneous multiple faults, RMS protects against data corruption by preventing a switchover. This design means that in certain circumstances, switchover may be prevented entirely.

Although high availability is the goal of RMS, data integrity takes priority over high availability.

## 5.2 RMS monitoring and switchover

The RMS software is composed of the following processes, scripts, files, and parameters that work together to maintain high availability of the RMS resources:

- Base monitor
- Configuration files
- Configuration scripts
- Detectors
- RMS environment variables

Configuration files, scripts, and environment variables can be customized, allowing switchover scenarios to be tailored for site-specific needs (for information on customization, refer to the section “Customization options”).

### 5.2.1 Base monitor

The base monitor is the central process that monitors a node from the RMS cluster. The base monitor performs the following functions:

- Controls and coordinates all state changes in RMS
- Ensures that the appropriate action is taken if there are any problems with the monitored resources
- Obtains information concerning resources and the action to be taken from an RMS configuration file, which the system administrator creates, according to the requirements for use by RMS

The base monitor is started by the Cluster Admin GUI, or with the `hvc` command, and runs as a process under the name `bm` (base monitor) on every node in the cluster.

## 5.2.2 Configuration file

An RMS *configuration file* is a text file that is usually generated by the RMS Wizard Tools. It provides input to the base monitor that consists of definitions of the resources that are to be monitored by RMS, including their interdependencies.

### 5.2.2.1 Interdependencies

In RMS, the terms *parent* and *child* are used to represent dependent relationships between objects and their resource objects. The term *leaf object* is used to indicate that an object in a system graph has no children. In the configuration file, the leaf object definition is at the beginning of the file. A leaf object cannot have children, while a child can be both a parent and a child.

Figure 24 helps illustrate the parent/child relationships between objects.

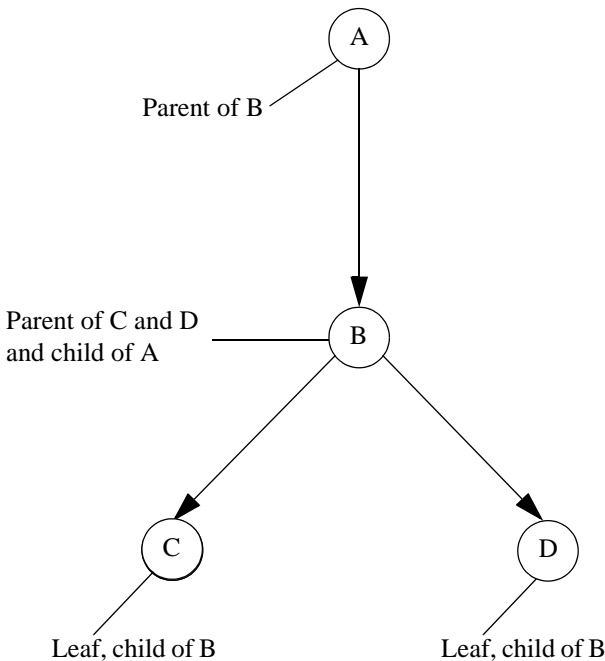


Figure 24: Configuration file interdependencies

Figure 24 shows the following relationships:

- Object A requires the resource B to function properly, making A a parent of B and B a child of A.
- Object B requires the resources C and D to function properly, making B a parent of C and a parent of D.
- Objects C and D are leaf objects because they do not have any children. They are also children of B.

### 5.2.2.2 Object types

A *object type* is a collection of similar resources monitored as a group (for example, all object types in a group using the same disk drives). Each object type has certain properties, or *attributes*, which limit or define what monitoring can occur for that type of resource. The attributes associated with a particular object type define how the base monitor acts and reacts for a resource of that object type during normal operations. Attributes commonly specify a device name or a script, and can be specified in any order in the object definition for that resource.

Some attributes are mandatory; others are optional. An example of a mandatory attribute is a device name for a `vdisk` object type. Typical optional attributes are scripts which are executed under certain conditions. Most attributes can be used for most object definitions.

Some attributes are valid only for particular object types, while some object types require that specific attributes be included in their object definitions.

### 5.2.2.3 Object definitions

A *object definition* is a statement in the configuration file, beginning with the keyword `object`, that describes a particular resource in terms that RMS understands. Specifics of an object definition include the following:

- Object type
- Resource name
- Attributes
- Child resources that the particular resource depends upon

After RMS has been installed and verified, the configuration file must be created before RMS can begin monitoring resources. If a resource does not have an object definition in the configuration file, the resource is *not* recognized by RMS.

### 5.2.3 Scripts

An RMS configuration *script* is a shell program or executable that reacts to and/or invokes a change in the state of an RMS resource.

The following states are possible for all RMS resources:

- Faulted
- Offline
- OfflineFault
- Online
- Unknown
- Wait
- Deact
- Inconsistent
- Standby
- Warning

All RMS actions and reactions are executed as scripts. Without scripts, RMS is only capable of monitoring resources, not activating any changes. Scripts are identified as attributes in the object definition for the particular resource, and are run by the base monitor as needed in response to detector-reported state changes. For example, if the state of an RMS network resource changes from *Online* to *Faulted*, the base monitor responds by initiating the fault script listed in the object definition for that resource.

RMS distinguishes between scripts which are designed to change a state (request-triggered scripts) and scripts which represent a reaction to a specific state (state-triggered scripts).

Request-triggered scripts include:

- PreOnlineScript
- PreOfflineScript
- PreCheckScript
- OnlineScript
- OfflineScript
- OfflineDoneScript

State-triggered scripts include:

- PostOnlineScript
- PostOfflineScript
- FaultScript

Your RMS may include additional scripts.

For further information on resource states and scripts, refer to the *RMS Configuration and Administration Guide*.

## 5.2.4 Detectors

A *detector* is a process that monitors the state of a certain type of resource, such as mirrored disks. A detector is started by the base monitor when RMS is started.

Each detector uses an internal table to track information about one or more resources. The detector scans each resource listed in its internal table, obtains the current state of each resource, and then compares the current state to the previous state. If the current state is not the same as the previous state, the detector reports the current state to the base monitor through the detector report queue for that type of resource. Based on the information from the configuration file, the base monitor then determines, what, if any, action is necessary for the particular situation involving that resource.

Figure 25 illustrates the interactions between the base monitor and the RMS detectors.

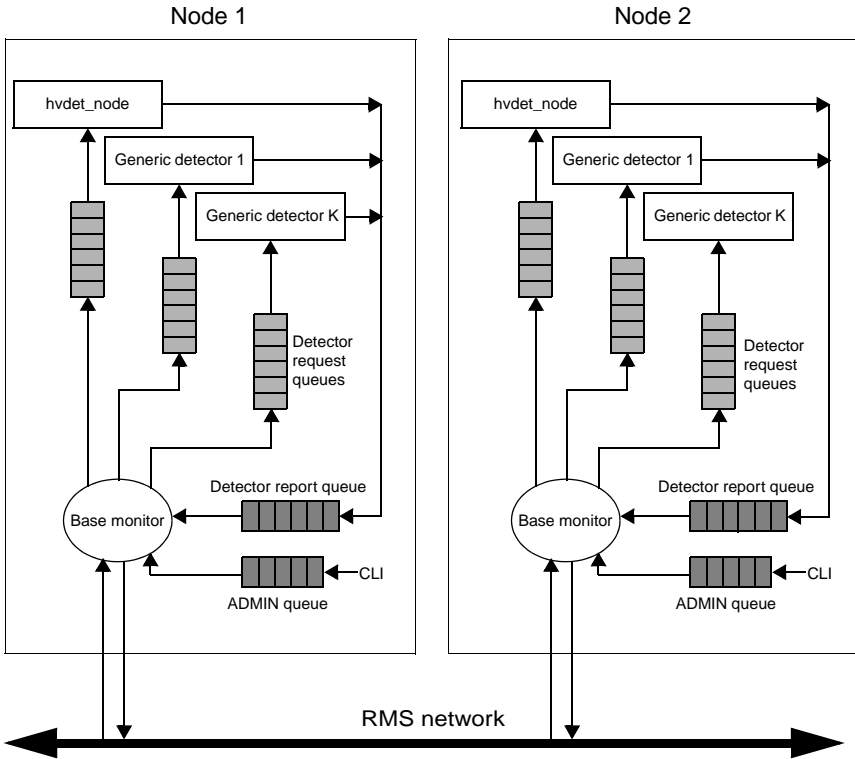


Figure 25: Base monitor interactions

### 5.2.5 RMS environment variables

Environment variables specify values for the base monitor during startup system events. RMS comes with a number of high availability environment variables, such as `HV_AUTOSTART_WAIT`, `RELIANT_PATH`, and so on. The default settings of these RMS environment variables can be adjusted as needed for individual clusters and applications.

## 5.3 RMS administration

You can administer RMS from the command-line interface (CLI) or from the graphical user interface (GUI). The preferred method for RMS administration is the GUI, which is called Cluster Admin.

## 5.4 Customization options

Fault detection and recovery schemes can be customized to fit the needs of each site by modifying configuration files, detectors, and scripts for the resources that are to be monitored by RMS. These modifications should be planned in advance and implemented after RMS installation is completed, and before RMS is declared operational.

### 5.4.1 Generic types and detectors

RMS provides a generic resource type for defining special resources that cannot use the system-level supplied resource types. The generic detector interface allows the definition of up to 64 custom resource types and their detectors.





---

## 6 RMS configuration tools

This chapter introduces the basic concepts of the RMS configuration tools. After a brief overview, it discusses the two main wizard products and some of the functions they provide.

This chapter discusses the following:

- The introduces the two wizard products: RMS Wizard Tools and RMS Wizard Kit.
- The section “RMS Wizard Tools” describes the architecture and functions of the RMS Wizard Tools.
- The section “RMS Wizard Kit” describes the architecture and functions of the RMS Wizard Kit.

### 6.1 RMS configuration overview

Creating an RMS configuration file (see the section “Configuration file”) is a complex task that requires detailed knowledge of both the user applications and the RMS environment.

RMS configuration tools simplify both the configuration and the operation of RMS. RMS configuration tools can manage the system, RMS, and applications.

The RMS configuration tools are divided into the following two main product areas:

- **RMS Wizard Tools**—This contains configuration tool foundations and interface with the RMS base. They simplify the configuration setup and work with RMS in the HA (high availability) cluster environment.
- **RMS Wizard Kit**—RMS Wizard Kit creates optimized HA environments for individual enterprise user applications.

The RMS Wizard Kit is layered onto the configuration tool foundations, which interacts directly with the RMS Base, as is shown in Figure 26.

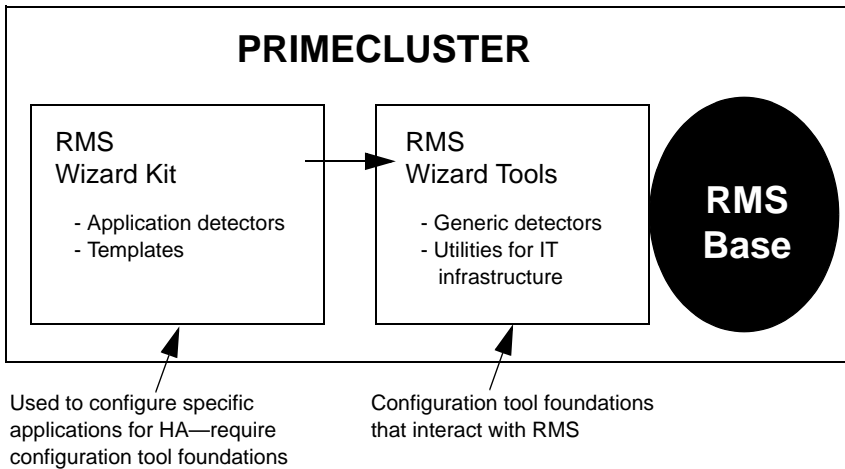


Figure 26: Wizard architecture



For additional information about the RMS Wizard Tools and the RMS Wizard Kit, see the *Reliant Monitor Services (RMS) with Wizard Tools (Solaris, Linux) Configuration and Administration Guide*.

## 6.2 RMS Wizard Tools

The RMS Wizard Tools interface with the RMS base. They simplify the configuration setup and work with RMS in the HA (high availability) cluster environment. Features of the RMS Wizard Tools include:

- Create and distribute complete RMS configuration files
- Prevent or allow configuration changes
- Guide the user through the process of selecting the components of the configuration, and prompts for specific information required by these components
- Present standard resources in a logical sequence to generate correct configuration structures
- Include quality-assured components to monitor and control (detectors and scripts) cluster operations
- Work in conjunction with RMS Wizard Kit to create RMS configurations

To support additional HA configurations, there are some utility wizards offered as part of the configuration tool foundation that manage some system standard resources and applications not covered by a specific application wizard, such as the following:

- Mounting/unmounting file systems
- Installing/uninstalling IP aliases
- Configuring/deconfiguring virtual disks

### Shared-storage applications

Shared-storage application wizards configure RMS for different types of disk storage. For example, you can configure software-based solutions such as GDS or Veritas VxVM, or you can configure hardware solutions like EMC. These wizards can also configure multiple SAN (Storage Area Network) modules, allowing you the capacity to stack modules.

## 6.3 RMS Wizard Kit

The RMS Wizard Kit configures RMS for a unique enterprise application. The RMS Wizard Kit produces a complete RMS configuration for the user, which is then used by the corresponding RMS configuration tool foundation and RMS. These wizards are distributed separately from the configuration tool foundation software.



For information on the availability of specific RMS Wizard Kit, contact your local customer support service.

RMS Wizard Kit allows the administrator to adapt to the user's needs for HA for a specific application. They contain predefined application-specific detectors, rules, and templates for entering user parameters for the application.

Application-specific detectors monitor RMS resources for a particular application. Failure of an RMS resource triggers a user-defined response. See the section “Detectors” for more information on how detectors work within RMS.



---

# 7 SIS

This chapter introduces the basic concepts and components of the Scalable Internet Services (SIS). After a brief overview, it explains how the SIS components function and the services that they provide.

This chapter discusses the following:

- The section “SIS overview” introduces the concepts of the SIS technology.
- The section “SIS design” shows an example of how a SIS cluster might function.
- The section “VIP load-balancing algorithms” details the algorithms available for users of SIS.
- The section “Proxy addresses” describes the uses and capabilities of proxy addresses.
- The section “Private addresses” describes the uses and capabilities of private addresses.
- The section “Failover” discusses the concepts and capabilities of backup nodes.

## 7.1 SIS overview

SIS provides scalable and fault tolerant network services based on the underlying PRIMECLUSTER technology. SIS enables the PRIMECLUSTER to act as a scalable, reliable, and easily managed network server system. The services in a SIS cluster are accessed via one or more Virtual Interface Provider (VIP) addresses, and the cluster appears to clients as a single network server.

Nodes in a SIS cluster share the load for different services. With SIS, users can configure load sharing per service, and they can fine-tune unique application and site needs with a variety of load-balancing algorithms.

SIS eliminates single points of failure and ensures availability as follows:

- If any of the SIS nodes fail, SIS schedules requests around the failed nodes.
- If any of the SIS modules fail, they gracefully recover.
- After a failed node restarts and you have started SIS on it, the node will seamlessly join the cluster to restore maximum performance.
- SIS will recover from NIC failures by routing outgoing packets from a working NIC on a functional node.

## 7.1.1 Features

SIS has the following features:

- Single IP target address for all external users
- Ease of adding nodes and services
- TCP and UDP services can be scheduled on a per-port basis
- Many load-balancing algorithms available
- Seamless handling of node failures, service failures, and component failures
- Flexible backup-node management
- Satellite nodes
- Support for Windows 2000 service nodes
- Proxy addresses for cluster use
- Private communication between nodes
- Software-based solution
- GUI-based configuration and administration interface

The sections that follow discuss the various types of SIS modules.

## 7.1.2 Service nodes

Service nodes offer network services such as web services and directory services. If a service node fails, services are scheduled around it. When a failed service node comes back up, it rejoins the SIS cluster seamlessly.

## 7.1.3 Gateway nodes

There is one gateway node per VIP address. All incoming packets are received by this node and forwarded to the selected service node, depending on the scheduling algorithm for the service. If a gateway node fails, another node seamlessly assumes the role of the failed gateway node.

## 7.1.4 Primary database node

The primary database node keeps the static and dynamic data of the SIS cluster. The static information may include the list of nodes in the SIS cluster, the VIP address and services offered, and the scheduling algorithms. The dynamic information includes the current list of connections and the current status of the SIS cluster.

## 7.1.5 Backup database nodes

A backup database node assumes the role of the primary database node when the primary database node fails for any reason. There can be more than one backup database node. Each backup database node contains the static configuration details, and SIS collects the dynamic configuration items from all the available nodes; therefore, network disruption is kept to a minimum.

### 7.1.6 Satellite nodes

Satellite nodes run a version of SIS that provides better performance while supporting the same network services as the complete SIS package. The major differences between satellite nodes and non-satellite nodes are:

- Satellite nodes do not need to be members of the CF cluster. The satellite version of SIS is available for Windows 2000 Server as well as Solaris and Linux.
- Satellite nodes cannot act as database or gateway nodes.

Figure 27 illustrates a five-node SIS configuration with three SIS core nodes, one Linux satellite node, and one Windows 2000 satellite node.

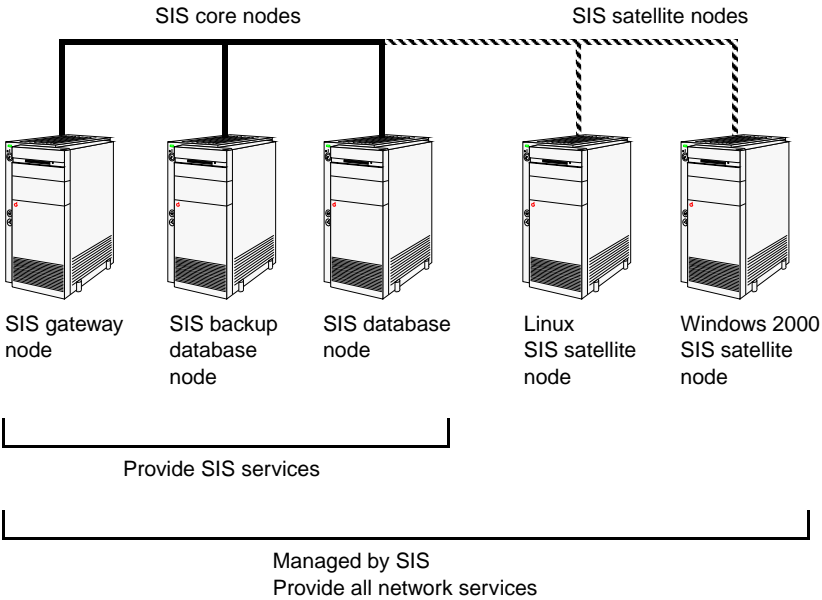


Figure 27: SIS cluster with satellites nodes example



## 7.2 SIS design

In Figure 28, the SIS cluster consists of one gateway node and two service nodes. The steps that follow give an example of how a SIS cluster performs after receiving a request from a client.

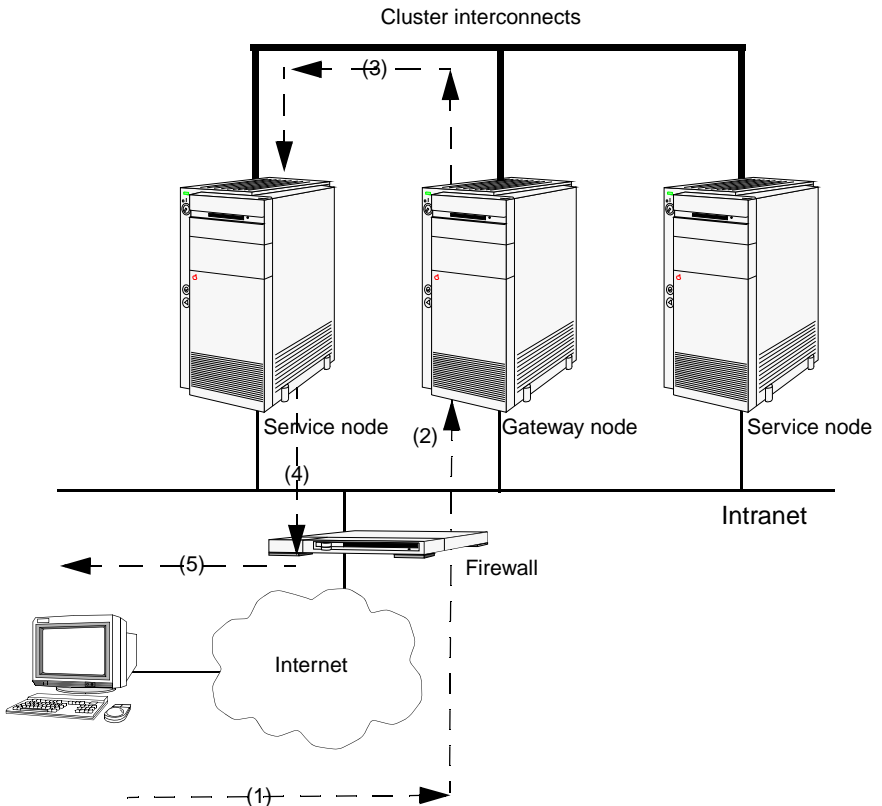


Figure 28: SIS client/server illustration

Referring to Figure 28, the SIS cluster functions as follows:

1. Client requests services through the network, seeing only one IP address.
2. The gateway node receives the initial connection request.

3. The gateway node and the database node choose a service node, which can be a satellite node, based on the scheduling policy and the availability services. Only the initial request requires this scheduling. Subsequent data packets are routed directly to the service node.
4. The service node processes the request and responds directly to the client.
5. The client receives the response from the network.

## 7.3 VIP load-balancing algorithms

SIS provides the following load-balancing algorithms, which can be configured on a per-port basis:

- Keep local—The gateway node answers the request without any processing overhead.
- Client based—The service node is chosen based on the client's IP address.
- System load—The system with the minimum system load is chosen. The system load is calculated by SIS using an internal algorithm.
- Round robin—All available nodes are chosen in a circular way.
- Spill over—If the system load on all primary nodes is equal or greater than the configured threshold, a backup node is chosen to lessen the load.
- Weighted connection count—SIS chooses the system with the least number of connections. Each node may be assigned a weight, and the connection count is normalized according to this weight.

## 7.4 Proxy addresses

Proxy addresses can be defined to assign one or more virtual addresses to a node. Since proxy addresses have failover capabilities, they also provide high availability. The possible uses are as follows:

- Co-hosting multiple addresses to one node
- Assigning external connectivity to nodes that do not have connections to the Internet
- Allocating backup nodes to a node

## 7.5 Private addresses

Private addresses can be configured on each of the available nodes in a SIS cluster. These addresses are commonly used for inter-node communication by means of the interconnects and the IP protocol. Private addresses are secure, fast, and use redundant interconnects to assure high availability.

## 7.6 Failover

One or more nodes can be service nodes for each service. Each of these service nodes can have one or more backup nodes. The backup nodes can be either hot standby or nodes which are doing something else. If a node or nodes fail, then the configured backup nodes replace the failed node or nodes as determined by the configuration file.



## 8 Appendix-Release information

No	VL	Edition	Section	Description
1	4.3A10	June 2011	"Availability"	Added explanations when two or more points fail at the same time.
2	4.3A10	June 2011	"Protecting data integrity" and "PRIME-CLUSTER SF"	Deleted explanations of RPS, RSB, netdump, and diskdump.
3	4.3A10	November 2011	"GDS"	Displayed the figure of "Disk Mirroring."
4	4.3A10	February 2012	"Related documentation" and "RMS Wizard Kit"	Deleted the explanation of RMS Wizards documentation package.
5	4.3A10	February 2012	"Abbreviations"	Deleted Solaris 9 and added Solaris 11 for the supported OSes.
6	4.3A10	February 2012	"Clustering technology overview" and "Virtualization support"	Added the explanation of virtualization support.
7	4.3A10	February 2012	"Protecting data integrity" and "PRIME-CLUSTER SF"	Changed the explanation to correspond to SPARC Enterprise M series.
8	4.3A10	February 2012	"PRIMECLUSTER components"	Deleted the explanation of RMS Wizard Kit.
9	4.3A10	February 2012	"PRIMECLUSTER components" and "CF"	Deleted the explanation of PRIMECLUSTER SNMP.
10	4.3A10	February 2012	"PRIMECLUSTER SF"	Deleted the explanation of SA (Shutdown Agent).
11	4.3A10	February 2012	"PRIMECLUSTER SF"	Changed "RCCU" to "XSCF/ILOM."
12	4.3A10	February 2012	"PRIMECLUSTER SF"	Changed the explanation of the procedure for MA in the event of node error.
13	4.3A10	February 2012	"CF"	Deleted the explanation of SCON (Solaris).

No	VL	Edition	Section	Description
14	4.3A10	February 2012	"RMS configuration tools"	Changed the explanation of RMS Wizard Kit.
15	4.3A10	February 2012	"GFS"	Deleted the explanation of the GFS local file system
16	4.3A10	February 2012	"GFS shared file system" and "Benefits"	Deleted the description "(Solaris only)" because this manual also for Linux.
17	4.3A10	February 2012	"Heartbeats"	Added a condition when the route is marked as DOWN.
18	4.3A10	February 2012	"Glossary"	Deleted the following items: - mirrored pieces (RCVM) - mirrored disks (RCVM)

---

# Glossary

Items in this glossary that apply to specific PRIMECLUSTER products are indicated with the following notation:

- (CF)—Cluster Foundation
- (RMS)—Reliant Monitor Services
- (SIS)—Scalable Internet Services

Some of these products may not be installed on your cluster. See your PRIMECLUSTER sales representative for more information.

## AC

See *Access Client*.

## Access Client

GFS kernel module on each node that communicates with the Meta Data Server and provides simultaneous access to a shared file system.

## activating a configuration (RMS)

Preparing an RMS configuration to be run on a cluster. This involves two major actions: first, the configuration is *generated* on the host where the configuration was created or edited; second, the configuration is *distributed* to all nodes affected by the configuration. The user can activate a configuration using the RMS Wizards, or the CLI.

See also *generating a configuration (RMS)*, *distributing a configuration (RMS)*.

## Administrative LAN

In PRIMECLUSTER configurations, an Administrative LAN is a private local area network (LAN) on which machines such as the System Console and Cluster Console reside. Because normal users do not have access to the Administrative LAN, it provides an extra level of security. The use of an Administrative LAN is optional.

See also *public LAN*.

## API

See *Application Program Interface*.

## Glossary

---

### **application (RMS)**

A resource categorized as a `userApplication` used to group resources into a logical collection.

### **Application Program Interface**

A shared boundary between a service provider and the application that uses that service.

### **application template (RMS)**

A predefined group of object definition value choices used by the RMS Wizard Kit to create object definitions for a specific type of application.

### **attribute (RMS)**

The part of an object definition that specifies how the base monitor acts and reacts for a particular object type during normal operations.

### **automatic power control**

This function is provided by the Enhanced Support Facility (ESF), and it automatically switches the server power on and off.

### **automatic switchover (RMS)**

The procedure by which RMS automatically switches control of a `userApplication` over to another node after specified conditions are detected.

See also *directed switchover (RMS)*, *failover (RMS, SIS)*, *switchover (RMS)*, *symmetrical switchover (RMS)*.

### **availability**

Availability describes the need of most enterprises to operate applications via the Internet 24 hours a day, 7 days a week. The relationship of the actual to the planned usage time determines the availability of a system.

### **base cluster foundation (CF)**

This PRIMECLUSTER module resides on top of the basic OS and provides internal interfaces for the CF (Cluster Foundation) functions that the PRIMECLUSTER services use in the layer above.

See also *Cluster Foundation (CF)*.



**base monitor (RMS)**

The RMS module that maintains the availability of resources. The base monitor is supported by daemons and detectors. Each node being monitored has its own copy of the base monitor.

**Cache Fusion**

The improved interprocess communication interface in Oracle 9i that allows logical disk blocks (buffers) to be cached in the local memory of each node. Thus, instead of having to flush a block to disk when an update is required, the block can be copied to another node by passing a message on the interconnect, thereby removing the physical I/O overhead.

**CCBR**

See *Cluster Configuration Backup and Restore*.

**CF**

See *Cluster Foundation (CF)*.

**child (RMS)**

A resource defined in the configuration file that has at least one parent. A child can have multiple parents, and can either have children itself (making it also a parent) or no children (making it a leaf object).

See also *resource (RMS)*, *object (RMS)*, *parent (RMS)*.

**cluster**

A set of computers that work together as a single computing source. Specifically, a cluster performs a distributed form of parallel computing.

See also *RMS configuration*.

**Cluster Admin**

A Java-based, OS-independent management tool for PRIMECLUSTER products such as CF, RMS, and SIS. Cluster Admin is available from the Web-Based Admin View interface.

See also *Cluster Foundation (CF)*, *Reliant Monitor Services (RMS)*, *Scalable Internet Services (SIS)*, *Web-Based Admin View*.

**Cluster Configuration Backup and Restore**

CCBR provides a simple method to save the current PRIMECLUSTER configuration information of a cluster node. It also provides a method to restore the configuration information.

**Cluster Foundation (CF)**

The set of PRIMECLUSTER modules that provides basic clustering communication services.

See also *base cluster foundation (CF)*.

**cluster interconnect (CF)**

The set of private network connections used exclusively for PRIME-CLUSTER communications.

**Cluster Join Services (CF)**

This PRIMECLUSTER module handles the forming of a new cluster and the addition of nodes.

**configuration file (RMS)**

The RMS configuration file that defines the monitored resources and establishes the interdependencies between them. The default name of this file is `config.us`.

**console**

See *single console*.

**custom detector (RMS)**

See *detector (RMS)*.

**custom type (RMS)**

See *generic type (RMS)*.

**daemon**

A continuous process that performs a specific function repeatedly.

**database node (SIS)**

Nodes that maintain the configuration, dynamic data, and statistics in a SIS configuration.

See also *gateway node (SIS)*, *service node (SIS)*, *Scalable Internet Services (SIS)*.

**detector (RMS)**

A process that monitors the state of a specific object type and reports a change in the resource state to the base monitor.

**directed switchover (RMS)**

The RMS procedure by which an administrator switches control of a userApplication over to another node.

See also *automatic switchover (RMS)*, *failover (RMS, SIS)*, *switchover (RMS)*, *symmetrical switchover (RMS)*.

**distributing a configuration (RMS)**

The process of copying a configuration file and all of its associated scripts and detectors to all nodes affected by the configuration. This is normally done automatically when the configuration is *activated* using the RMS Wizards, or the CLI.

See also *activating a configuration (RMS)*, *generating a configuration (RMS)*.

**DOWN (CF)**

A node state that indicates that the node is unavailable (marked as down). A LEFTCLUSTER node must be marked as DOWN before it can rejoin a cluster.

See also *UP (CF)*, *LEFTCLUSTER (CF)*, *node state (CF)*.

**ENS (CF)**

See *Event Notification Services (CF)*.

**environment variables**

Variables or parameters that are defined globally.

**error detection (RMS)**

The process of detecting an error. For RMS, this includes initiating a log entry, sending a message to a log file, or making an appropriate recovery response.

**Event Notification Services (CF)**

This PRIMECLUSTER module provides an atomic-broadcast facility for events.

**failover (RMS, SIS)**

With SIS, this process switches a failed node to a backup node. With RMS, this process is known as switchover.

See also *automatic switchover (RMS)*, *directed switchover (RMS)*, *switchover (RMS)*, *symmetrical switchover (RMS)*.

### **gateway node (SIS)**

Gateway nodes have an external network interface. All incoming packets are received by this node and forwarded to the selected service node, depending on the scheduling algorithm for the service.

See also *service node (SIS)*, *database node (SIS)*, *Scalable Internet Services (SIS)*.

### **GDS**

See *Global Disk Services*.

### **generating a configuration (RMS)**

The process of creating a single configuration file that can be distributed to all nodes affected by the configuration. This is normally done automatically when the configuration is *activated* using the RMS Wizards, or the CLI.

See also *activating a configuration (RMS)*, *distributing a configuration (RMS)*.

### **GFS**

See *Global File Services*.

### **GLS**

See *Global Link Services*.

### **Global Disk Services**

This optional product provides volume management that improves the availability and manageability of information stored on the disk unit of the Storage Area Network (SAN).

### **Global File Services**

This optional product provides direct, simultaneous accessing of the file system on the shared storage unit from two or more nodes within a cluster.

### **Global Link Services**

This PRIMECLUSTER optional module provides network high availability solutions by multiplying a network route.

**generic type (RMS)**

An object type which has generic properties. A generic type is used to customize RMS for monitoring resources that cannot be assigned to one of the supplied object types.

See also *object type (RMS)*.

**graph (RMS)**

See *system graph (RMS)*.

**graphical user interface**

A computer interface with windows, icons, toolbars, and pull-down menus that is designed to be simpler to use than the command-line interface.

**GUI**

See *graphical user interface*.

**high availability**

A system design philosophy in which redundant resources are employed to avoid single points of failure.

See also *Reliant Monitor Services (RMS)*.

**interconnect (CF)**

See *cluster interconnect (CF)*.

**Internet Protocol address**

A numeric address that can be assigned to computers or applications.

See also *IP aliasing*.

**Internode Communications facility**

This module is the network transport layer for all PRIMECLUSTER internode communications. It interfaces by means of OS-dependent code to the network I/O subsystem and guarantees delivery of messages queued for transmission to the destination node in the same sequential order unless the destination node fails.

**IP address**

See *Internet Protocol address*.

### IP aliasing

This enables several IP addresses (aliases) to be allocated to one physical network interface. With IP aliasing, the user can continue communicating with the same IP address, even though the application is now running on another node.

See also *Internet Protocol address*.

### JOIN (CF)

See *Cluster Join Services (CF)*.

### keyword

A word that has special meaning in a programming language. For example, in the configuration file, the keyword `object` identifies the kind of definition that follows.

### leaf object (RMS)

A bottom object in a system graph. In the configuration file, this object definition is at the beginning of the file. A leaf object does not have children.

### LEFTCLUSTER (CF)

A node state that indicates that the node cannot communicate with other nodes in the cluster. That is, the node has left the cluster. The reason for the intermediate `LEFTCLUSTER` state is to avoid the network partition problem.

See also *UP (CF)*, *DOWN (CF)*, *network partition (CF)*, *node state (CF)*.

### link (RMS)

Designates a child or parent relationship between specific resources.

### local area network

See *public LAN*.

### local node

The node from which a command or process is initiated.

See also *remote node*, *node*.

### log file

The file that contains a record of significant system events or messages. The base monitor, wizards, and detectors can have their own log files.

**MDS**

See *Meta Data Server*.

**message**

A set of data transmitted from one software process to another process, device, or file.

**message queue**

A designated memory area which acts as a holding place for messages.

**Meta Data Server**

GFS daemon that centrally manages the control information of a file system (meta-data).

**mixed model cluster**

A cluster system that is built from different SPARC Enterprise models. For example, one node is SPARC Enterprise M3000, and another node is SPARC Enterprise M4000. The models are divided into the following groups: SPARC T3-1/T3-2/T3-4, SPARC Enterprise T1000/T2000, SPARC Enterprise T5120/T5220/T5140/T5240/T5440, and SPARC Enterprise M3000/M4000/M5000/M8000/M9000.

**MMB**

Abbreviation for Management Board, which is one of the hardware units installed in PRIMEQUEST.

**mount point**

The point in the directory tree where a file system is attached.

**multihosting**

Multiple controllers simultaneously accessing a set of disk drives.

**native operating system**

The part of an operating system that is always active and translates system calls into activities.

**network partition (CF)**

This condition exists when two or more nodes in a cluster cannot communicate over the interconnect; however, with applications still running, the nodes can continue to read and write to a shared device, compromising data integrity.

**node**

A host which is a member of a cluster. A computer node is the same as a computer.

**node state (CF)**

Every node in a cluster maintains a local state for every other node in that cluster. The node state of every node in the cluster must be either UP, DOWN, or LEFTCLUSTER.

See also *UP (CF)*, *DOWN (CF)*, *LEFTCLUSTER (CF)*.

**object (RMS)**

In the configuration file or a system graph, this is a representation of a physical or virtual resource.

See also *leaf object (RMS)*, *object definition (RMS)*, *object type (RMS)*.

**object definition (RMS)**

An entry in the configuration file that identifies a resource to be monitored by RMS. Attributes included in the definition specify properties of the corresponding resource. The keyword associated with an object definition is `object`.

See also *attribute (RMS)*, *object type (RMS)*.

**object type (RMS)**

A category of similar resources monitored as a group, such as disk drives. Each object type has specific properties, or attributes, which limit or define what monitoring or action can occur. When a resource is associated with a particular object type, attributes associated with that object type are applied to the resource.

See also *generic type (RMS)*.

**online maintenance**

The capability of adding, removing, replacing, or recovering devices without shutting or powering off the node.

**operating system dependent (CF)**

This module provides an interface between the native operating system and the abstract, OS-independent interface that all PRIMECLUSTER modules depend upon.



**Oracle Real Application Clusters (RAC)**

Oracle RAC allows access to all data in a database to users and applications in a clustered or MPP (massively parallel processing) platform. Formerly known as Oracle Parallel Server (OPS).

**OSD (CF)**

See *operating system dependent (CF)*.

**parent (RMS)**

An object in the configuration file or system graph that has at least one child.

See also *child (RMS)*, *configuration file (RMS)*, *system graph (RMS)*.

**primary node (RMS)**

The default node on which a user application comes online when RMS is started. This is always the nodename of the first child listed in the `userApplication` object definition.

**PRIMECLUSTER services (CF)**

Service modules that provide services and internal interfaces for clustered applications.

**private network addresses**

Private network addresses are a reserved range of IP addresses specified by the Internet Assigned Numbers Authority. They may be used internally by any organization but, because different organizations can use the same addresses, they should never be made visible to the public internet.

**private resource (RMS)**

A resource accessible only by a single node and not accessible to other RMS nodes.

See also *resource (RMS)*, *shared resource*.

**public LAN**

The local area network (LAN) by which normal users access a machine.

See also *Administrative LAN*.

**queue**

See *message queue*.

### **RCCU**

Abbreviation for Remote Console Connection Unit.

See also *remote console connection unit*.

### **redundancy**

This is the capability of one object to assume the resource load of any other object in a cluster, and the capability of RAID hardware and/or RAID software to replicate data stored on secondary storage devices.

### **Reliant Monitor Services (RMS)**

The package that maintains high availability of user-specified resources by providing monitoring and switchover capabilities.

### **remote console connection unit**

Device that converts an RS232C interface and a LAN interface. This device allows another device (personal computer) that is connected to the LAN to use the TTY console functions through the Telnet function.

### **remote node**

A node that is accessed through a LAN or telecommunications line.

See also *local node*, *node*.

### **reporting message (RMS)**

A message that a detector uses to report the state of a particular resource to the base monitor.

### **resource (RMS)**

A hardware or software element (private or shared) that provides a function, such as a mirrored disk, mirrored disk pieces, or a database server. A local resource is monitored only by the local node.

See also *private resource (RMS)*, *shared resource*.

### **resource definition (RMS)**

See *object definition (RMS)*.

### **resource label (RMS)**

The name of the resource as displayed in a system graph.

### **resource state (RMS)**

Current state of a resource.

**RMS**

See *Reliant Monitor Services (RMS)*.

**RMS commands**

Commands that enable RMS resources to be administered from the command line.

**RMS configuration**

A configuration made up of two or more nodes connected to shared resources. Each node has its own copy of operating system and RMS software, as well as its own applications.

**RMS Wizard Kit**

RMS configuration products that have been designed for specific applications. Each component of the Wizard Kit includes customized default settings, subapplications, detectors, and scripts. These application wizards also tailor the Wizard Tools interface to provide controls for the additional features.

See also *RMS Wizard Tools, Reliant Monitor Services (RMS)*.

**RMS Wizard Tools**

A software package composed of various configuration and administration tools used to create and manage applications in an RMS configuration.

See also *RMS Wizard Kit, Reliant Monitor Services (RMS)*.

**SAN**

See *Storage Area Network*.

**Scalable Internet Services (SIS)**

Scalable Internet Services is a TCP connection load balancer, and dynamically balances network access loads across cluster nodes while maintaining normal client/server sessions for each connection.

**scalability**

The ability of a computing system to dynamically handle any increase in work load. Scalability is especially important for Internet-based applications where growth caused by Internet usage presents a scalable challenge.

### SCON

See *single console*.

### script (RMS)

A shell program executed by the base monitor in response to a state transition in a resource. The script may cause the state of a resource to change.

### service node (SIS)

Service nodes provide one or more TCP services (such as FTP, Telnet, and HTTP) and receive client requests forwarded by the gateway nodes.

See also *database node (SIS)*, *gateway node (SIS)*, *Scalable Internet Services (SIS)*.

### shared resource

A resource, such as a disk drive, that is accessible to more than one node.

See also *private resource (RMS)*, *resource (RMS)*.

### single console

The workstation that acts as the single point of administration for nodes being monitored by RMS. The single console software, SCON, is run from the single console.

### single-node cluster

An operation mode of a cluster system consisting of one node.

### SIS

See *Scalable Internet Services (SIS)*.

### state

See *resource state (RMS)*.

### Storage Area Network

The high-speed network that connects multiple, external storage units and storage units with multiple computers. The connections are generally fiber channels.

**switching mode**

LAN duplexing mode presented by GLS.

There is a total of five switching mode types: fast switching mode, NIC switching mode, GS/SURE linkage mode, multipath mode, and multilink Ethernet mode.

**switchover (RMS)**

The process by which RMS switches control of a userApplication over from one monitored node to another.

See also *automatic switchover (RMS)*, *directed switchover (RMS)*, *failover (RMS, SIS)*, *symmetrical switchover (RMS)*.

**symmetrical switchover (RMS)**

This means that every RMS node is able to take on resources from any other RMS node.

See also *automatic switchover (RMS)*, *directed switchover (RMS)*, *failover (RMS, SIS)*, *switchover (RMS)*.

**synchronized power control**

When the power of one node is turned in the cluster system, this function turns on all other powered-off nodes and disk array unit that are connected to nodes through RCI cables.

**system disk (GDS)**

Disk on which the active operating system is installed. System disk refers to the entire disk that contains the slices that are currently operating as one of the following file systems (or the swap area):

For Solaris: /, /usr, /var, or swap area

For Linux: /, /usr, /var, /boot, /boot/efi, or swap area

**system graph (RMS)**

A visual representation (a map) of monitored resources used to develop or interpret the configuration file.

See also *configuration file (RMS)*.

**template**

See *application template (RMS)*.

**type**

See *object type (RMS)*.

## Glossary

---

### **UP** (CF)

A node state that indicates that the node can communicate with other nodes in the cluster.

See also *DOWN (CF)*, *LEFTCLUSTER (CF)*, *node state (CF)*.

### **Web-Based Admin View**

A Java-based, OS-independent interface to PRIMECLUSTER management components.

See also *Cluster Admin*.

### **wizard** (RMS)

An interactive software tool that creates a specific type of application using pretested object definitions. An enabler is a type of wizard.

### **Wizard Kit** (RMS)

See *RMS Wizard Kit*.

### **Wizard Tools** (RMS)

See *RMS Wizard Tools*.

---

# Abbreviations

**AC**

Access Client

**API**

application program interface

**bm**

base monitor

**CCBR**

Cluster Configuration Backup/Restore

**CDL**

Configuration Definition Language

**CF**

Cluster Foundation or Cluster Framework

**CIM**

Cluster Integrity Monitor

**CIP**

Cluster Interconnect Protocol

**CLI**

command-line interface

**CLM**

Cluster Manager

**CRM**

Cluster Resource Management

**DLPI**

Data Link Provider Interface

**ENS**

Event Notification Services

## Abbreviations

---

<b>GDS</b>	Global Disk Services
<b>GFS</b>	Global File Services
<b>GLS</b>	Global Link Services
<b>GUI</b>	graphical user interface
<b>HA</b>	high availability
<b>ICF</b>	Internode Communication Facility
<b>I/O</b>	input/output
<b>JOIN</b>	cluster join services module
<b>LAN</b>	local area network
<b>MDS</b>	Meta Data Server
<b>MIB</b>	Management Information Base
<b>MIPC</b>	Mesh Interprocessor Communication
<b>MMB</b>	Management Board
<b>NIC</b>	network interface card



<b>NSM</b>	Node State Monitor
<b>OSD</b>	operating system dependent
<b>PAS</b>	Parallel Application Services
<b>RCCU</b>	Remote Console Connection Unit
<b>RCI</b>	Remote Cabinet Interface
<b>RMS</b>	Reliant Monitor Services
<b>SA</b>	Shutdown Agent
<b>SAN</b>	Storage Area Network
<b>SCON</b>	single console software
<b>SD</b>	Shutdown Daemon
<b>SF</b>	Shutdown Facility
<b>SIS</b>	Scalable Internet Services
<b>VIP</b>	Virtual Interface Provider
<b>XSCF</b>	eXtended System Control Facility



---

# Figures

Figure 1:	Typical two-node cluster . . . . .	10
Figure 2:	Cluster partition in two-node cluster . . . . .	14
Figure 3:	Existing cluster system and cluster system in a virtual environment (For example, KVM environment) . . . . .	21
Figure 4:	Cluster system in a virtual environment (For example, KVM/Xen environment) . . . . .	21
Figure 5:	Cluster system in a virtual environment (For example, VMware environment) . . . . .	22
Figure 6:	Cluster system in a virtual environment (For example, Oracle VM Server for SPARC environment) . . . . .	22
Figure 7:	Cluster system in a virtual environment (For example, Oracle Solaris Zones environment) . . . . .	23
Figure 8:	Diagram of a typical PRIMECLUSTER setup . . . . .	26
Figure 9:	PRIMECLUSTER framework overview . . . . .	27
Figure 10:	MA normal operation . . . . .	36
Figure 11:	MA operation in the event of node failure . . . . .	37
Figure 12:	Node recovery . . . . .	38
Figure 13:	State confirmation by SA_icmp if the other node goes down	43
Figure 14:	State confirmation by SA_icmp if the cluster interconnect fails	43
Figure 15:	RMS resource monitoring . . . . .	45
Figure 16:	Example of a SIS cluster . . . . .	47
Figure 17:	Disk Mirroring . . . . .	49
Figure 18:	SAN (Storage Area Network) . . . . .	50
Figure 19:	Operation continuation when a node is down . . . . .	53
Figure 20:	NIC switching mode . . . . .	55

## Figures

---

Figure 21:	Connecting multiple switches . . . . .	60
Figure 22:	Typical four-node cluster . . . . .	61
Figure 23:	RMS cluster on Solaris . . . . .	70
Figure 24:	Configuration file interdependencies . . . . .	74
Figure 25:	Base monitor interactions . . . . .	78
Figure 26:	Wizard architecture . . . . .	82
Figure 27:	SIS cluster with satellites nodes example . . . . .	88
Figure 28:	SIS client/server illustration . . . . .	89

---

# Index

## A

- administering
  - Cluster Admin 33
  - RMS 79
- allowing configuration changes 82
- applications
  - monitored resource 44
  - PAS 47
- assigning external connectivity 90
- attributes
  - mandatory 75
  - node 75
  - object properties 71
  - optional 75
- availability 28
  - design feature 29
  - high 11

## B

- backup SIS database nodes 87
- bandwidth, interconnect 63
- base monitor
  - description 73
  - for central monitoring 73

## C

- CF
  - See* Cluster Foundation
- CF/IP 59
- client-based algorithm 90
- cluster
  - architecture 25
  - partition 13
  - SIS 46
- Cluster Admin 30, 33
  - CF 33
  - RMS 79
- Cluster Foundation 30, 31
  - Cluster Admin 33
  - driver 31
  - heartbeats 65
- cluster interconnect 12, 57

- properties 63
- reliability 66
- sustained outage 57
- See also* interconnects

- clusters 9
  - parallel databases 47
  - SIS 85
- configuration
  - allowing changes 82
  - copying 82
  - creating 82
  - file 73, 74
  - preventing changes 82
  - scripts 76
  - supporting additional 83
  - tools 30, 45, 46, 74, 81, 82
  - wizards 30, 45, 46, 74, 81, 82
- configuring
  - Cluster Admin 33
  - hub 58
- copying complete configuration 82
- creating RMS configuration 82
- customizing RMS 79

## D

- data integrity, protecting 12, 29
- Data Link Provider Interface 66
- database nodes, SIS 87
  - and satellite nodes 88
- detectors 77
  - application specific 83
  - generic 79
- device and network access 26
- diagnostics services 33
- distributed
  - computing 9
  - environment 19
- DLPI
  - See* Data Link Provider Interface
- documentation, related 2
- DOWN, state 62

## Index

---

- drivers
  - CF 31
  - network device 62
  - pseudo device 31
- duplexed network interface cards 55
- E**
- ENS
  - See* Event Notification Services
- environment variables 78
- Ethernets
  - error rate 66
  - fast 62
  - Gigabit 62
  - network availability 58
- Event Notification Services 32
- F**
- failover 72, 91
- features
  - availability 29
  - data integrity 29
  - scalability 29
- Fibre Channel controller 62
- file systems
  - high availability 52
  - mount points 71
  - network 44
  - update log 52
- files
  - cluster access 12
  - clusterwide system 29
  - configuration 73, 75, 79
  - custom 79
  - RMS software 73
  - same source 31
  - user-specified configuration 72
- G**
- gateway nodes, SIS 87
  - and satellite nodes 88
- Gigabit Ethernet 62
- Global Disk Services 30
- Global File Services 30
- Global Link Services 30
  - NIC switching mode 55
  - redundant NICs 54
- GLS
  - See* Global Link Services
- guaranteed data integrity 28, 29
- H**
- heartbeats 57
  - request failures 62
  - requests 57, 65
  - responses 57
  - RMS 12
- high availability 11
  - clustering 9
  - file systems 52
  - multiple NICs 54
- hub, configure 58
- I**
- ICF
  - See* Internode Communication Facility
- installing aliases 83
- interconnects
  - bandwidth 63
  - fast switching mode 55
  - latency 65
  - NIC switching mode 55
  - non-symmetric 62
  - protocol 59
  - See also* cluster interconnect
- interface for cluster administration 30
- Internode Communication Facility 31, 59, 71
- IP aliasing 72
- J**
- join services (JOIN) 32
- K**
- keep local algorithm 90

**L**

- latency 65
- LEFTCLUSTER, state 62
- link recovery 62
- load-balancing algorithms 90

**M**

- manual switchovers 72
- memory allocations 26
- modular components 30
- modularity 28
- monitoring
  - agent 18, 34
  - applications 12
  - components 12
- multiple network interface cards 54

**N**

- network interface cards
  - configuring 58
  - multiple 54
  - speeds 58
- networks
  - availability 58
  - configuring NIC 58
  - file systems 44
  - interfaces 44
- NIC switching mode 55
- NICs
  - See* network interface cards
- nodes
  - assigning weight 90
  - backup 91
  - client IP address 90
  - communication 91
  - definition 75
  - monitored resource 44
  - multiple proxy addresses 90
  - satellite, SIS 88
  - service 87
  - SIS database 87, 88
  - SIS database, backup 87
  - SIS gateway 87, 88, 89

type 75

- non-symmetric interconnects 62

**O**

- operating system dependant 31
- operations 33
- OSD
  - See* operating system dependant
- OS-independent interface 31

**P**

- Parallel Application Services 30
  - description 47
  - modularity 28
- PAS
  - See* Parallel Application Services
- path failure protection 54
- platform independence 28
- preventing configuration changes 82
- primary SIS database nodes 87
- PRIMECLUSTER modules 30
- protecting data integrity 12, 29
- proxy addresses 90
- pseudo device driver 31

**R**

- redundancy scheme 70
- redundant
  - cluster interconnect 60
  - components 11
- related documentation 2
- reliability, interconnects 66
- resources
  - automatic switchover 72
  - detectors 77, 79
  - heartbeats 65
  - high availability 70
  - manual switchover 72
  - monitoring 76
  - states 76
  - switchover 71
- RMS 28, 30, 44
  - administering 79
  - configuration script 76

- configuration tool types 81
- configuration tools 19, 30, 45, 46,  
74, 81, 82, 83
- detectors 83
- environment variables 78
- HA manager 12
- overview 73, 78
- RMS Application Wizards 46
- RMS Wizard Kit 81, 82
- RMS Wizard Tools 30, 45, 74, 81,  
82
- utility wizards 83
- round robin algorithm 90
- routers
  - configuring 58
  - latencies 66

## S

### SAN

*See* Storage Area Network

SAP R/3 19

satellite nodes, SIS 88

scalability 9, 28

- basic types of applications 19

- network services 85

- quantity of nodes 29

- SIS 85

scalable applications 20

Scalable Internet Services 28, 46

- architecture 30

- description 46

- network services 85

scripts

- optional attribute 75

- RMS configuration 76

service nodes, SIS 87, 91

services

- parallel application 47

- scalable Internet 30

### SIS

- backup database nodes 87

- gateway nodes 87

- primary database nodes 87

- satellite nodes 88

- service nodes 87
- spill over algorithm 90
- spin locks 26
- states
  - controlling changes 73
  - detectors 77
  - DOWN 62
  - LEFTCLUSTER 62
  - monitoring changes 73
  - Offline 72
  - resources 76
  - scripts 77
  - UP 61
- Storage Area Network
  - high availability 11
  - wizards 83
- sustained outage 57
- switchover 72
- synchronizations 26
- system load algorithm 90

## U

uninstalling aliases 83

UP, state 61

update log 52

utility wizards 83

## W

Web-Based Admin View 30

Weighted Connection Count  
algorithm 90

Windows 2000 Server, as satellite  
node 88