Sun Cluster Concepts Guide for Solaris OS

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 817–6537–10
September 2004, Revision A

040812@9495

Contents

Preface

Sun™ Cluster Concepts Guide for Solaris OS contains conceptual and reference information for the SunPlex™ system on both SPARC™ and x86 based systems.

Note – In this document, the term “x86” refers to the Intel 32-bit family of microprocessor chips and compatible microprocessor chips made by AMD.

The SunPlex system includes all hardware and software components that make up Sun’s cluster solution.

This document is intended for experienced system administrators who are trained on Sun Cluster software. Do not use this document as a planning or presales guide. You should have already determined your system requirements and purchased the appropriate equipment and software before reading this document.

To understand the concepts described in this book, you should have knowledge of the Solaris™ operating environment and expertise with volume manager software used with the SunPlex system.

Note – Sun Cluster software runs on two platforms, SPARC and x86. The information in this document pertains to both platforms unless otherwise specified in a special chapter, section, note, bulleted item, figure, table, or example.

Typographic Conventions

The following table describes the typographic changes that are used in this book.

TABLE P–1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
AaBbCc123	Command-line placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. These are called *class* options. Do *not* save the file. (Emphasis sometimes appears in bold online.)

Shell Prompts in Command Examples

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell prompt	`machine_name%`
C shell superuser prompt	`machine_name#`
Bourne shell and Korn shell prompt	`$`

TABLE P–2 Shell Prompts *(Continued)*

Shell	Prompt
Bourne shell and Korn shell superuser prompt	#

Related Documentation

Information about related Sun Cluster topics is available in the documentation that is listed in the following table. All Sun Cluster documentation is available at `http://docs.sun.com`.

Topic	Documentation
Overview	*Sun Cluster Overview for Solaris OS*
Concepts	*Sun Cluster Concepts Guide for Solaris OS*
Hardware installation and administration	*Sun Cluster 3.x Hardware Administration Manual for Solaris OS* Individual hardware administration guides
Software installation	*Sun Cluster Software Installation Guide for Solaris OS*
Data service installation and administration	*Sun Cluster Data Services Planning and Administration Guide for Solaris OS* Individual data service guides
Data service development	*Sun Cluster Data Services Developer's Guide for Solaris OS*
System administration	*Sun Cluster System Administration Guide for Solaris OS*
Error messages	*Sun Cluster Error Messages Guide for Solaris OS*
Command and function references	*Sun Cluster Reference Manual for Solaris OS*

For a complete list of Sun Cluster documentation, see the release notes for your release of Sun Cluster software at `http://docs.sun.com`.

Accessing Sun Documentation Online

The docs.sun.com[SM] Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is `http://docs.sun.com`.

Ordering Sun Documentation

Sun Microsystems offers select product documentation in print. For a list of documents and how to order them, see "Buy printed documentation" at `http://docs.sun.com`.

Getting Help

If you have problems installing or using the SunPlex system, contact your service provider and provide the following information:

- Your name and email address (if available)
- Your company name, address, and phone number
- The model and serial numbers of your systems
- The release number of the operating environment (for example, Solaris 9)
- The release number of the Sun Cluster software (for example, 3.1 4/04)

Use the following commands to gather information about each node on your system for your service provider.

Command	Function
`prtconf -v`	Displays the size of the system memory and reports information about peripheral devices
`psrinfo -v`	Displays information about processors
`showrev -p`	Reports which patches are installed
SPARC: `prtdiag -v`	Displays system diagnostic information

Command	Function
`scinstall -pv`	Displays Sun Cluster software release and package version information
`scstat`	Provides a snapshot of the cluster status
`scconf –p`	Lists cluster configuration information
`scrgadm -p`	Displays information on installed resources, resource groups, and resource types

Also have available the contents of the `/var/adm/messages` file.

CHAPTER 1

Introduction and Overview

The SunPlex system is an integrated hardware and Sun Cluster software solution that is used to create highly available and scalable services.

Sun Cluster Concepts Guide for Solaris OS provides the conceptual information needed by the primary audience for SunPlex documentation. This audience includes

- Service providers who install and service cluster hardware
- System administrators who install, configure, and administer Sun Cluster software
- Application developers who develop failover and scalable services for applications not currently included with the Sun Cluster product

This book works with the rest of the SunPlex documentation set to provide a complete view of the SunPlex system.

This chapter

- Provides an introduction and high-level overview of the SunPlex system
- Describes the several viewpoints of the SunPlex audience
- Identifies key concepts you need to understand before working with the SunPlex system
- Maps key concepts to the SunPlex documentation that includes procedures and related information
- Maps cluster-related tasks to the documentation containing procedures used to accomplish those tasks

Introduction to the SunPlex System

The SunPlex system extends the Solaris operating environment into a cluster operating system. A cluster, or plex, is a collection of loosely coupled computing nodes that provides a single client view of network services or applications, including databases, web services, and file services.

Each cluster node is a standalone server that runs its own processes. These processes communicate with one another to form what looks like (to a network client) a single system that cooperatively provides applications, system resources, and data to users.

A cluster offers several advantages over traditional single-server systems. These advantages include support for failover and scalable services, capacity for modular growth, and low entry price compared to traditional hardware fault-tolerant systems.

The goals of the SunPlex system are:

- Reduce or eliminate system downtime because of software or hardware failure
- Ensure availability of data and applications to end users, regardless of the kind of failure that would normally take down a single-server system
- Increase application throughput by enabling services to scale to additional processors by adding nodes to the cluster
- Provide enhanced availability of the system by enabling you to perform maintenance without shutting down the entire cluster

For more information about fault-tolerance and high availability, see "Making Applications Highly Available With Sun Cluster" in *Sun Cluster Overview for Solaris OS*.

Refer to "High Availability FAQs" on page 87 for questions and answers on high availability.

Three Viewpoints of the SunPlex System

This section describes three different viewpoints on the SunPlex system and the key concepts and documentation relevant to each viewpoint. These viewpoints come from:

- Hardware installation and service personnel
- System administrators
- Application programmers

Hardware Installation and Service Viewpoint

To hardware service professionals, the SunPlex system looks like a collection of off-the-shelf hardware that includes servers, networks, and storage. These components are all cabled together so that every component has a backup and no single point of failure exists.

Key Concepts – Hardware

Hardware service people need to understand the following cluster concepts.

- Cluster hardware configurations and cabling
- Installing and servicing (adding, removing, replacing):
  - Network interface components (adapters, junctions, cables)
  - Disk interface cards
  - Disk arrays
  - Disk drives
  - The administrative console and the console access device
- Setting up the administrative console and console access device

Suggested Hardware Conceptual References

The following sections contain material relevant to the preceding key concepts:

- "Cluster Nodes" on page 20
- "Multihost Devices" on page 22
- "Local Disks" on page 24
- "Cluster Interconnect" on page 24
- "Public Network Interfaces" on page 25
- "Client Systems" on page 25
- "Administrative Console" on page 26
- "Console Access Devices" on page 26
- "SPARC: Clustered Pairs Topology" on page 27
- "SPARC: N+1 (Star) Topology" on page 29

Relevant SunPlex Documentation

The following SunPlex document includes procedures and information associated with hardware service concepts:

Sun Cluster 3.x Hardware Administration Manual for Solaris OS

System Administrator Viewpoint

To the system administrator, the SunPlex system looks like a set of servers (nodes) cabled together, sharing storage devices. The system administrator sees:

- Specialized cluster software integrated with Solaris software to monitor the connectivity between cluster nodes
- Specialized software that monitors the health of user application programs running on the cluster nodes
- Volume management software that sets up and administers disks
- Specialized cluster software that enables all nodes to access all storage devices, even those not directly connected to disks
- Specialized cluster software that enables files to appear on every node as though they were locally attached to that node

Key Concepts – System Administration

System administrators need to understand the following concepts and processes:

- The interaction between the hardware and software components
- The general flow of how to install and configure the cluster including:
  - Installing the Solaris operating environment
  - Installing and configuring Sun Cluster software
  - Installing and configuring a volume manager
  - Installing and configuring application software to be cluster ready
  - Installing and configuring Sun Cluster data service software
- Cluster administrative procedures for adding, removing, replacing, and servicing cluster hardware and software components
- Configuration modifications to improve performance

Suggested System Administrator Conceptual References

The following sections contain material relevant to the preceding key concepts:

- "Administrative Interfaces" on page 33
- "Cluster Time" on page 34
- "High-Availability Framework" on page 35
- "Global Devices" on page 37
- "Disk Device Groups" on page 38
- "Global Namespace" on page 42
- "Cluster File Systems" on page 43
- "Disk-Path Monitoring" on page 46
- "About Failure Fencing" on page 51
- "Data Services" on page 60

Relevant SunPlex Documentation – System Administrator

The following SunPlex documents include procedures and information associated with the system administration concepts:

- *Sun Cluster Software Installation Guide for Solaris OS*
- *Sun Cluster System Administration Guide for Solaris OS*
- *Sun Cluster Error Messages Guide for Solaris OS*
- *Sun Cluster 3.1 9/04 Release Notes for Solaris OS*
- *Sun Cluster 3.x Release Notes Supplement*

Application Programmer Viewpoint

The SunPlex system provides *data services* for such applications as Oracle (on SPARC based systems), NFS, DNS, Sun™ Java System Web Server (formerly Sun Java System Web Server), Apache Web Server (on SPARC based systems), and Sun Java System Directory Server (formerly Sun Java System Directory Server). Data services are created by configuring an off-the-shelf applications to run under control of the Sun Cluster software. The Sun Cluster software provides configuration files and management methods that start, stop, and monitor the applications. If you need to create a new failover or scalable service, you can use the SunPlex Application Programming Interface (API) and the Data Service Enabling Technologies API (DSET API) to develop the necessary configuration files and management methods that enable its application to run as a data service on the cluster.

Key Concepts – Application Programmer

Application programmers need to understand the following:

- The characteristics of their application to determine whether it can be made to run as a failover or scalable data service.
- The Sun Cluster API, DSET API, and the "generic" data service. Programmers need to determine which tool is most suitable for them to use to write programs or scripts to configure their application for the cluster environment.

Suggested Application Programmer Conceptual References

The following sections contain material relevant to the preceding key concepts:

- "Data Services" on page 60
- "Resources, Resource Groups, and Resource Types" on page 70
- Chapter 4

Relevant SunPlex Documentation – Application Programmer

The following SunPlex documents include procedures and information associated with the application programmer concepts:

- *Sun Cluster Data Services Developer's Guide for Solaris OS*

- *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

SunPlex System Task

All SunPlex system tasks require some conceptual background. The following table provides a high-level view of the tasks and the documentation that describes task steps. The concepts sections in this book describe how the concepts map to these tasks.

TABLE 1–1 Task Map: Mapping User Tasks to Documentation

To Do This Task...	Use This Documentation...
Install cluster hardware	*Sun Cluster 3.x Hardware Administration Manual for Solaris OS*
Install Solaris software on the cluster	*Sun Cluster Software Installation Guide for Solaris OS*
SPARC: Install Sun™ Management Center software	*Sun Cluster Software Installation Guide for Solaris OS*
Install and configure Sun Cluster software	*Sun Cluster Software Installation Guide for Solaris OS*
Install and configure volume management software	*Sun Cluster Software Installation Guide for Solaris OS* Your volume management documentation
Install and configure Sun Cluster data services	*Sun Cluster Data Services Planning and Administration Guide for Solaris OS*
Service cluster hardware	*Sun Cluster 3.x Hardware Administration Manual for Solaris OS*
Administer Sun Cluster software	*Sun Cluster System Administration Guide for Solaris OS*
Administer volume management software	*Sun Cluster System Administration Guide for Solaris OS* and your volume management documentation
Administer application software	Your application documentation
Problem identification and suggested user actions	*Sun Cluster Error Messages Guide for Solaris OS*
Create a new data service	*Sun Cluster Data Services Developer's Guide for Solaris OS*

CHAPTER 2

Key Concepts – Hardware Service Providers

This chapter describes the key concepts related to the hardware components of a SunPlex system configuration. The topics covered include:

The SunPlex System Hardware and Software Components

This information is directed primarily toward hardware service providers. These concepts can help service providers understand the relationships between the hardware components before they install, configure, or service cluster hardware. Cluster system administrators might also find this information useful as background to installing, configuring, and administering cluster software.

A cluster is composed of several hardware components including:

- Cluster nodes with local disks (unshared)
- Multihost storage (disks shared between nodes)
- Removable media (tapes and CD-ROM)

- Cluster interconnect
- Public network interfaces
- Client systems
- Administrative console
- Console access devices

The SunPlex system enables you to combine these components into a variety of configurations, described in "SPARC: Sun Cluster Topology Examples" on page 27.

For an illustration of a sample two-node cluster configuration, see "Sun Cluster Hardware Environment" in *Sun Cluster Overview for Solaris OS*.

Cluster Nodes

A cluster node is a machine running both the Solaris operating environment and Sun Cluster software, and is either a current member of the cluster (a *cluster member*), or a potential member.

SPARC: The Sun Cluster software enables you to have from two to eight nodes in a cluster. See "SPARC: Sun Cluster Topology Examples" on page 27 for the supported node configurations.

x86: The Sun Cluster software enables you to have two nodes in a cluster. See "x86: Sun Cluster Topology Examples" on page 31 for the supported node configurations.

Cluster nodes are generally attached to one or more multihost devices. Nodes not attached to multihost devices use the cluster file system to access the multihost devices. For example, one scalable services configuration allows nodes to service requests without being directly attached to multihost devices.

In addition, nodes in parallel database configurations share concurrent access to all the disks. See "Multihost Devices" on page 22 and Chapter 3 for more information on parallel database configurations.

All nodes in the cluster are grouped under a common name—the cluster name—which is used for accessing and managing the cluster.

Public network adapters attach nodes to the public networks, providing client access to the cluster.

Cluster members communicate with the other nodes in the cluster through one or more physically independent networks. This set of physically independent networks is referred to as the *cluster interconnect*.

Every node in the cluster is aware when another node joins or leaves the cluster. Additionally, every node in the cluster is aware of the resources that are running locally as well as the resources that are running on the other cluster nodes.

Nodes in the same cluster should have similar processing, memory, and I/O capability to enable failover to occur without significant degradation in performance. Because of the possibility of failover, every node must have enough excess capacity to take on the workload of all nodes for which they are a backup or secondary.

Each node boots its own individual root (/) file system.

Software Components for Cluster Hardware Members

To function as a cluster member, the following software must be installed:

- Solaris operating environment
- Sun Cluster software
- Data service application
- Volume management (Solaris Volume Manager™ or VERITAS Volume Manager)

  An exception is a configuration that uses hardware redundant array of independent disks (RAID). This configuration may not require a software volume manager such as Solaris Volume Manager or VERITAS Volume Manager.
- See the *Sun Cluster Software Installation Guide for Solaris OS* for information on how to install the Solaris operating environment, Sun Cluster, and volume management software.
- See the *Sun Cluster Data Services Planning and Administration Guide for Solaris OS* for information on how to install and configure data services.
- See Chapter 3 for conceptual information on the preceding software components.

The following figure provides a high-level view of the software components that work together to create the Sun Cluster software environment.

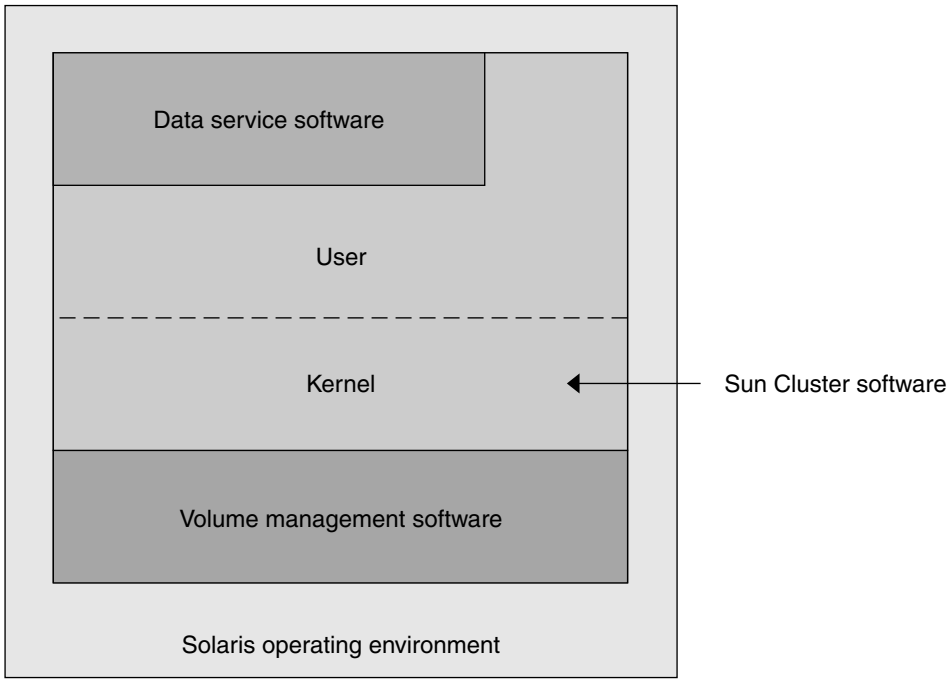


FIGURE 2–1 High-Level Relationship of Sun Cluster Software Components

See Chapter 4 for questions and answers about cluster members.

Multihost Devices

Disks that can be connected to more than one node at a time are multihost devices. In the Sun Cluster environment, multihost storage makes disks highly available. Sun Cluster requires multihost storage for two-node clusters to establish quorum. Greater than three-node clusters do not need multihost storage.

Multihost devices have the following characteristics.

- They can tolerate single node failures.
- They store application data and can also store application binaries and configuration files.
- They protect against node failures. If client requests are accessing the data through one node and it fails, the requests are switched over to use another node that has a direct connection to the same disks.
- They are either accessed globally through a primary node that “masters” the disks, or by direct concurrent access through local paths. The only application that uses direct concurrent access currently is Oracle Real Application Clusters.

A volume manager provides for mirrored or RAID-5 configurations for data redundancy of the multihost devices. Currently, Sun Cluster supports Solaris Volume Manager™ and VERITAS Volume Manager, which is available for use in only SPARC based clusters, as volume managers, and the RDAC RAID-5 hardware controller on several hardware RAID platforms.

Combining multihost devices with disk mirroring and striping protects against both node failure and individual disk failure.

See Chapter 4 for questions and answers about multihost storage.

Multi-Initiator SCSI

This section applies only to SCSI storage devices and not to Fibre Channel storage used for the multihost devices.

In a standalone server, the server node controls the SCSI bus activities by way of the SCSI host adapter circuit connecting this server to a particular SCSI bus. This SCSI host adapter circuit is referred to as the *SCSI initiator*. This circuit initiates all bus activities for this SCSI bus. The default SCSI address of SCSI host adapters in Sun systems is 7.

Cluster configurations share storage between multiple server nodes, using multihost devices. When the cluster storage consists of singled-ended or differential SCSI devices, the configuration is referred to as multi-initiator SCSI. As this terminology implies, more than one SCSI initiator exists on the SCSI bus.

The SCSI specification requires that each device on a SCSI bus has a unique SCSI address. (The host adapter is also a device on the SCSI bus.) The default hardware configuration in a multi-initiator environment results in a conflict because all SCSI host adapters default to 7.

To resolve this conflict, on each SCSI bus, leave one of the SCSI host adapters with the SCSI address of 7, and set the other host adapters to unused SCSI addresses. Proper planning dictates that these “unused” SCSI addresses include both currently and eventually unused addresses. An example of addresses unused in the future is the addition of storage by installing new drives into empty drive slots.

In most configurations, the available SCSI address for a second host adapter is 6.

You can change the selected SCSI addresses for these host adapters by using one of the following tools to set the `scsi-initiator-id` property:

- `eeprom`(1M)
- The OpenBoot PROM on a SPARC based system
- The SCSI utility that you optionally run after the BIOS boots on an x86 based system

You can set this property globally for a node or on a per-host-adapter basis. Instructions for setting a unique `scsi-initiator-id` for each SCSI host adapter are included in the chapter for each disk enclosure in the *Sun Cluster Hardware Collection*.

Local Disks

Local disks are the disks that are only connected to a single node. They are, therefore, not protected against node failure (not highly available). However, all disks, including local disks, are included in the global namespace and are configured as *global devices*. Therefore, the disks themselves are visible from all cluster nodes.

You can make the file systems on local disks available to other nodes by putting them under a global mount point. If the node that currently has one of these global file systems mounted fails, all nodes lose access to that file system. Using a volume manager lets you mirror these disks so that a failure cannot cause these file systems to become inaccessible, but volume managers do not protect against node failure.

See the section "Global Devices" on page 37 for more information about global devices.

Removable Media

Removable media such as tape drives and CD-ROM drives are supported in a cluster. In general, you install, configure, and service these devices in the same way as in a non-clustered environment. These devices are configured as global devices in Sun Cluster, so each device can be accessed from any node in the cluster. Refer to *Sun Cluster 3.x Hardware Administration Manual for Solaris OS* for information on installing and configuring removable media.

See the section "Global Devices" on page 37 for more information about global devices.

Cluster Interconnect

The *cluster interconnect* is the physical configuration of devices used to transfer cluster-private communications and data service communications between cluster nodes. Because the interconnect is used extensively for cluster-private communications, it can limit performance.

Only cluster nodes can be connected to the cluster interconnect. The Sun Cluster security model assumes that only cluster nodes have physical access to the cluster interconnect.

All nodes must be connected by the cluster interconnect through at least two redundant physically independent networks, or paths, to avoid a single point of failure. You can have several physically independent networks (two to six) between any two nodes. The cluster interconnect consists of three hardware components: adapters, junctions, and cables.

The following list describes each of these hardware components.

- Adapters – The network interface cards that reside in each cluster node. Their names are constructed from a device name immediately followed by a physical-unit number, for example, qfe2. Some adapters have only one physical network connection, but others, like the qfe card, have multiple physical connections. Some also contain both network interfaces and storage interfaces.

  A network adapter with multiple interfaces could become a single point of failure if the entire adapter fails. For maximum availability, plan your cluster so that the only path between two nodes does not depend on a single network adapter.
- Junctions – The switches that reside outside of the cluster nodes. They perform pass-through and switching functions to enable you to connect more than two nodes together. In a two-node cluster, you do not need junctions because the nodes can be directly connected to each other through redundant physical cables connected to redundant adapters on each node. Greater than two-node configurations generally require junctions.
- Cables – The physical connections that go either between two network adapters or between an adapter and a junction.

See Chapter 4 for questions and answers about the cluster interconnect.

Public Network Interfaces

Clients connect to the cluster through the public network interfaces. Each network adapter card can connect to one or more public networks, depending on whether the card has multiple hardware interfaces. You can set up nodes to include multiple public network interface cards configured so that multiple cards are active, and serve as failover backups for one another. If one of the adapters fails, IP Network Multipathing software is called to fail over the defective interface to another adapter in the group.

No special hardware considerations relate to clustering for the public network interfaces.

See Chapter 4 for questions and answers about public networks.

Client Systems

Client systems include workstations or other servers that access the cluster over the public network. Client-side programs use data or other services provided by server-side applications running on the cluster.

Client systems are not highly available. Data and applications on the cluster are highly available.

See Chapter 4 for questions and answers about client systems.

Console Access Devices

You must have console access to all cluster nodes. To gain console access, use the terminal concentrator purchased with your cluster hardware, the System Service Processor (SSP) on Sun Enterprise E10000™ servers (for SPARC based clusters), the system controller on Sun Fire™ servers (also for SPARC based clusters), or another device that can access `ttya` on each node.

Only one supported terminal concentrator is available from Sun and use of the supported Sun terminal concentrator is optional. The terminal concentrator enables access to `/dev/console` on each node by using a TCP/IP network. The result is console-level access for each node from a remote workstation anywhere on the network.

The System Service Processor (SSP) provides console access for Sun Enterprise E10000 servers. The SSP is a machine on an Ethernet network that is configured to support the Sun Enterprise E10000 server. The SSP is the administrative console for the Sun Enterprise E10000 server. Using the Sun Enterprise E10000 Network Console feature, any workstation in the network can open a host console session.

Other console access methods include other terminal concentrators, `tip(1)` serial port access from another node and dumb terminals. You can use Sun™ keyboards and monitors, or other serial port devices if your hardware service provider supports them.

Administrative Console

You can use a dedicated UltraSPARC® workstation or a Sun Fire™ V65x server, known as the *administrative console*, to administer the active cluster. Usually, you install and run administrative tool software, such as the Cluster Control Panel (CCP) and the Sun Cluster module for the Sun Management Center™ product (for use with SPARC based clusters only), on the administrative console. Using `cconsole` under the CCP enables you to connect to more than one node console at a time. For more information about using the CCP, see the *Sun Cluster System Administration Guide*.

The administrative console is not a cluster node. You use the administrative console for remote access to the cluster nodes, either over the public network, or optionally through a network-based terminal concentrator. If your cluster consists of the Sun Enterprise E10000 platform, you must have the ability to log in from the administrative console to the System Service Processor (SSP) and connect by using the `netcon(1M)` command.

Typically, you configure nodes without monitors. Then, you access the node's console through a `telnet` session from the administrative console, which is connected to a terminal concentrator, and from the terminal concentrator to the node's serial port. (In the case of a Sun Enterprise E10000 server, you connect from the System Service Processor.) See "Console Access Devices" on page 26 for more information.

Sun Cluster does not require a dedicated administrative console, but using one provides these benefits:

- Enables centralized cluster management by grouping console and management tools on the same machine
- Provides potentially quicker problem resolution by your hardware service provider

See Chapter 4 for questions and answers about the administrative console.

SPARC: Sun Cluster Topology Examples

A topology is the connection scheme that connects the cluster nodes to the storage platforms that are used in the cluster. Sun Cluster supports any topology that adheres to the following guidelines.

- Sun Cluster that is composed of SPARC based systems supports a maximum of eight nodes in a cluster, regardless of the storage configurations that you implement.
- A shared storage device can connect to as many nodes as the storage device supports.
- Shared storage devices do not need to connect to all nodes of the cluster. However, these storage devices must connect to at least two nodes.

Sun Cluster does not require you to configure a cluster by using specific topologies. The following topologies are described to provide the vocabulary to discuss a cluster's connection scheme. These topologies are typical connection schemes.

- Clustered pairs
- Pair+N
- N+1 (star)
- N*N (scalable)

The following sections include sample diagrams of each topology.

SPARC: Clustered Pairs Topology

A clustered pairs topology is two or more pairs of nodes operating under a single cluster administrative framework. In this configuration, failover occurs only between a pair. However, all nodes are connected by the cluster interconnect and operate under Sun Cluster software control. You might use this topology to run a parallel database application on one pair and a failover or scalable application on another pair.

Using the cluster file system, you could also have a two-pair configuration where more than two nodes run a scalable service or parallel database even though all of the nodes are not directly connected to the disks that store the application data.

The following figure illustrates a clustered pair configuration.

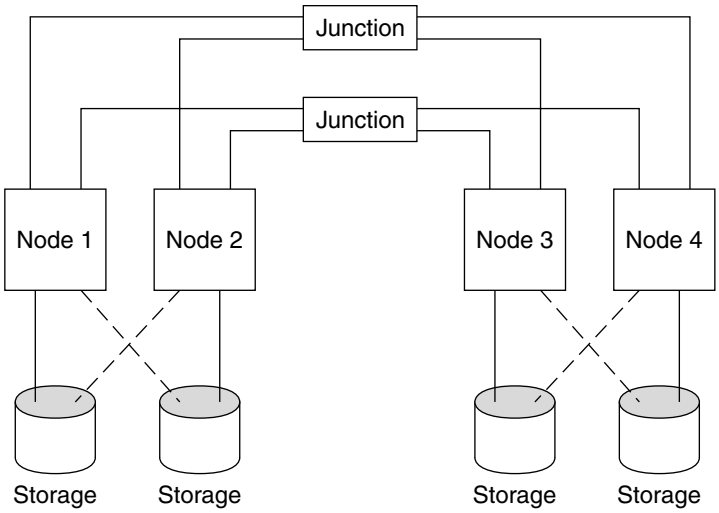


FIGURE 2–2 SPARC: Clustered Pairs Topology

SPARC: Pair+N Topology

The pair+N topology includes a pair of nodes directly connected to shared storage and an additional set of nodes that use the cluster interconnect to access shared storage—they have no direct connection themselves.

The following figure illustrates a pair+N topology where two of the four nodes (Node 3 and Node 4) use the cluster interconnect to access the storage. This configuration can be expanded to include additional nodes that do not have direct access to the shared storage.

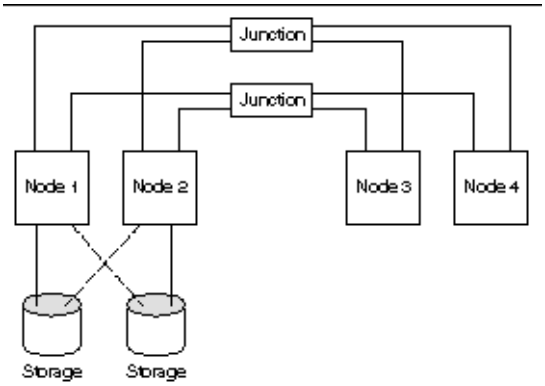


FIGURE 2–3 SPARC: Pair+N Topology

SPARC: N+1 (Star) Topology

An N+1 topology includes some number of primary nodes and one secondary node. You do not have to configure the primary nodes and secondary node identically. The primary nodes actively provide application services. The secondary node need not be idle while waiting for a primary to fail.

The secondary node is the only node in the configuration that is physically connected to all the multihost storage.

If a failure occurs on a primary, Sun Cluster fails over the resources to the secondary, where the resources function until they are switched back (either automatically or manually) to the primary.

The secondary must always have enough excess CPU capacity to handle the load if one of the primaries fails.

The following figure illustrates an N+1 configuration.

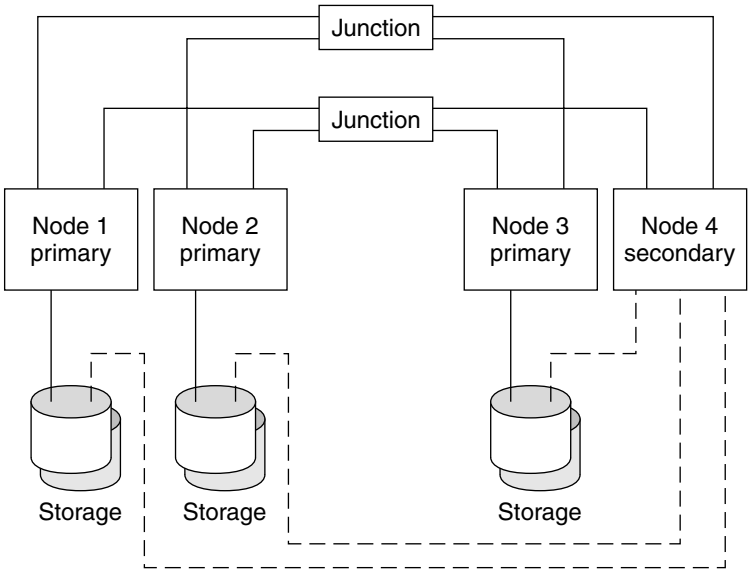


FIGURE 2–4 SPARC: N+1 Topology

SPARC: N*N (Scalable) Topology

An N*N topology allows every shared storage device in the cluster to connect to every node in the cluster. This topology allows highly available applications to failover from one node to another without service degradation. When failover occurs, the new node can access the storage device using a local path instead of the private interconnect.

The following figure illustrates an N*N configuration.

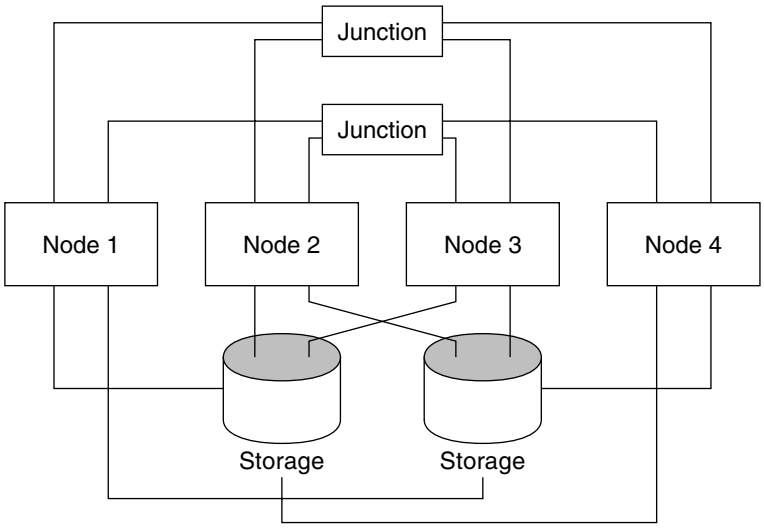


FIGURE 2–5 SPARC: N*N Topology

x86: Sun Cluster Topology Examples

A topology is the connection scheme that connects the cluster nodes to the storage platforms that are used in the cluster. Sun Cluster supports any topology that adheres to the following guidelines.

- Sun Cluster that is composed of x86 based systems supports two nodes in a cluster.
- Shared storage devices must connect to both nodes.

Sun Cluster does not require you to configure a cluster by using specific topologies. The following clustered pair topology, which is the only topology for clusters that are composed of x86 based nodes, is described to provide the vocabulary to discuss a cluster's connection scheme. This topology is a typical connection scheme.

The following section includes a sample diagram of the topology.

x86: Clustered Pair Topology

A clustered pair topology is two nodes operating under a single cluster administrative framework. In this configuration, failover occurs only between a pair. However, all nodes are connected by the cluster interconnect and operate under Sun Cluster software control. You might use this topology to run a parallel database or a failover or scalable application on the pair.

The following figure illustrates a clustered pair configuration.

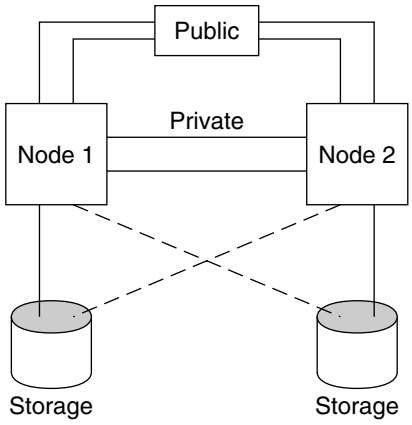


FIGURE 2–6 x86: Clustered Pair Topology

CHAPTER 3

Key Concepts – Administration and Application Development

This chapter describes the key concepts related to the software components of a SunPlex system. The topics covered include:

This information is directed primarily toward system administrators and application developers using the SunPlex API and SDK. Cluster system administrators can use this information in preparation for to installing, configuring, and administering cluster software. Application developers can use the information to understand the cluster environment in which they will be working.

Administrative Interfaces

You can choose how you install, configure, and administer the SunPlex system from several user interfaces. You can accomplish system administration tasks either through the SunPlex Manager graphic user interface (GUI), or through the documented command-line interface. On top of the command-line interface are some utilities, such as `scinstall` and `scsetup`, to simplify selected installation and configuration tasks.

The SunPlex system also has a module that runs as part of Sun Management Center that provides a GUI to particular cluster tasks. This module is available for use in only SPARC based clusters. Refer to "Administration Tools" in *Sun Cluster System Administration Guide for Solaris OS* for complete descriptions of the administrative interfaces.

Cluster Time

Time between all nodes in a cluster must be synchronized. Whether you synchronize the cluster nodes with any outside time source is not important to cluster operation. The SunPlex system employs the Network Time Protocol (NTP) to synchronize the clocks between nodes.

In general, a change in the system clock of a fraction of a second causes no problems. However, if you run `date(1)`, `rdate(1M)`, or `xntpdate(1M)` (interactively, or within `cron` scripts) on an active cluster, you can force a time change much larger than a fraction of a second to synchronize the system clock to the time source. This forced change might cause problems with file modification timestamps or confuse the NTP service.

When you install the Solaris operating environment on each cluster node, you have an opportunity to change the default time and date setting for the node. In general, you can accept the factory default.

When you install Sun Cluster software using `scinstall(1M)`, one step in the process is to configure NTP for the cluster. Sun Cluster software supplies a template file, `ntp.cluster` (see `/etc/inet/ntp.cluster` on an installed cluster node), that establishes a peer relationship between all cluster nodes, with one node being the "preferred" node. Nodes are identified by their private host names and time synchronization occurs across the cluster interconnect. The instructions for how to configure the cluster for NTP, see "Installing and Configuring Sun Cluster Software" in *Sun Cluster Software Installation Guide for Solaris OS*.

Alternately, you can set up one or more NTP servers outside the cluster and change the `ntp.conf` file to reflect that configuration.

In normal operation, you should never need to adjust the time on the cluster. However, if the time was set incorrectly when you installed the Solaris operating environment and you want to change it, the procedure for doing so is included in "Administering the Cluster" in *Sun Cluster System Administration Guide for Solaris OS*.

High-Availability Framework

The SunPlex system makes all components on the "path" between users and data highly available, including network interfaces, the applications themselves, the file system, and the multihost devices. In general, a cluster component is highly available if it survives any single (software or hardware) failure in the system.

The following table shows the kinds of SunPlex component failures (both hardware and software) and the kinds of recovery built into the high-availability framework.

TABLE 3–1 Levels of SunPlex Failure Detection and Recovery

Failed Cluster Component	Software Recovery	Hardware Recovery
Data service	HA API, HA framework	N/A
Public network adapter	IP Network Multipathing	Multiple public network adapter cards
Cluster file system	Primary and secondary replicas	Multihost devices
Mirrored multihost device	Volume management (Solaris Volume Manager and VERITAS Volume Manager, which is available in SPARC based clusters only)	Hardware RAID-5 (for example, Sun StorEdge™ A3x00)
Global device	Primary and secondary replicas	Multiple paths to the device, cluster transport junctions
Private network	HA transport software	Multiple private hardware-independent networks
Node	CMM, failfast driver	Multiple nodes

Sun Cluster software's high-availability framework detects a node failure quickly and creates a new equivalent server for the framework resources on a remaining node in the cluster. At no time are all framework resources unavailable. Framework resources unaffected by a crashed node are fully available during recovery. Furthermore, framework resources of the failed node become available as soon as they are recovered. A recovered framework resource does not have to wait for all other framework resources to complete their recovery.

Most highly available framework resources are recovered transparently to the applications (data services) using the resource. The semantics of framework resource access are fully preserved across node failure. The applications simply cannot tell that the framework resource server has been moved to another node. Failure of a single node is completely transparent to programs on remaining nodes using the files, devices, and disk volumes attached to this node, as long as an alternative hardware path exists to the disks from another node. An example is the use of multihost devices that have ports to multiple nodes.

Cluster Membership Monitor

To ensure that data is kept safe from corruption, all nodes must reach a consistent agreement on the cluster membership. When necessary, the CMM coordinates a cluster reconfiguration of cluster services (applications) in response to a failure.

The CMM receives information about connectivity to other nodes from the cluster transport layer. The CMM uses the cluster interconnect to exchange state information during a reconfiguration.

After detecting a change in cluster membership, the CMM performs a synchronized configuration of the cluster, where cluster resources might be redistributed based on the new membership of the cluster.

Unlike previous Sun Cluster software releases, CMM runs entirely in the kernel.

See "About Failure Fencing" on page 51 for more information on how the cluster protects itself from partitioning into multiple separate clusters.

Failfast Mechanism

If the CMM detects a critical problem with a node, it calls upon the cluster framework to forcibly shut down (panic) the node and to remove it from the cluster membership. The mechanism by which this occurs is called *failfast*. Failfast will cause a node to shut down in two ways.

- If a node leaves the cluster and then attempts to start a new cluster without having quorum, it is "fenced" from accessing the shared disks. See "About Failure Fencing" on page 51 for details on this use of failfast.
- If one or more cluster-specific daemons die (`clexecd`, `rpc.pmfd`, `rgmd`, or `rpc.ed`) the failure is detected by the CMM and the node panics. When the death of a cluster daemon causes a node to panic, a message similar to the following will display on the console for that node.

```
panic[cpu0]/thread=40e60: Failfast: Aborting because "pmfd" died 35 seconds ago.
409b8 cl_runtime:__0FZsc_syslog_msg_log_no_argsPviTCPCcTB+48 (70f900, 30, 70df54, 407acc, 0)
%l0-7: 1006c80 000000a 000000a 10093bc 406d3c80 7110340 0000000 4001 fbf0
```

After the panic, the node might reboot and attempt to rejoin the cluster or, if the cluster is composed of SPARC based systems, stay at the OpenBoot™ PROM (OBP) prompt. The action that is taken is determined by the setting of the `auto-boot?` parameter. You can set `auto-boot?` with `eeprom`(1M), at the OpenBoot PROM `ok` prompt.

Cluster Configuration Repository (CCR)

The CCR uses a two-phase commit algorithm for updates: An update must complete successfully on all cluster members or the update is rolled back. The CCR uses the cluster interconnect to apply the distributed updates.

Caution – Although the CCR consists of text files, never edit the CCR files manually. Each file contains a checksum record to ensure consistency between nodes. Manually updating CCR files can cause a node or the entire cluster to stop functioning.

The CCR relies on the CMM to guarantee that a cluster is running only when quorum is established. The CCR is responsible for verifying data consistency across the cluster, performing recovery as necessary, and facilitating updates to the data.

Global Devices

The SunPlex system uses *global devices* to provide cluster-wide, highly available access to any device in a cluster, from any node, without regard to where the device is physically attached. In general, if a node fails while providing access to a global device, the Sun Cluster software automatically discovers another path to the device and redirects the access to that path. SunPlex global devices include disks, CD-ROMs, and tapes. However, disks are the only supported multiported global devices. This means that CD-ROM and tape devices are not currently highly available devices. The local disks on each server are also not multiported, and thus are not highly available devices.

The cluster automatically assigns unique IDs to each disk, CD-ROM, and tape device in the cluster. This assignment allows consistent access to each device from any node in the cluster. The global device namespace is held in the `/dev/global` directory. See "Global Namespace" on page 42 for more information.

Multiported global devices provide more than one path to a device. In the case of multihost disks, because the disks are part of a disk device group hosted by more than one node, the multihost disks are made highly available.

Device ID (DID)

The Sun Cluster software manages global devices through a construct known as the device ID (DID) pseudo driver. This driver is used to automatically assign unique IDs to every device in the cluster, including multihost disks, tape drives, and CD-ROMs.

The device ID (DID) pseudo driver is an integral part of the global device access feature of the cluster. The DID driver probes all nodes of the cluster and builds a list of unique disk devices, assigning each a unique major and minor number that is consistent on all nodes of the cluster. Access to the global devices is performed utilizing the unique device ID assigned by the DID driver instead of the traditional Solaris device IDs, such as `c0t0d0` for a disk.

This approach ensures that any application accessing disks (such as a volume manager or applications using raw devices) uses a consistent path across the cluster. This consistency is especially important for multihost disks, because the local major and minor numbers for each device can vary from node to node, thus changing the Solaris device naming conventions as well. For example, `Node 1` might see a multihost disk as `c1t2d0`, and node2 might see the same disk completely differently, as `c3t2d0`. The DID driver assigns a global name, such as d10, that the nodes would use instead, giving each node a consistent mapping to the multihost disk.

You update and administer Device IDs through `scdidadm(1M)` and `scgdevs(1M)`. See the following man pages for more information:

- `scdidadm`(1M)
- `scgdevs`(1M)

Disk Device Groups

In the SunPlex system, all multihost devices must be under control of the Sun Cluster software. You first create volume manager disk groups—either Solaris Volume Manager disk sets or VERITAS Volume Manager disk groups (available for use in only SPARC based clusters)—on the multihost disks. Then, you register the volume manager disk groups as *disk device groups*. A disk device group is a type of global device. In addition, the Sun Cluster software automatically creates a raw disk device group for each disk and tape device in the cluster. However, these cluster device groups remain in an offline state until you access them as global devices.

Registration provides the SunPlex system information about which nodes have a path to what volume manager disk groups. At this point, the volume manager disk groups become globally accessible within the cluster. If more than one node can write to (master) a disk device group, the data stored in that disk device group becomes highly available. The highly available disk device group can be used to house cluster file systems.

Note – Disk device groups are independent of resource groups. One node can master a resource group (representing a group of data service processes) while another can master the disk group(s) being accessed by the data services. However, the best practice is to keep the disk device group that stores a particular application's data and the resource group that contains the application's resources (the application daemon) on the same node. Refer to "Relationship Between Resource Groups and Disk Device Groups" in *Sun Cluster Data Services Planning and Administration Guide for Solaris OS* for more information about the association between disk device groups and resource groups.

With a disk device group, the volume manager disk group becomes "global" because it provides multipath support to the underlying disks. Each cluster node physically attached to the multihost disks provides a path to the disk device group.

Disk Device Group Failover

Because a disk enclosure is connected to more than one node, all disk device groups in that enclosure are accessible through an alternate path if the node currently mastering the device group fails. The failure of the node mastering the device group does not affect access to the device group except for the time it takes to perform the recovery and consistency checks. During this time, all requests are blocked (transparently to the application) until the system makes the device group available.

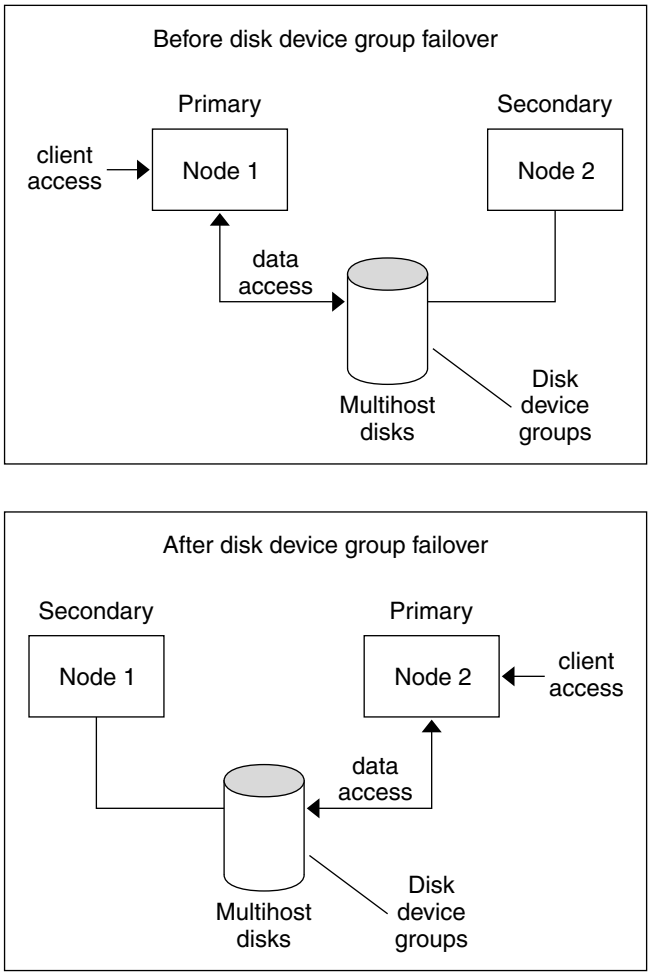


FIGURE 3–1 Disk Device Group Failover

Multiported Disk Device Groups

This section describes disk device group properties that enable you to balance performance and availability in a multiported disk configuration. Sun Cluster software provides two properties used to configure a multiported disk configuration: `preferenced` and `numsecondaries`. You can control the order in which nodes attempt to assume control if a failover occurs by using the `preferenced` property. Use the `numsecondaries` property to set a desired number of secondary nodes for a device group.

A highly available service is considered down when the primary goes down and when no eligible secondary nodes can be promoted to primary. If service failover occurs and the `preferenced` property is `true`, then the nodes follow the order in the nodelist to select a secondary. the nodelist that is set by the defines the order in which nodes will attempt to assume primary control or transition from spare to secondary. You can dynamically change the preference of a device service by using the `scsetup`(1M) utility. The preference that is associated with dependent service providers, for example a global file system, will be that of the device service.

Secondary nodes are check-pointed by the primary node during normal operation. In a multiported disk configuration, checkpointing each secondary node causes cluster performance degradation and memory overhead. Spare node support was implemented to minimize the performance degradation and memory overhead caused by checkpointing. By default, your disk device group will have one primary and one secondary. The remaining available provider nodes will come online in the spare state. If failover occurs, the secondary will become primary and the node highest in priority on the nodelist will become secondary.

The desired number of secondary nodes can be set to any integer between one and the number of operational non-primary provider nodes in the device group.

Note – If you are using Solaris Volume Manager, you must create the disk device group before you can set the `numsecondaries` property to a number other than the default.

The default desired number of secondaries for device services is one. The actual number of secondary providers that is maintained by the replica framework is the desired number, unless the number of operational non-primary providers is less than the desired number. You will want to alter the `numsecondaries` property and double check the nodelist if you are adding or removing nodes from your configuration. Maintaining the nodelist and desired number of secondaries will prevent conflict between the configured number of secondaries and the actual number allowed by the framework.Use the `metaset`(1M) command for Solaris Volume Manager device groups or, if you're using Veritas Volume Manager, the `scconf`(1M) command for VxVM disk device groups in conjunction with the `preferenced` and `numsecondaries` property settings to manage addition and removal of nodes from your configuration. Refer to "Administering Cluster File Systems Overview" in *Sun Cluster System Administration Guide for Solaris OS* for procedural information about changing disk device group properties.

Global Namespace

The Sun Cluster software mechanism that enables global devices is the *global namespace*. The global namespace includes the `/dev/global/` hierarchy as well as the volume manager namespaces. The global namespace reflects both multihost disks and local disks (and any other cluster device, such as CD-ROMs and tapes), and provides multiple failover paths to the multihost disks. Each node physically connected to multihost disks provides a path to the storage for any node in the cluster.

Normally, for Solaris Volume Manager, the volume manager namespaces are located in the `/dev/md/`*diskset*`/dsk` (and `rdsk`) directories. For Veritas VxVM, the volume manager namespaces are located in the `/dev/vx/dsk/`*disk-group* and `/dev/vx/rdsk/`*disk-group* directories. These namespaces consist of directories for each Solaris Volume Manager diskset and each VxVM disk group imported throughout the cluster, respectively. Each of these directories houses a device node for each metadevice or volume in that diskset or disk group.

In the SunPlex system, each of the device nodes in the local volume manager namespace is replaced by a symbolic link to a device node in the `/global/.devices/node@`*nodeID* file system, where *nodeID* is an integer that represents the nodes in the cluster. Sun Cluster software continues to present the volume manager devices, as symbolic links, in their standard locations as well. Both the global namespace and standard volume manager namespace are available from any cluster node.

The advantages of the global namespace include:

- Each node remains fairly independent, with little change in the device administration model.
- Devices can be selectively made global.
- Third-party link generators continue to work.
- Given a local device name, an easy mapping is provided to obtain its global name.

Local and Global Namespaces Example

The following table shows the mappings between the local and global namespaces for a multihost disk, `c0t0d0s0`.

TABLE 3–2 Local and Global Namespaces Mappings

Component/Path	Local Node Namespace	Global Namespace
Solaris logical name	`/dev/dsk/c0t0d0s0`	`/global/.devices/node@`*nodeID*`/dev/dsk/c0t0d0s0`
DID name	`/dev/did/dsk/d0s0`	`/global/.devices/node@`*nodeID*`/dev/did/dsk/d0s0`
Solaris Volume Manager	`/dev/md/`*diskset*`/dsk/d0`	`/global/.devices/node@`*nodeID*`/dev/md/`*diskset*`/dsk/d0`
SPARC: VERITAS Volume Manager	`/dev/vx/dsk/`*disk-group*`/v0`	`/global/.devices/node@`*nodeID*`/dev/vx/dsk/`*disk-group*`/v0`

The global namespace is automatically generated on installation and updated with every reconfiguration reboot. You can also generate the global namespace by running the `scgdevs(1M)` command.

Cluster File Systems

The cluster file system has the following features:

- File access locations are transparent. A process can open a file located anywhere in the system and processes on all nodes can use the same path name to locate a file.

  Note – When the cluster file system reads files, it does not update the access time on those files.

- Coherency protocols are used to preserve the UNIX file access semantics even if the file is accessed concurrently from multiple nodes.
- Extensive caching is used along with zero-copy bulk I/O movement to move file data efficiently.
- The cluster file system provides highly available advisory file locking functionality using the `fcntl(2)` interfaces. Applications running on multiple cluster nodes can synchronize access to data using advisory file locking on a cluster file system file. File locks are recovered immediately from nodes that leave the cluster, and from applications that fail while holding locks.
- Continuous access to data is ensured, even when failures occur. Applications are not affected by failures as long as a path to disks is still operational. This guarantee is maintained for raw disk access and all file system operations.

- Cluster file systems are independent from the underlying file system and volume management software. Cluster file systems make any supported on-disk file system global.

You can mount a file system on a global device globally with `mount -g` or locally with `mount`.

Programs can access a file in a cluster file system from any node in the cluster through the same file name (for example, `/global/foo`).

A cluster file system is mounted on all cluster members. You cannot mount a cluster file system on a subset of cluster members.

A cluster file system is not a distinct file system type. That is, clients see the underlying file system (for example, UFS).

Using Cluster File Systems

In the SunPlex system, all multihost disks are placed into disk device groups, which can be Solaris Volume Manager disksets, VxVM disk groups, or individual disks that are not under control of a software-based volume manager.

For a cluster file system to be highly available, the underlying disk storage must be connected to more than one node. Therefore, a local file system (a file system that is stored on a node's local disk) that is made into a cluster file system is not highly available.

As with normal file systems, you can mount cluster file systems in two ways:

- **Manually**—Use the `mount` command and the `-g` or `-o global` mount options to mount the cluster file system from the command line, for example:

  ```
  SPARC: # mount -g /dev/global/dsk/d0s0 /global/oracle/data
  ```

- **Automatically**—Create an entry in the `/etc/vfstab` file with a `global` mount option to mount the cluster file system at boot. You then create a mount point under the `/global` directory on all nodes. The directory `/global` is a recommended location, not a requirement. Here's a sample line for a cluster file system from an `/etc/vfstab` file:

```
SPARC: /dev/md/oracle/dsk/d1 /dev/md/oracle/rdsk/d1 /global/oracle/data ufs 2 yes global,logging
```

Note – While Sun Cluster software does not impose a naming policy for cluster file systems, you can ease administration by creating a mount point for all cluster file systems under the same directory, such as `/global/`*disk-device-group*. See *Sun Cluster 3.1 9/04 Software Collection for Solaris OS (SPARC Platform Edition)* and *Sun Cluster System Administration Guide for Solaris OS* for more information.

HAStoragePlus Resource Type

The HAStoragePlus resource type is designed to make non-global file system configurations such as UFS and VxFS highly available. Use HAStoragePlus to integrate your local file system into the Sun Cluster environment and make the file system highly available. HAStoragePlus provides additional file system capabilities such as checks, mounts, and forced unmounts that enable Sun Cluster to fail over local file systems. In order to fail over, the local file system must reside on global disk groups with affinity switchovers enabled.

See "Enabling Highly Available Local File Systems" in *Sun Cluster Data Services Planning and Administration Guide for Solaris OS* for information on how to use the HAStoragePlus resource type.

HAStoragePlus can also used to synchronize the startup of resources and disk device groups upon which the resources depend. For more information, see "Resources, Resource Groups, and Resource Types" on page 70.

The Syncdir Mount Option

The syncdir mount option can be used for cluster file systems that use UFS as the underlying file system. However, there is a significant performance improvement if you do not specify syncdir. If you specify syncdir, the writes are guaranteed to be POSIX compliant. If you do not, you will have the same behavior that is seen with NFS file systems. For example, under some cases, without syncdir, you would not discover an out of space condition until you close a file. With syncdir (and POSIX behavior), the out-of-space condition would have been discovered during the write operation. The cases in which you could have problems if you do not specify syncdir are rare, so it is recommended that you do not specify syncdir and receive the performance benefit.

If you are using a SPARC based cluster, Veritas VxFS does not have a mount-option equivalent to the syncdir mount option for UFS. VxFS behavior is the same as for UFS when the syncdir mount option is not specified.

See "File Systems FAQs" on page 88 for frequently asked questions about global devices and cluster file systems.

Disk-Path Monitoring

The current release of Sun Cluster software supports disk-path monitoring (DPM). This section provides conceptual information about DPM, the DPM daemon, and administration tools that you use to monitor disk paths. Refer to *Sun Cluster System Administration Guide for Solaris OS* for procedural information about how to monitor, unmonitor, and check the status of disk paths.

Note – DPM is not supported on nodes that run versions that were released prior to Sun Cluster 3.1 4/04 software. Do not use DPM commands while a rolling upgrade is in progress. After all nodes are upgraded, the nodes must be online to use DPM commands.

Overview

DPM improves the overall reliability of failover and switchover by monitoring the secondary disk-path availability. Use the `scdpm` command to verify availability of the disk path that is used by a resource before the resource is switched. Options that are provided with the `scdpm` command enable you to monitor disk paths to a single node or to all nodes in the cluster. See the `scdpm`(1M) man page for more information about command-line options.

The DPM components are installed from the `SUNWscu` package. The `SUNWscu` package is installed by the standard Sun Cluster installation procedure. See the `scinstall`(1M) man page for installation interface details. The following table describes the default install location of DPM components.

Location	Component
Daemon	`/usr/cluster/lib/sc/scdpmd`
Command-line interface	`/usr/cluster/bin/scdpm`
Share libraries	`/user/cluster/lib/libscdpm.so`
Daemon status file (created at runtime)	`/var/run/cluster/scdpm.status`

A multithreaded DPM daemon runs on each node. The DPM daemon (`scdpmd`) is started by an `rc.d` script when a node boots. If a problem arises, the daemon is managed by `pmfd` and restarts automatically. The following list describes how the `scdpmd` works on initial startup.

Note – At startup, the status for each disk path is initialized to `UNKNOWN`.

1. The DPM daemon gathers disk path and node name information from the previous status file or from the CCR database. Refer to "Cluster Configuration Repository (CCR)" on page 37 for more information about the CCR. After a DPM daemon is started, you can force the daemon to read the list of monitored disks from a specified file name.
2. The DPM daemon initializes the communication interface to answer requests from components that are external to the daemon, such as the command-line interface.
3. The DPM daemon pings each disk path in the monitored list every 10 minutes by using `scsi_inquiry` commands. Each entry is locked to prevent the communication interface access to the content of an entry that is being modified.
4. The DPM daemon notifies the Sun Cluster Event Framework and logs the new status of the path through the UNIX `syslogd`(1M) mechanism.

Note – All errors that are related to the daemon are reported by `pmfd` (1M). All the functions from the API return `0` on success and `-1` for any failure.

The DPM Daemon monitors the availability of the logical path that is visible through multipath drivers such as MPxIO, HDLM, and PowerPath. The individual physical paths that are managed by these drivers are not monitored, because the multipath driver masks individual failures from the DPM daemon.

Monitoring Disk Paths

This section describes two methods for monitoring disk paths in your cluster. The first method is provided by the `scdpm` command. Use this command to monitor, unmonitor, or display the status of disk paths in your cluster. This command is also useful for printing the list of faulted disks and monitoring disk paths from a file.

The second method for monitoring disk paths in your cluster is provided by the SunPlex Manager graphical user interface (GUI). SunPlex Manager provides a topological view of the monitored disk paths in your cluster. The view is updated every 10 minutes to provide information about the number of failed pings. Use the information that is provided by the SunPlex Manager GUI in conjunction with the `scdpm`(1M) command to administer disk paths. Refer to "Administering Sun Cluster With the Graphical User Interfaces" in *Sun Cluster System Administration Guide for Solaris OS* for information about SunPlex Manager.

Using the `scdpm` Command to Monitor Disk Paths

The `scdpm`(1M) command provides DPM administration commands that enable you to perform the following tasks:

- Monitoring a new disk path
- Unmonitoring a disk path
- Rereading the configuration data from the CCR database
- Reading the disks to monitor or unmonitor from a specified file
- Reporting the status of a disk path or all disk paths in the cluster
- Printing all the disk paths that are accessible from a node

Issue the `scdpm`(1M) command with the disk-path argument from any active node to perform DPM administration tasks on the cluster. The disk-path argument is always constituted of a node name and a disk name. The node name is not required and defaults to `all` if none is specified. The following table describes naming conventions for the disk path.

Note – Use of the global disk-path name is strongly recommended, because the global disk-path name is consistent throughout the cluster. The UNIX disk-path name is not consistent throughout the cluster. The UNIX disk path for one disk can differ from cluster node to cluster node. The disk path could be `c1t0d0` on one node and `c2t0d0` on another node. If you use UNIX disk-path names, use the `scdidadm -L` command to map the UNIX disk-path name to the global disk-path name before issuing DPM commands. See the `scdidadm`(1M) man page.

TABLE 3–3 Sample Disk-Path Names

Name Type	Sample Disk Path Name	Description
Global disk path	`schost-1:/dev/did/dsk/d1`	Disk path `d1` on the `schost-1` node
	`all:d1`	Disk path `d1` on all nodes in the cluster
UNIX disk path	`schost-1:/dev/rdsk/c0t0d0s0`	Disk path `c0t0d0s0` on the `schost-1` node
	`schost-1:all`	All disk paths on the `schost-1` node
All disk paths	`all:all`	All disk paths on all nodes of the cluster

Using SunPlex Manager to Monitor Disk Paths

SunPlex Manager enables you to perform the following basic DPM administration tasks:

- Monitoring a disk path
- Unmonitoring a disk path

- Viewing the status of all disk paths in the cluster.

Refer to the SunPlex Manager online help for procedural information about how to perform disk-path administration by using SunPlex Manager.

Quorum and Quorum Devices

This section contains the following topics:

Note – For a list of the specific devices that Sun Cluster software supports as quorum devices, contact your Sun service provider.

Because cluster nodes share data and resources, a cluster must never split into separate partitions that are active at the same time because multiple active partitions might cause data corruption. The Cluster Membership Monitor (CMM) and quorum algorithm guarantee that at most one instance of the same cluster is operational at any time, even if the cluster interconnect is partitioned.

For more information about CMM, see "Cluster Membership" in *Sun Cluster Overview for Solaris OS*

Two types of problems arise from cluster partitions:

- Split brain
- Amnesia

Split brain occurs when the cluster interconnect between nodes is lost and the cluster becomes partitioned into subclusters. Each partition believes that it is the only partition because the nodes in one partition cannot communicate with the node in other partition.

Amnesia occurs when the cluster restarts after a shutdown with cluster configuration data older than at the time of the shutdown. This problem can occur when you start up the cluster on a node that was not in the last functioning cluster partition.

Sun Cluster software avoids split brain and amnesia by:

- Assigning each node one vote
- Mandating a majority of votes for an operational cluster

A partition with the majority of votes gains *quorum* and is allowed to operate. This majority vote mechanism prevents split brain and amnesia when more than two nodes are configured in a cluster. However, counting node votes alone is not sufficient when more than two nodes are configured in a cluster. In a two-node cluster, a majority is two. If such a two-node cluster becomes partitioned, an external vote is needed for either partition to gain quorum. This external vote is provided by a *quorum device*.

About Quorum Vote Counts

Use the `scstat -q` command to determine the following information:

- Total configured votes
- Current present votes
- Votes required for quorum

For more information on this command, see `scstat`(1M).

Both nodes and quorum devices contribute votes to the cluster to form quorum.

A node contributes votes depending on the node's state:

- A node has a vote count of *one* when it boots and becomes a cluster member.
- A node has a vote count of *zero* when the node is being installed.
- A node has a vote count of *zero* when an system administrator places the node into maintenance state.

Quorum devices contribute votes based on the number of votes connected to the device. When you configure a quorum device, Sun Cluster software assigns the quorum device a vote count of *N*-1 where *N* is the number of connected votes to the quorum device. For example, a quorum device connected to two nodes with nonzero vote counts has a quorum count of one (two minus one).

A quorum device contributes votes if *one* of the following two conditions are true:

- At least one of the nodes to which the quorum device is currently attached is a cluster member.
- At least one of the nodes to which the quorum device is currently attached is booting, and that node was a member of the last cluster partition to own the quorum device.

You configure quorum devices during the cluster installation, or later by using the procedures that are described in "Administering Quorum" in *Sun Cluster System Administration Guide for Solaris OS*.

About Failure Fencing

A major issue for clusters is a failure that causes the cluster to become partitioned (called *split brain*). When this happens, not all nodes can communicate, so individual nodes or subsets of nodes might try to form individual or subset clusters. Each subset or partition might believe it has sole access and ownership to the multihost devices. Multiple nodes attempting to write to the disks can result in data corruption.

Failure fencing limits node access to multihost devices by physically preventing access to the disks. When a node leaves the cluster (it either fails or becomes partitioned), failure fencing ensures that the node can no longer access the disks. Only current member nodes have access to the disks, resulting in data integrity.

Disk device services provide failover capability for services that make use of multihost devices. When a cluster member currently serving as the primary (owner) of the disk device group fails or becomes unreachable, a new primary is chosen, enabling access to the disk device group to continue with only minor interruption. During this process, the old primary must give up access to the devices before the new primary can be started. However, when a member drops out of the cluster and becomes unreachable, the cluster cannot inform that node to release the devices for which it was the primary. Thus, you need a means to enable surviving members to take control of and access global devices from failed members.

The SunPlex system uses SCSI disk reservations to implement failure fencing. Using SCSI reservations, failed nodes are “fenced” away from the multihost devices, preventing them from accessing those disks.

SCSI-2 disk reservations support a form of reservations, which either grants access to all nodes attached to the disk (when no reservation is in place) or restricts access to a single node (the node that holds the reservation).

When a cluster member detects that another node is no longer communicating over the cluster interconnect, it initiates a failure fencing procedure to prevent the other node from accessing shared disks. When this failure fencing occurs, it is normal to have the fenced node panic with a “reservation conflict” messages on its console.

The reservation conflict occurs because after a node has been detected to no longer be a cluster member, a SCSI reservation is put on all of the disks that are shared between this node and other nodes. The fenced node might not be aware that it is being fenced and if it tries to access one of the shared disks, it detects the reservation and panics.

Failfast Mechanism for Failure Fencing

The mechanism by which the cluster framework ensures that a failed node cannot reboot and begin writing to shared storage is called *failfast*.

Nodes that are cluster members continuously enable a specific ioctl, MHIOCENFAILFAST, for the disks to which they have access, including quorum disks. This ioctl is a directive to the disk driver, and gives a node the capability to panic itself if it cannot access the disk due to the disk being reserved by some other node.

The MHIOCENFAILFAST ioctl causes the driver to check the error return from every read and write that a node issues to the disk for the `Reservation_Conflict` error code. The ioctl periodically, in the background, issues a test operation to the disk to check for `Reservation_Conflict`. Both the foreground and background control flow paths panic if `Reservation_Conflict` is returned.

For SCSI-2 disks, reservations are not persistent—they do not survive node reboots. For SCSI-3 disks with Persistent Group Reservation (PGR), reservation information is stored on the disk and persists across node reboots. The failfast mechanism works the same regardless of whether you have SCSI-2 disks or SCSI-3 disks.

If a node loses connectivity to other nodes in the cluster, and it is not part of a partition that can achieve quorum, it is forcibly removed from the cluster by another node. Another node that is part of the partition that can achieve quorum places reservations on the shared disks and when the node that does not have quorum attempts to access the shared disks, it receives a reservation conflict and panics as a result of the failfast mechanism.

After the panic, the node might reboot and attempt to rejoin the cluster or, if the cluster is composed of SPARC based systems, stay at the OpenBoot™ PROM (OBP) prompt. The action that is taken is determined by the setting of the `auto-boot?` parameter. You can set `auto-boot?` with `eeprom`(1M), at the OpenBoot PROM `ok` prompt in a SPARC based cluster, or with the SCSI utility that you optionally run after the BIOS boots in an x86 based cluster.

About Quorum Configurations

The following list contains facts about quorum configurations:

- Quorum devices can contain user data.
- In an N+1 configuration where *N* quorum devices are each connected to one of the `1` through *N* nodes and the `N+1` node, the cluster survives the death of either all `1` through *N* nodes or any of the *N*/**2** nodes. This availability assumes that the quorum device is functioning correctly.
- In an *N*-node configuration where a single quorum device connects to all nodes, the cluster can survive the death of any of the *N*-**1** nodes. This availability assumes that the quorum device is functioning correctly.
- In an *N*-node configuration where a single quorum device connects to all nodes, the cluster can survive the failure of the quorum device if all cluster nodes are available.

For examples of quorum configurations to avoid, see "Bad Quorum Configurations" on page 59. For examples of recommended quorum configurations, see "Recommended Quorum Configurations" on page 55.

Adhering to Quorum Device Requirements

You must adhere to the following requirements. If you do not, you might compromise your cluster's availability.

- Ensure that Sun Cluster software supports your specific device as a quorum device.

  Note – For a list of the specific devices that Sun Cluster software supports as quorum devices, contact your Sun service provider.

  Sun Cluster software supports two types of quorum devices:

  - Multi-hosted shared disks that support SCSI-3 PGR reservations
  - Dual-hosted shared disks that support SCSI-2 reservations

- In a two–node configuration, you must configure at least one quorum device to ensure that a single node can continue if the other node fails. See Figure 3–2.

For examples of quorum configurations to avoid, see "Bad Quorum Configurations" on page 59. For examples of recommended quorum configurations, see "Recommended Quorum Configurations" on page 55.

Adhering to Quorum Device Best Practices

Use the following information to evaluate the best quorum configuration for your topology:

- Do you have a device that is capable of being connected to all nodes of the cluster?
  - If yes, configure that device as your one quorum device. You do *not* need to configure another quorum device because your configuration is the most optimal configuration.

    Caution – If you ignore this requirement and add another quorum device, the additional quorum device reduces your cluster's availability.

  - If no, configure your dual-ported device or devices.
- Ensure that the total number of votes contributed by quorum devices is strictly less than the total number of votes contributed by nodes. Otherwise, your nodes cannot form a cluster if all disks are unavailable—even if all nodes are functioning.

Note – Sometimes, in particular environments, it may be desireable to reduce overall cluster availability in order to meet your needs. In these situations, you can ignore this best practice. However, not adhering to this best practice decreases overall availability. For example, in the configuration that is outlined in "Atypical Quorum Configurations" on page 58 the cluster is less available: the quorum votes exceed the node votes. The cluster has the property that if access to the shared storage between Nodes A and Node B is lost, the entire cluster will fail.

See "Atypical Quorum Configurations" on page 58 for the exception to this best practice.

- Specify a quorum device between every pair of nodes that shares access to a storage device. This quorum configuration speeds up the failure fencing process. See "Quorum in Greater Than Two–Node Configurations" on page 56.
- In general, if the addition of a quorum device makes the total cluster vote even, the total cluster availability decreases.
- Quorum devices slightly slow reconfigurations after a node joins or a node dies. Therefore, do not add more quorum devices than are necessary.

For examples of quorum configurations to avoid, see "Bad Quorum Configurations" on page 59. For examples of recommended quorum configurations, see "Recommended Quorum Configurations" on page 55.

Recommended Quorum Configurations

For examples of quorum configurations to avoid, see "Bad Quorum Configurations" on page 59.

Quorum in Two–Node Configurations

Two quorum votes are required for a two-node cluster to form. These two votes can come from the two cluster nodes, or from just one node and a quorum device.

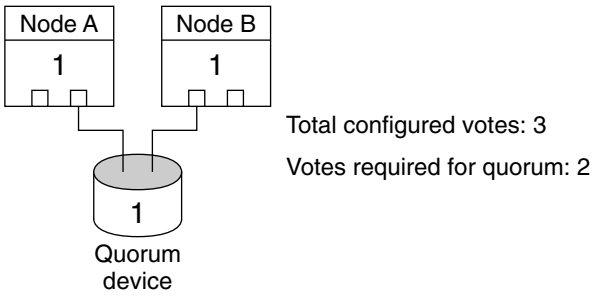
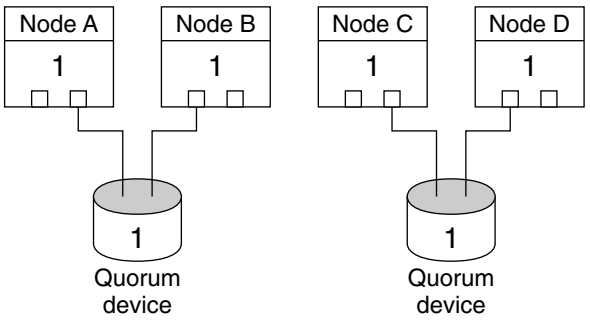


FIGURE 3–2 Two–Node Configuration

Quorum in Greater Than Two–Node Configurations

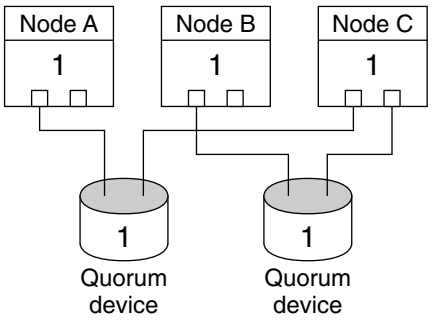
It is valid to configure a greater than two-node cluster without a quorum device. However, if you do so, you will not be able start the cluster without a majority of nodes in the cluster.



Total configured votes: 6

Votes required for quorum: 4

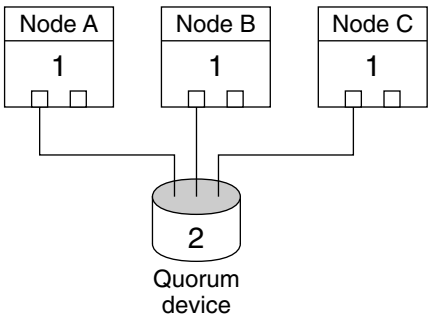
In this configuration, each pair must be available for either pair to survive.



Total configured votes: 5

Votes required for quorum: 3

In this configuration, usually applications are configured to run on Node A and Node B and use Node C as a hot spare.



Total configured votes: 5

Votes required for quorum: 3

In this configuration, the combination of any one or more nodes and the quorum device can form a cluster.

Atypical Quorum Configurations

Figure 3–3 assumes you are running mission-critical applications (Oracle database for example) on `Node A` and `Node B`. If `Node A` and `Node B` are unavailable and cannot access shared data, you might want the entire cluster to be down. Otherwise, this configuration is suboptimal because it does not provide high availability.

For information about the best practice to which this exception relates, see "Adhering to Quorum Device Best Practices" on page 53.

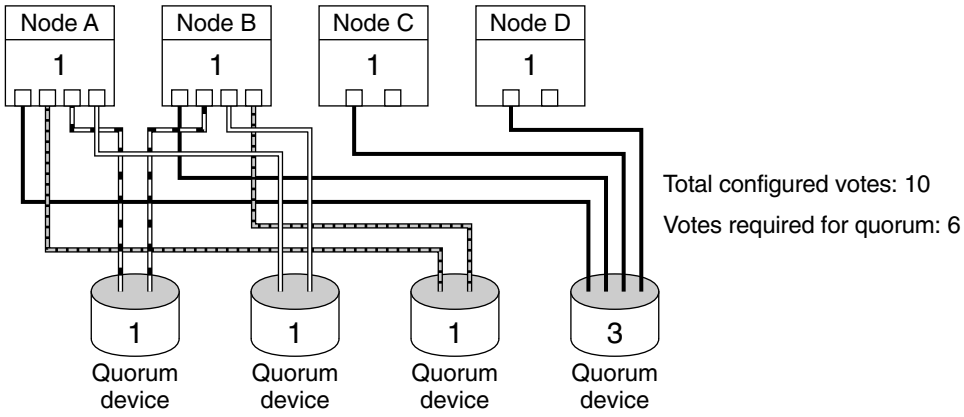
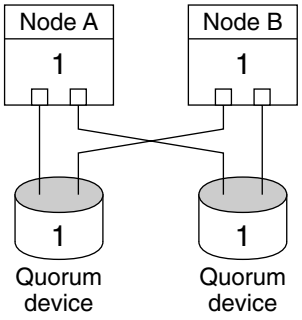


FIGURE 3–3 Atypical Configuration

Bad Quorum Configurations

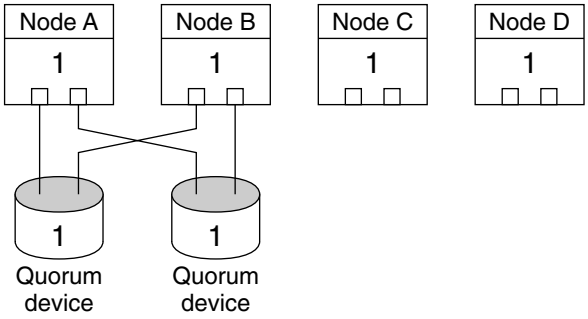
For examples of recommended quorum configurations, see "Recommended Quorum Configurations" on page 55.



Total configured votes: 4

Votes required for quorum: 3

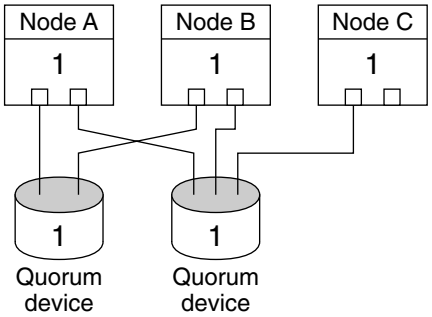
This configuration violates the best practice that quorum device votes should be strictly less than votes of nodes.



Total configured votes: 6

Votes required for quorum: 4

This configuration violates the best practice that you should not add quorum devices to make total votes even. This configuration does not add availability.



Total configured votes: 5

Votes required for quorum: 3

This configuration violates the best practice that quorum device votes should be strictly less than votes of nodes.

Data Services

The term *data service* describes a third-party application, such as Sun Java System Web Server (formerly Sun Java System Web Server) or, for SPARC based clusters, Oracle, that has been configured to run on a cluster rather than on a single server. A data service consists of an application, specialized Sun Cluster configuration files, and Sun Cluster management methods that control the following actions of the application.

- Start
- Stop
- Monitor and take corrective measures
- For information about data service types, see "Data Services" in *Sun Cluster Overview for Solaris OS*.

Figure 3–4 compares an application that runs on a single application server (the single-server model) to the same application running on a cluster (the clustered-server model). Note that from the user's perspective, there is no difference between the two configurations except that the clustered application might run faster and will be more highly available.

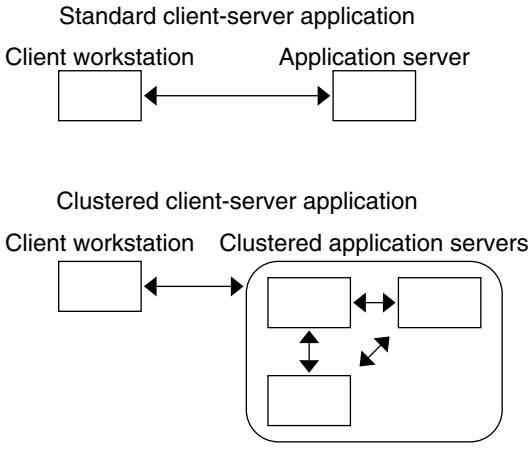


FIGURE 3–4 Standard Versus Clustered Client/Server Configuration

In the single-server model, you configure the application to access the server through a particular public network interface (a hostname). The hostname is associated with that physical server.

In the clustered-server model, the public network interface is a *logical hostname* or a *shared address*. The term *network resources* is used to refer to both logical hostnames and shared addresses.

Some data services require you to specify either logical hostnames or shared addresses as the network interfaces—they are not interchangeable. Other data services allow you to specify either logical hostnames or shared addresses. Refer to the installation and configuration for each data service for details on the type of interface you must specify.

A network resource is not associated with a specific physical server—it can migrate between physical servers.

A network resource is initially associated with one node, the *primary*. If the primary fails, the network resource, and the application resource, fails over to a different cluster node (a secondary). When the network resource fails over, after a short delay, the application resource continues to run on the secondary.

Figure 3–5 compares the single-server model with the clustered-server model. Note that in the clustered-server model, a network resource (logical hostname, in this example) can move between two or more of the cluster nodes. The application is configured to use this logical hostname in place of a hostname associated with a particular server.

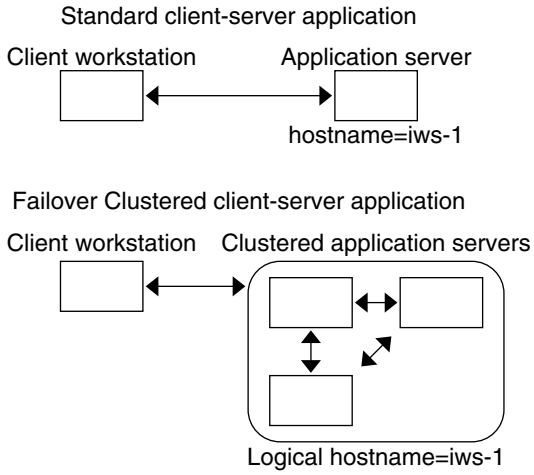


FIGURE 3–5 Fixed Hostname Versus Logical Hostname

A shared address is also initially associated with one node. This node is called the global interface node . A shared address is used as the single network interface to the cluster. It is known as the *global interface*.

The difference between the logical hostname model and the scalable service model is that in the latter, each node also has the shared address actively configured up on its loopback interface. This configuration makes it possible to have multiple instances of a data service active on several nodes simultaneously. The term “scalable service” means that you can add more CPU power to the application by adding additional cluster nodes and the performance will scale.

If the global interface node fails, the shared address can be brought up on another node that is also running an instance of the application (thereby making this other node the new global interface node). Or, the shared address can fail over to another cluster node that was not previously running the application.

Figure 3–6 compares the single-server configuration with the clustered-scalable service configuration. Note that in the scalable service configuration, the shared address is present on all nodes. Similar to how a logical hostname is used for a failover data service, the application is configured to use this shared address in place of a hostname associated with a particular server.

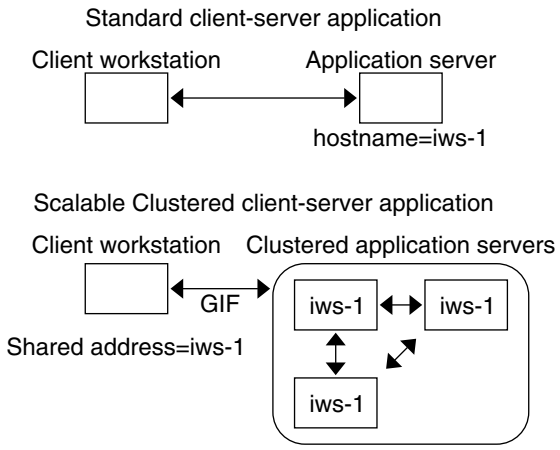


FIGURE 3–6 Fixed Hostname Versus Shared Address

Data Service Methods

The Sun Cluster software supplies a set of service management methods. These methods run under the control of the Resource Group Manager (RGM), which uses them to start, stop, and monitor the application on the cluster nodes. These methods, along with the cluster framework software and multihost devices, enable applications to become failover or scalable data services.

The RGM also manages resources in the cluster, including instances of an application and network resources (logical hostnames and shared addresses).

In addition to Sun Cluster software-supplied methods, the SunPlex system also supplies an API and several data service development tools. These tools enable application programmers to develop the data service methods needed to make other applications run as highly available data services with the Sun Cluster software.

Failover Data Services

If the node on which the data service is running (the primary node) fails, the service is migrated to another working node without user intervention. Failover services use a *failover resource group*, which is a container for application instance resources and network resources (*logical hostnames*). Logical hostnames are IP addresses that can be configured up on one node, and later, automatically configured down on the original node and configured up on another node.

For failover data services, application instances run only on a single node. If the fault monitor detects an error, it either attempts to restart the instance on the same node, or to start the instance on another node (failover), depending on how the data service has been configured.

Scalable Data Services

The scalable data service has the potential for active instances on multiple nodes. Scalable services use two resource groups: a *scalable resource group* to contain the application resources and a failover resource group to contain the network resources (*shared addresses*) on which the scalable service depends. The scalable resource group can be online on multiple nodes, so multiple instances of the service can be running at once. The failover resource group that hosts the shared address is online on only one node at a time. All nodes hosting a scalable service use the same shared address to host the service.

Service requests come into the cluster through a single network interface (the global interface) and are distributed to the nodes based on one of several predefined algorithms set by the *load-balancing policy*. The cluster can use the load-balancing policy to balance the service load between several nodes. Note that there can be multiple global interfaces on different nodes hosting other shared addresses.

For scalable services, application instances run on several nodes simultaneously. If the node that hosts the global interface fails, the global interface fails over to another node. If an application instance running fails, the instance attempts to restart on the same node.

If an application instance cannot be restarted on the same node, and another unused node is configured to run the service, the service fails over to the unused node. Otherwise, it continues to run on the remaining nodes, possibly causing a degradation of service throughput.

Note – TCP state for each application instance is kept on the node with the instance, not on the global interface node. Therefore, failure of the global interface node does not affect the connection.

Figure 3–7 shows an example of failover and a scalable resource group and the dependencies that exist between them for scalable services. This example shows three resource groups. The failover resource group contains application resources for highly available DNS, and network resources used by both highly available DNS and highly available Apache Web Server (available for use in SPARC-based clusters only). The scalable resource groups contain only application instances of the Apache Web Server. Note that resource group dependencies exist between the scalable and failover resource groups (solid lines) and that all of the Apache application resources are dependent on the network resource `schost-2`, which is a shared address (dashed lines).

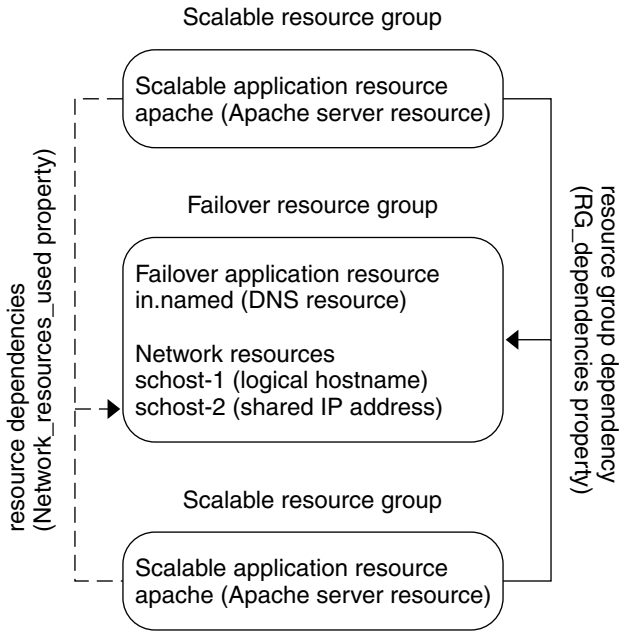


FIGURE 3–7 SPARC: Failover and Scalable Resource Group Example

Load-Balancing Policies

Load balancing improves performance of the scalable service, both in response time and in throughput.

There are two classes of scalable data services: *pure* and *sticky*. A pure service is one where any instance of it can respond to client requests. A sticky service is one where a client sends requests to the same instance. Those requests are not redirected to other instances.

A pure service uses a weighted load-balancing policy. Under this load-balancing policy, client requests are by default uniformly distributed over the server instances in the cluster. For example, in a three-node cluster, let us suppose that each node has the weight of 1. Each node will service 1/3 of the requests from any client on behalf of that service. Weights can be changed at any time by the administrator through the `scrgadm(1M)` command interface or through the SunPlex Manager GUI.

A sticky service has two flavors, *ordinary sticky* and *wildcard sticky*. Sticky services allow concurrent application-level sessions over multiple TCP connections to share in-state memory (application session state).

Ordinary sticky services permit a client to share state between multiple concurrent TCP connections. The client is said to be "sticky" with respect to that server instance listening on a single port. The client is guaranteed that all of his requests go to the same server instance, provided that instance remains up and accessible and the load balancing policy is not changed while the service is online.

For example, a web browser on the client connects to a shared IP address on port 80 using three different TCP connections, but the connections are exchanging cached session information between them at the service.

A generalization of a sticky policy extends to multiple scalable services exchanging session information behind the scenes at the same instance. When these services exchange session information behind the scenes at the same instance, the client is said to be "sticky" with respect to multiple server instances on the same node listening on different ports.

For example, a customer on an e-commerce site fills his shopping cart with items using ordinary HTTP on port 80, but switches to SSL on port 443 to send secure data in order to pay by credit card for the items in the cart.

Wildcard sticky services use dynamically assigned port numbers, but still expect client requests to go to the same node. The client is "sticky wildcard" over ports with respect to the same IP address.

A good example of this policy is passive mode FTP. A client connects to an FTP server on port 21 and is then informed by the server to connect back to a listener port server in the dynamic port range. All requests for this IP address are forwarded to the same node that the server informed the client through the control information.

Note that for each of these sticky policies the weighted load-balancing policy is in effect by default, thus, a client's initial request is directed to the instance dictated by the load balancer. After the client has established an affinity for the node where the instance is running, then future requests are directed to that instance as long as the node is accessible and the load balancing policy is not changed.

Additional details of the specific load balancing policies are discussed below.

- Weighted. The load is distributed among various nodes according to specified weight values. This policy is set using the `LB_WEIGHTED` value for the `Load_balancing_weights` property. If a weight for a node is not explicitly set, the weight for that node defaults to one.

  The weighted policy redirects a certain percentage of the traffic from clients to a particular node. Given X=weight and A=the total weights of all active nodes, an active node can expect approximately X/A of the total new connections to be directed to the active node, when the total number of connections is large enough. This policy does not address individual requests.

  Note that this policy is not round robin. A round-robin policy would always cause each request from a client to go to a different node: the first request to node 1, the second request to node 2, and so on.
- Sticky. In this policy, the set of ports is known at the time the application resources are configured. This policy is set using the `LB_STICKY` value for the `Load_balancing_policy` resource property.
- Sticky-wildcard. This policy is a superset of the ordinary "sticky" policy. For a scalable service identified by the IP address, ports are assigned by the server (and are not known in advance). The ports might change. This policy is set using the `LB_STICKY_WILD` value for the `Load_balancing_policy` resource property.

Failback Settings

Resource groups fail over from one node to another. When this occurs, the original secondary becomes the new primary. The failback settings specify the actions that will take place when the original primary comes back online. The options are to have the original primary become the primary again (failback) or to allow the current primary to remain. You specify the option you want using the `Failback` resource group property setting.

In certain instances, if the original node hosting the resource group is failing and rebooting repeatedly, setting failback might result in reduced availability for the resource group.

Data Services Fault Monitors

Each SunPlex data service supplies a fault monitor that periodically probes the data service to determine its health. A fault monitor verifies that the application daemon(s) are running and that clients are being served. Based on the information returned by probes, predefined actions such as restarting daemons or causing a failover, can be initiated.

Developing New Data Services

Sun supplies configuration files and management methods templates that enable you to make various applications operate as failover or scalable services within a cluster. If the application that you want to run as a failover or scalable service is not one that is currently offered by Sun, you can use an API or the DSET API to configure it to run as a failover or scalable service.

There is a set of criteria for determining whether an application can become a failover service. The specific criteria is described in the SunPlex documents that describe the APIs you can use for your application.

Here, we present some guidelines to help you understand whether your service can take advantage of the scalable data services architecture. Review the section, "Scalable Data Services" on page 63 for more general information on scalable services.

New services that satisfy the following guidelines may make use of scalable services. If an existing service doesn't follow these guidelines exactly, portions may need to be rewritten so that the service complies with the guidelines.

A scalable data service has the following characteristics. First, such a service is composed of one or more server *instances*. Each instance runs on a different node of the cluster. Two or more instances of the same service cannot run on the same node.

Second, if the service provides an external logical data store, then concurrent access to this store from multiple server instances must be synchronized to avoid losing updates or reading data as it's being changed. Note that we say "external" to distinguish the store from in-memory state, and "logical" because the store appears as a single entity, although it may itself be replicated. Furthermore, this logical data store has the property that whenever any server instance updates the store, that update is immediately seen by other instances.

The SunPlex system provides such an external storage through its cluster file system and its global raw partitions. As an example, suppose a service writes new data to an external log file or modifies existing data in place. When multiple instances of this service run, each has access to this external log, and each may simultaneously access this log. Each instance must synchronize its access to this log, or else the instances interfere with each other. The service could use ordinary Solaris file locking via `fcntl(2)` and `lockf(3C)` to achieve the desired synchronization.

Another example of this type of store is a back-end database, such as highly available Oracle or Oracle Real Application Clusters for SPARC-based clusters. Note that this type of back-end database server provides built-in synchronization using database query or update transactions, and so multiple server instances need not implement their own synchronization.

An example of a service that is not a scalable service in its current incarnation is Sun's IMAP server. The service updates a store, but that store is private and when multiple IMAP instances write to this store, they overwrite each other because the updates are not synchronized. The IMAP server must be rewritten to synchronize concurrent access.

Finally, note that instances may have private data that's disjoint from the data of other instances. In such a case, the service need not concern itself with synchronizing concurrent access because the data is private, and only that instance can manipulate it. In this case, you must be careful not to store this private data under the cluster file system because it has the potential to become globally accessible.

Data Service API and Data Service Development Library API

The SunPlex system provides the following to make applications highly available:

- Data services supplied as part of the SunPlex system
- A data service API
- A data service development library API
- A "generic" data service

The *Sun Cluster Data Services Planning and Administration Guide for Solaris OS* describes how to install and configure the data services supplied with the SunPlex system. The *Sun Cluster 3.1 9/04 Software Collection for Solaris OS (SPARC Platform Edition)* describes how to instrument other applications to be highly available under the Sun Cluster framework.

The Sun Cluster APIs enable application programmers to develop fault monitors and scripts that start and stop data services instances. With these tools, an application can be instrumented to be a failover or a scalable data service. In addition, the SunPlex system provides a "generic" data service that can be used to quickly generate an application's required start and stop methods to make it run as a failover or scalable service.

Using the Cluster Interconnect for Data Service Traffic

A cluster must have multiple network connections between nodes, forming the cluster interconnect. The clustering software uses multiple interconnects both for high availability and to improve performance. For internal traffic (for example, file system data or scalable services data), messages are striped across all available interconnects in a round-robin fashion.

The cluster interconnect is also available to applications, for highly available communication between nodes. For example, a distributed application might have components running on different nodes that need to communicate. By using the cluster interconnect rather than the public transport, these connections can withstand the failure of an individual link.

To use the cluster interconnect for communication between nodes, an application must use the private hostnames configured when the cluster was installed. For example, if the private hostname for node 1 is `clusternode1-priv`, use that name to communicate over the cluster interconnect to node 1. TCP sockets opened using this name are routed over the cluster interconnect and can be transparently re-routed in the event of network failure.

Note that because the private hostnames can be configured during installation, the cluster interconnect can use any name chosen at that time. The actual name can be obtained from `scha_cluster_get`(3HA) with the `scha_privatelink_hostname_node` argument.

For application-level use of the cluster interconnect, a single interconnect is used between each pair of nodes, but separate interconnects are used for different node pairs, if possible. For example, consider an application running on three SPARC based nodes and communicating over the cluster interconnect. Communication between nodes 1 and 2 might take place on interface `hme0`, while communication between nodes 1 and 3 might take place on interface `qfe1`. That is, application communication between any two nodes is limited to a single interconnect, while internal clustering communication is striped over all interconnects.

Note that the application shares the interconnect with internal clustering traffic, so the bandwidth available to the application depends on the bandwidth used for other clustering traffic. In the event of a failure, internal traffic can round-robin over the remaining interconnects, while application connections on a failed interconnect can switch to a working interconnect.

Two types of addresses support the cluster interconnect, and `gethostbyname`(3N) on a private hostname normally returns two IP addresses. The first address is called the *logical pairwise address*, and the second address is called the *logical pernode address*.

A separate logical pairwise address is assigned to each pair of nodes. This small logical network supports failover of connections. Each node is also assigned a fixed pernode address. That is, the logical pairwise addresses for `clusternode1-priv` are different on each node, while the logical pernode address for `clusternode1-priv` is the same on each node. A node does not have a pairwise address to itself, however, so `gethostbyname`(`clusternode1-priv`) on node 1 returns only the logical pernode address.

Note that applications accepting connections over the cluster interconnect and then verifying the IP address for security reasons must check against all IP addresses returned from `gethostbyname`, not just the first IP address.

If you need consistent IP addresses in your application at all points, configure the application to bind to the pernode address on both the client and the server side so that all connections can appear to come and go from the pernode address.

Resources, Resource Groups, and Resource Types

Data services utilize several types of *resources*: applications such as Sun Java System Web Server (formerly Sun Java System Web Server) or Apache Web Server use network addresses (logical hostnames and shared addresses) upon which the applications depend. Application and network resources form a basic unit that is managed by the RGM.

Data services are resource types. For example, Sun Cluster HA for Oracle is the resource type `SUNW.oracle-server` and Sun Cluster HA for Apache is the resource type `SUNW.apache`.

Note – The resource type `SUNW.oracle-server` is used in only SPARC based clusters

---

A resource is an instantiation of a *resource type* that is defined cluster wide. There are several resource types defined.

Network resources are either `SUNW.LogicalHostname` or `SUNW.SharedAddress` resource types. These two resource types are pre-registered by the Sun Cluster software.

The `SUNW.HAStorage` and `HAStoragePlus` resource types are used to synchronize the startup of resources and disk device groups upon which the resources depend. It ensures that before a data service starts, the paths to cluster file system mount points, global devices, and device group names are available. For more information, see "Synchronizing the Startups Between Resource Groups and Disk Device Groups" in the *Data Services Installation and Configuration Guide*. (The `HAStoragePlus` resource type became available in Sun Cluster 3.0 5/02 and added another feature, enabling local file systems to be highly available. For more information on this feature, see "`HAStoragePlus` Resource Type" on page 45.)

RGM-managed resources are placed into groups, called *resource groups*, so that they can be managed as a unit. A resource group is migrated as a unit if a failover or switchover is initiated on the resource group.

Note – When you bring a resource group containing application resources online, the application is started. The data service start method waits until the application is up and running before exiting successfully. The determination of when the application is up and running is accomplished the same way the data service fault monitor determines that a data service is serving clients. Refer to the *Sun Cluster Data Services Planning and Administration Guide for Solaris OS* for more information on this process.

Resource Group Manager (RGM)

The RGM controls data services (applications) as resources, which are managed by *resource type* implementations. These implementations are either supplied by Sun or created by a developer with a generic data service template, the Data Service Development Library API (DSDL API), or the Resource Management API (RMAPI). The cluster administrator creates and manages resources in containers called *resource groups*. The RGM stops and starts resource groups on selected nodes in response to cluster membership changes.

The RGM acts on *resources* and *resource groups.* RGM actions cause resources and resource groups to move between online and offline states. A complete description of the states and settings that can be applied to resources and resource groups is in the section "Resource and Resource Group States and Settings" on page 71. Refer to "Resources, Resource Groups, and Resource Types" on page 70 for information about how to launch a resource management project under RGM control.

Resource and Resource Group States and Settings

An administrator applies static settings to resources and resource groups. These settings can only be changed through administrative actions. The RGM moves resource groups between dynamic "states." These settings and states are described in the following list.

- Managed or unmanaged – These are cluster-wide settings that apply only to resource groups. Resource groups are managed by the RGM. The `scrgadm(1M)` command can be used to cause the RGM to manage or to unmanage a resource group. These settings do not change with a cluster reconfiguration.

  When a resource group is first created, it is unmanaged. It must be managed before any resources placed in the group can become active.

  In some data services, for example a scalable web server, work must be done prior to starting up network resources and after they are stopped. This work is done by initialization (INIT) and finish (FINI) data service methods. The INIT methods only run if the resource group in which the resources reside is in the managed state.

  When a resource group is moved from unmanaged to managed, any registered INIT methods for the group are run on the resources in the group.

When a resource group is moved from managed to unmanaged, any registered FINI methods are called to perform cleanup.

The most common use of INIT and FINI methods are for network resources for scalable services, but they can be used for any initialization or cleanup work that is not done by the application.

- Enabled or disabled – These are cluster-wide settings that apply to resources. The `scrgadm(1M)` command can be used to enable or disable a resource. These settings do not change with a cluster reconfiguration.

  The normal setting for a resource is that it is enabled and actively running in the system.

  If for some reason, you want to make the resource unavailable on all cluster nodes, you disable the resource. A disabled resource is not available for general use.

- Online or offline – These are dynamic states that apply to both resource and resource groups.

  These states change as the cluster transitions through cluster reconfiguration steps during switchover or failover. They can also be changed through administrative actions. The `scswitch(1M)` can be used to change the online or offline state of a resource or resource group.

  A failover resource or resource group can only be online on one node at any time. A scalable resource or resource group can be online on some nodes and offline on others. During a switchover or failover, resource groups and the resources within them are taken offline on one node and then brought online on another node.

  If a resource group is offline then all of its resources are offline. If a resource group is online, then all of its enabled resources are online.

  Resource groups can contain several resources, with dependencies between resources. These dependencies require that the resources be brought online and offline in a particular order. The methods used to bring resources online and offline might take different amounts of time for each resource. Because of resource dependencies and start and stop time differences, resources within a single resource group can have different online and offline states during a cluster reconfiguration.

Resource and Resource Group Properties

You can configure property values for resources and resource groups for your SunPlex data services. Standard properties are common to all data services. Extension properties are specific to each data service. Some standard and extension properties are configured with default settings so that you do not have to modify them. Others need to be set as part of the process of creating and configuring resources. The documentation for each data service specifies which resource properties can be set and how to set them.

The standard properties are used to configure resource and resource group properties that are usually independent of any particular data service. For the set of standard properties, see "Standard Properties" in *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*.

The RGM extension properties provide information such as the location of application binaries and configuration files. You modify extension properties as you configure your data services. The set of extension properties is described in the individual guide for the data service.

Data Service Project Configuration

Data services may be configured to launch under a Solaris project name when brought online using the RGM. The configuration associates a resource or resource group managed by the RGM with a Solaris project ID. The mapping from your resource or resource group to a project ID gives you the ability to use sophisticated controls that are available in the Solaris environment to manage workloads and consumption within your cluster.

Note – You can perform this configuration only if you are running the current release of Sun Cluster software with Solaris 9.

Using the Solaris management functionality in a cluster environment enables you to ensure that your most important applications are given priority when sharing a node with other applications. Applications might share a node if you have consolidated services or because applications have failed over. Use of the management functionality described herein might improve availability of a critical application by preventing other low priority applications from over-consuming system supplies such as CPU time.

Note – The Solaris documentation of this feature describes CPU time, processes, tasks and similar components as 'resources'. Meanwhile, Sun Cluster documentation uses the term 'resources' to describe entities that are under the control of the RGM. The following section will use the term 'resource' to refer to Sun Cluster entities under the control of the RGM and use the term 'supplies' to refer to CPU time, processes, and tasks.

This section provides a conceptual description of configuring data services to launch processes in a specified Solaris 9 `project`(4). This section also describes several failover scenarios and suggestions for planning to use the management functionality

provided by the Solaris environment. For detailed conceptual and procedural documentation of the management feature, refer to *System Administration Guide: Resource Management and Network Services* in the *Solaris 9 System Administrator Collection.*

When configuring resources and resource groups to use Solaris management functionality in a cluster, consider using the following high-level process:

1. Configure applications as part of the resource.
2. Configure resources as part of a resource group.
3. Enable resources in the resource group.
4. Make the resource group managed.
5. Create a Solaris project for your resource group.
6. Configure standard properties to associate the resource group name with the project you created in step 5.
7. Bring the resource group online.

To configure the standard `Resource_project_name` or `RG_project_name` properties to associate the Solaris project ID with the resource or resource group, use the `-y` option with the `scrgadm`(1M) command. Set the property values to the resource or resource group. See "Standard Properties" in *Sun Cluster Data Services Planning and Administration Guide for Solaris OS* for property definitions. Refer to `r_properties`(5) and `rg_properties`(5) for property descriptions.

The specified project name must exist in the projects database (`/etc/project`) and the root user must be configured as a member of the named project. Refer to "Projects and Tasks" in *System Administration Guide: Resource Management and Network Services* in the *Solaris 9 System Administrator Collection* for conceptual information about the project name database. Refer to `project`(4) for a description of project file syntax.

When the RGM brings resources or resource groups online, it launches the related processes under the project name.

Note – Users can associate the resource or resource group with a project at any time. However, the new project name is not effective until the resource or resource group is taken offline and brought back online using the RGM.

Launching resources and resource groups under the project name enables you to configure the following features to manage system supplies across your cluster.

- Extended Accounting – Provides a flexible way to record consumption on a task or process basis. Extended accounting enables you to examine historical usage and make assessments of capacity requirements for future workloads.
- Controls – Provide a mechanism for constraint on system supplies. Processes, tasks, and projects can be prevented from consuming large amounts of specified system supplies.

- Fair Share Scheduling (FSS) – Provides the ability to control the allocation of available CPU time among workloads, based on their importance. Workload importance is expressed by the number of shares of CPU time that you assign to each workload. Refer to `dispadmin`(1M) for a command line description of setting FSS as your default scheduler. See also `priocntl`(1), `ps`(1), and `FSS`(7) for more information.
- Pools – Provide the ability to use partitions for interactive applications according to the application's requirements. Pools can be used to partition a server that supports a number of different software applications. The use of pools results in a more predictable response for each application.

Determining Requirements for Project Configuration

Before you configure data services to use the controls provided by Solaris in a Sun Cluster environment, you must decide how you want to control and track resources across switchovers or failovers. Consider identifying dependencies within your cluster before configuring a new project. For example, resources and resource groups depend on disk device groups. Use the `nodelist`, `failback`, `maximum_primaries` and `desired_primaries` resource group properties, configured with `scrgadm`(1M) to identify nodelist priorities for your resource group. Refer to "Relationship Between Resource Groups and Disk Device Groups" in *Sun Cluster Data Services Planning and Administration Guide for Solaris OS* for a brief discussion of the node list dependencies between resource groups and disk device groups. For detailed property descriptions, refer to `rg_properties`(5).

Use the `preferenced` and `failback` properties configured with `scrgadm`(1M) and `scsetup`(1M) to determine disk device group nodelist priorities. For procedural information, see "How To Change Disk Device Properties" in "Administering Disk Device Groups" in *Sun Cluster System Administration Guide for Solaris OS*. Refer to "The SunPlex System Hardware and Software Components" on page 19 for conceptual information about node configuration and the behavior of failover and scalable data services.

If you configure all cluster nodes identically, usage limits are enforced identically on primary and secondary nodes. The configuration parameters of projects need not be identical for all applications in the configuration files on all nodes. All projects associated with the application must at least be accessible by the project database on all potential masters of that application. Suppose that Application 1 is mastered by *phys-schost-1* but could potentially be switched over or failed over to *phys-schost-2* or *phys-schost-3*. The project associated with Application 1 must be accessible on all three nodes (*phys-schost-1*, *phys-schost-2*, and *phys-schost-3*).

Note – Project database information can be a local `/etc/project` database file or may be stored in the NIS map or the LDAP directory service.

The Solaris environment allows for flexible configuration of usage parameters, and few restrictions are imposed by Sun Cluster. Configuration choices depend on the needs of the site. Consider the general guidelines in the following sections before configuring your systems.

Setting Per-Process Virtual Memory Limits

Set the `process.max-address-space` control to limit virtual memory on a per-process basis. Refer to `rctladm`(1M) for detailed information about setting the `process.max-address-space` value.

When using management controls with Sun Cluster, configure memory limits appropriately to prevent unnecessary failover of applications and a "ping-pong" effect of applications. In general:

- Do not set memory limits too low.

  When an application reaches its memory limit, it might fail over. This guideline is especially important for database applications, when reaching a virtual memory limit can have unexpected consequences.
- Do not set memory limits identically on primary and secondary nodes.

  Identical limits can cause a ping-pong effect when an application reaches its memory limit and fails over to a secondary node with an identical memory limit. Set the memory limit slightly higher on the secondary node. The difference in memory limits helps prevent the ping-pong scenario and gives the system administrator a period of time in which to adjust the parameters as necessary.
- Do use the resource management memory limits for load-balancing.

  For example, you can use memory limits to prevent an errant application from consuming excessive swap space.

Failover Scenarios

You can configure management parameters so that the allocation in the project configuration (`/etc/project`) works in normal cluster operation and in switchover or failover situations.

The following sections are example scenarios.

- The first two sections, "Two-Node Cluster With Two Applications" and "Two-Node Cluster With Three Applications," show failover scenarios for entire nodes.

- The section “Failover of Resource Group Only“ illustrates failover operation for an application only.

In a cluster environment, an application is configured as part of a resource and a resource is configured as part of a resource group (RG). When a failure occurs, the resource group along with its associated applications, fails over to another node. In the following examples the resources are not shown explicitly. Assume that each resource has only one application.

Note – Failover occurs in the preferenced nodelist order that is set in the RGM.

The following examples have these constraints:

- Application 1 (App-1) is configured in resource group RG-1.
- Application 2 (App-2) is configured in resource group RG-2.
- Application 3 (App-3) is configured in resource group RG-3.

Although the numbers of assigned shares remain the same, the percentage of CPU time allocated to each application changes after failover. This percentage depends on the number of applications that are running on the node and the number of shares that are assigned to each active application.

In these scenarios, assume the following configurations.

- All applications are configured under a common project.
- Each resource has only one application.
- The applications are the only active processes on the nodes.
- The projects databases are configured the same on each node of the cluster.

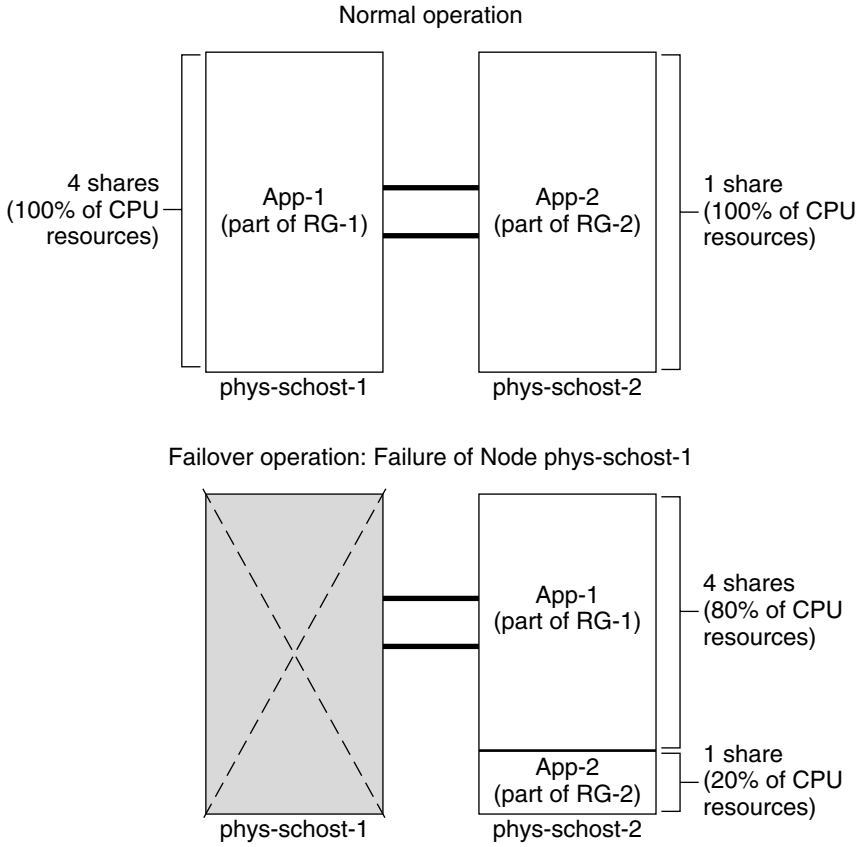
Two-Node Cluster With Two Applications

You can configure two applications on a two-node cluster to ensure that each physical host (*phys-schost-1*, *phys-schost-2*) acts as the default master for one application. Each physical host acts as the secondary node for the other physical host. All projects associated with Application 1 and Application 2 must be represented in the projects database files on both nodes. When the cluster is running normally, each application is running on its default master, where it is allocated all CPU time by the management facility.

After a failover or switchover occurs, both applications run on a single node where they are allocated shares as specified in the configuration file. For example, this entry in the `/etc/project` file specifies that Application 1 is allocated 4 shares and Application 2 is allocated 1 share.

```
Prj_1:100:project for App-1:root::project.cpu-shares=(privileged,4,none)
Prj_2:101:project for App-2:root::project.cpu-shares=(privileged,1,none)
```

The following diagram illustrates the normal and failover operations of this configuration. The number of shares that are assigned does not change. However, the percentage of CPU time available to each application can change, depending on the number of shares assigned to each process demanding CPU time.



Two-Node Cluster With Three Applications

On a two-node cluster with three applications, you can configure one physical host (*phys-schost-1*) as the default master of one application and the second physical host (*phys-schost-2*) as the default master for the remaining two applications. Assume the following example projects database file on every node. The projects database file does not change when a failover or switchover occurs.

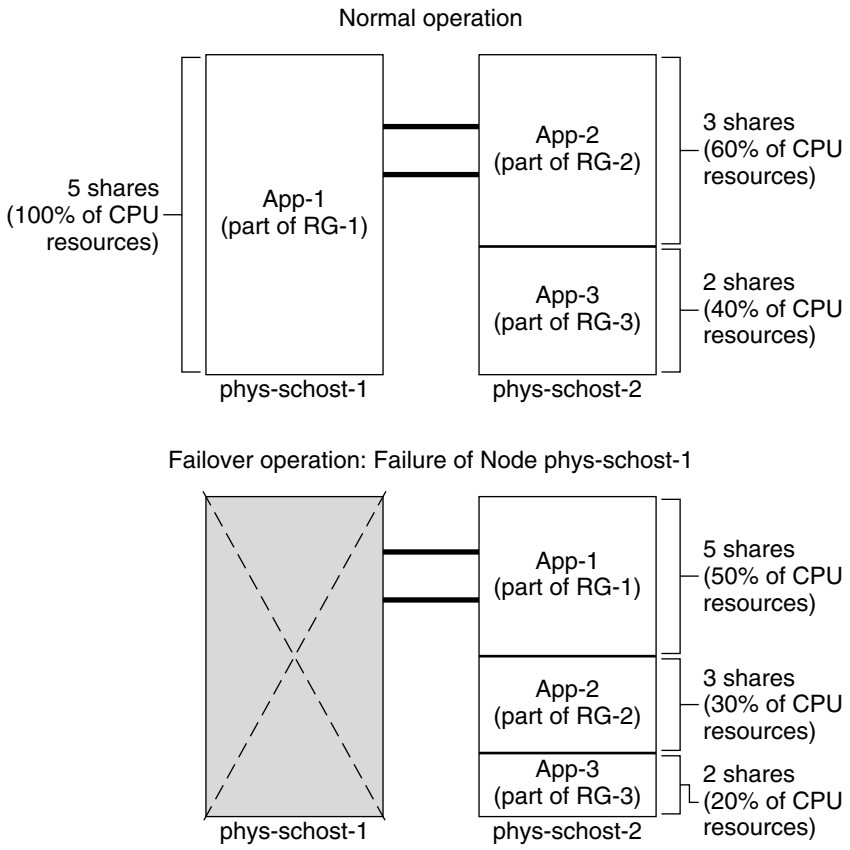
```
Prj_1:103:project for App-1:root::project.cpu-shares=(privileged,5,none)
Prj_2:104:project for App_2:root::project.cpu-shares=(privileged,3,none)
Prj_3:105:project for App_3:root::project.cpu-shares=(privileged,2,none)
```

When the cluster is running normally, Application 1 is allocated 5 shares on its default master, *phys-schost-1*. This number is equivalent to 100 percent of CPU time because it is the only application that demands CPU time on that node. Applications 2 and 3 are allocated 3 and 2 shares, respectively, on their default master, *phys-schost-2*. Application 2 would receive 60 percent of CPU time and Application 3 would receive 40 percent of CPU time during normal operation.

If a failover or switchover occurs and Application 1 is switched over to *phys-schost-2*, the shares for all three applications remain the same. However, the percentages of CPU resources are reallocated according to the projects database file.

- Application 1, with 5 shares, receives 50 percent of CPU.
- Application 2, with 3 shares, receives 30 percent of CPU.
- Application 3, with 2 shares, receives 20 percent of CPU.

The following diagram illustrates the normal operations and failover operations of this configuration.



Failover of Resource Group Only

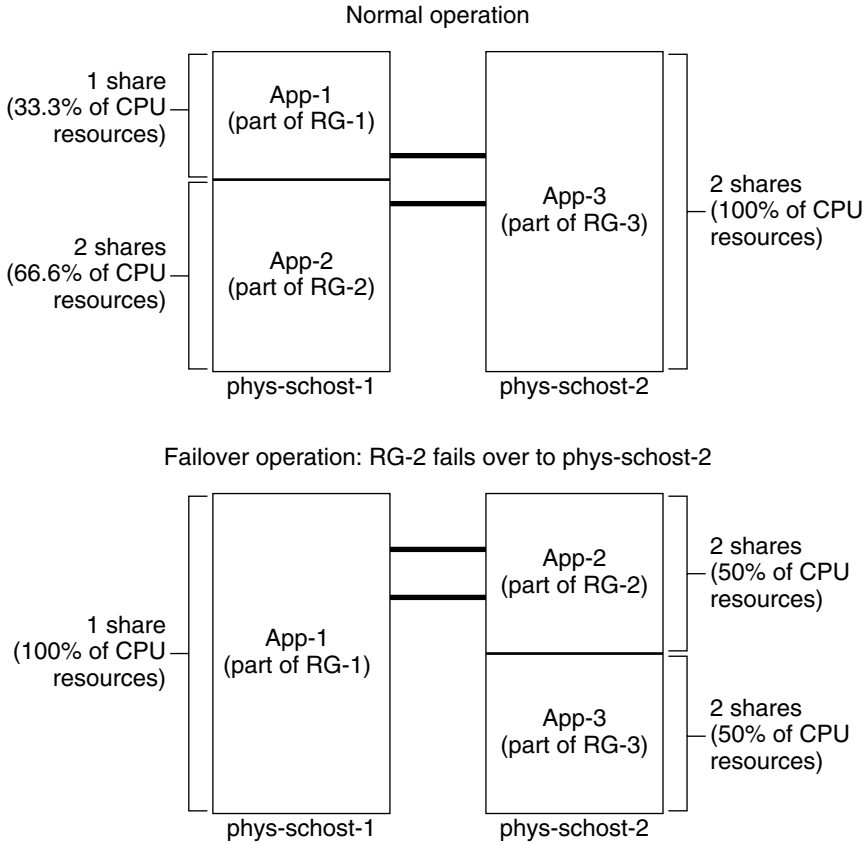
In a configuration in which multiple resource groups have the same default master, a resource group (and its associated applications) can fail over or be switched over to a secondary node. Meanwhile, the default master is running in the cluster.

Note – During failover, the application that fails over is allocated resources as specified in the configuration file on the secondary node. In this example, the projects database files on the primary and secondary nodes have the same configurations.

For example, this sample configuration file specifies that Application 1 is allocated 1 share, Application 2 is allocated 2 shares, and Application 3 is allocated 2 shares.

```
Prj_1:106:project for App_1:root::project.cpu-shares=(privileged,1,none)
Prj_2:107:project for App_2:root::project.cpu-shares=(privileged,2,none)
Prj_3:108:project for App_3:root::project.cpu-shares=(privileged,2,none)
```

The following diagram illustrates the normal and failover operations of this configuration, where RG-2, containing Application 2, fails over to *phys-schost-2*. Note that the number of shares assigned does not change. However, the percentage of CPU time available to each application can change, depending on the number of shares assigned to each application demanding CPU time.



Public Network Adapters and IP Network Multipathing

Clients make data requests to the cluster through the public network. Each cluster node is connected to at least one public network through a pair of public network adapters.

Solaris Internet Protocol (IP) Network Multipathing software on Sun Cluster provides the basic mechanism for monitoring public network adapters and failing over IP addresses from one adapter to another when a fault is detected. Each cluster node has its own IP Network Multipathing configuration, which can be different from that on other cluster nodes.

Public network adapters are organized into *IP multipathing groups* (multipathing groups). Each multipathing group has one or more public network adapters. Each adapter in a multipathing group can be active, or you can configure standby interfaces that are inactive unless there is a failover. The `in.mpathd` multipathing daemon uses a test IP address to detect failures and repairs. If a fault is detected on one of the adapters by the multipathing daemon, a failover occurs. All network access fails over from the faulted adapter to another functional adapter in the multipathing group, thereby maintaining public network connectivity for the node. If a standby interface was configured, the daemon chooses the standby interface. Otherwise, `in.mpathd` chooses the interface with the least number of IP addresses. Because the failover happens at the adapter interface level, higher-level connections such as TCP are not affected, except for a brief transient delay during the failover. When the failover of IP addresses completes successfully, gratuitous ARP broadcasts are sent. The connectivity to remote clients is therefore maintained.

Note – Because of the congestion recovery characteristics of TCP, TCP endpoints can suffer further delay after a successful failover as some segments could be lost during the failover, activating the congestion control mechanism in TCP.

Multipathing groups provide the building blocks for logical hostname and shared address resources. You can also create multipathing groups independently of logical hostname and shared address resources to monitor public network connectivity of cluster nodes. The same multipathing group on a node can host any number of logical hostname or shared address resources. For more information on logical hostname and shared address resources, see the *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*.

Note – The design of the IP Network Multipathing mechanism is meant to detect and mask adapter failures. The design is not intended to recover from an administrator using `ifconfig(1M)` to remove one of the logical (or shared) IP addresses. The Sun Cluster software views the logical and shared IP addresses as resources managed by the RGM. The correct way for an administrator to add or remove an IP address is to use `scrgadm`(1M) to modify the resource group containing the resource.

For more information about the Solaris implementation of IP Network Multipathing, see the appropriate documentation for the Solaris operating environment installed on your cluster.

Operating Environment Release	For Instructions, Go To...
Solaris 8 operating environment	*IP Network Multipathing Administration Guide*

Operating Environment Release	For Instructions, Go To...
Solaris 9 operating environment	"IP Network Multipathing Topics" in *System Administration Guide: IP Services*

SPARC: Dynamic Reconfiguration Support

Sun Cluster 3.1 4/04 support for the dynamic reconfiguration (DR) software feature is being developed in incremental phases. This section describes concepts and considerations for Sun Cluster 3.1 4/04 support of the DR feature.

Note that all of the requirements, procedures, and restrictions that are documented for the Solaris DR feature also apply to Sun Cluster DR support (except for the operating environment quiescence operation). Therefore, review the documentation for the Solaris DR feature *before* using the DR feature with Sun Cluster software. You should review in particular the issues that affect non-network IO devices during a DR detach operation. The *Sun Enterprise 10000 Dynamic Reconfiguration User Guide* and the *Sun Enterprise 10000 Dynamic Reconfiguration Reference Manual* (from the *Solaris 8 on Sun Hardware* or *Solaris 9 on Sun Hardware* collections) are both available for download from `http://docs.sun.com`.

SPARC: Dynamic Reconfiguration General Description

The DR feature allows operations, such as the removal of system hardware, in running systems. The DR processes are designed to ensure continuous system operation with no need to halt the system or interrupt cluster availability.

DR operates at the board level. Therefore, a DR operation affects all of the components on a board. Each board can contain multiple components, including CPUs, memory, and peripheral interfaces for disk drives, tape drives, and network connections.

Removing a board containing active components would result in system errors. Before removing a board, the DR subsystem queries other subsystems, such as Sun Cluster, to determine whether the components on the board are being used. If the DR subsystem finds that a board is in use, the DR remove-board operation is not done. Therefore, it is always safe to issue a DR remove-board operation since the DR subsystem rejects operations on boards containing active components.

The DR add-board operation is always safe also. CPUs and memory on a newly added board are automatically brought into service by the system. However, the system administrator must manually configure the cluster in order to actively use components that are on the newly added board.

Note – The DR subsystem has several levels. If a lower level reports an error, the upper level also reports an error. However, when the lower level reports the specific error, the upper level will report "Unknown error." System administrators should ignore the "Unknown error" reported by the upper level.

The following sections describe DR considerations for the different device types.

SPARC: DR Clustering Considerations for CPU Devices

Sun Cluster software will not reject a DR remove-board operation due to the presence of CPU devices.

When a DR add-board operation succeeds, CPU devices on the added board are automatically incorporated in system operation.

SPARC: DR Clustering Considerations for Memory

For the purposes of DR, there are two types of memory to consider. These two types differ only in usage. The actual hardware is the same for both types.

The memory used by the operating system is called the kernel memory cage. Sun Cluster software does not support remove-board operations on a board that contains the kernel memory cage and will reject any such operation. When a DR remove-board operation pertains to memory other than the kernel memory cage, Sun Cluster will not reject the operation.

When a DR add-board operation that pertains to memory succeeds, memory on the added board is automatically incorporated in system operation.

SPARC: DR Clustering Considerations for Disk and Tape Drives

Sun Cluster rejects DR remove-board operations on active drives in the primary node. DR remove-board operations can be performed on non-active drives in the primary node and on any drives in the secondary node. After the DR operation, cluster data access continues as before.

Note – Sun Cluster rejects DR operations that impact the availability of quorum devices. For considerations about quorum devices and the procedure for performing DR operations on them, see "SPARC: DR Clustering Considerations for Quorum Devices" on page 85.

See "Task Map: Dynamic Reconfiguration with Quorum Devices" in *Sun Cluster System Administration Guide for Solaris OS* for detailed instructions on how to perform these actions.

SPARC: DR Clustering Considerations for Quorum Devices

If the DR remove-board operation pertains to a board containing an interface to a device configured for quorum, Sun Cluster rejects the operation and identifies the quorum device that would be affected by the operation. You must disable the device as a quorum device before you can perform a DR remove-board operation.

See "Task Map: Dynamic Reconfiguration with Quorum Devices" in *Sun Cluster System Administration Guide for Solaris OS* for detailed instructions on how to perform these actions.

SPARC: DR Clustering Considerations for Cluster Interconnect Interfaces

If the DR remove-board operation pertains to a board containing an active cluster interconnect interface, Sun Cluster rejects the operation and identifies the interface that would be affected by the operation. You must use a Sun Cluster administrative tool to disable the active interface before the DR operation can succeed (also see the caution below).

See "Administering the Cluster Interconnects" in *Sun Cluster System Administration Guide for Solaris OS* for detailed instructions on how to perform these actions.

Caution – Sun Cluster requires that each cluster node has at least one functioning path to every other cluster node. Do not disable a private interconnect interface that supports the last path to any cluster node.

SPARC: DR Clustering Considerations for Public Network Interfaces

If the DR remove-board operation pertains to a board containing an active public network interface, Sun Cluster rejects the operation and identifies the interface that would be affected by the operation. Before removing a board with an active network interface present, all traffic on that interface must first be switched over to another functional interface in the multipathing group by using the `if_mpadm`(1M) command.

Caution – If the remaining network adapter fails while you are performing the DR remove operation on the disabled network adapter, availability is impacted. The remaining adapter has no place to fail over for the duration of the DR operation.

See "Administering the Public Network" in *Sun Cluster System Administration Guide for Solaris OS* for detailed instructions on how to perform a DR remove operation on a public network interface.

CHAPTER 4

Frequently Asked Questions

INDEXTERM-343

This chapter includes answers to the most frequently asked questions about the SunPlex system. The questions are organized by topic.

High Availability FAQs

- **What exactly is a highly available system?**

  The SunPlex system defines high availability (HA) as the ability of a cluster to keep an application up and running, even though a failure has occurred that would normally make a server system unavailable.

- **What is the process by which the cluster provides high availability?**

  Through a process known as failover, the cluster framework provides a highly available environment. Failover is a series of steps performed by the cluster to migrate data service resources from a failing node to another operational node in the cluster.

- **What is the difference between a failover and scalable data service?**

  There are two types of highly available data services, failover and scalable.

  A failover data service runs an application on only one primary node in the cluster at a time. Other nodes might run other applications, but each application runs on only a single node. If a primary node fails, the applications running on the failed node fail over to another node and continue running.

  A scalable service spreads an application across multiple nodes to create a single, logical service. Scalable services leverage the number of nodes and processors in the entire cluster on which they run.

For each application, one node hosts the physical interface to the cluster. This node is called a Global Interface (GIF) Node. There can be multiple GIF nodes in the cluster. Each GIF node hosts one or more logical interfaces that can be used by scalable services. These logical interfaces are called *global interfaces*. One GIF node hosts a global interface for all requests for a particular application and dispatches them to multiple nodes on which the application server is running. If the GIF node fails, the global interface fails over to a surviving node.

If any of the nodes on which the application is running fails, the application continues to run on the other nodes with some performance degradation until the failed node returns to the cluster.

File Systems FAQs

- **Can I run one or more of the cluster nodes as highly available NFS server(s) with other cluster nodes as clients?**

  No, do not do a loopback mount.

- **Can I use a cluster file system for applications that are not under Resource Group Manager control?**

  Yes. However, without RGM control, the applications need to be restarted manually after the failure of the node on which they are running.

- **Must all cluster file systems have a mount point under the /global directory?**

  No. However, placing cluster file systems under the same mount point, such as `/global`, enables better organization and management of these file systems.

- **What are the differences between using the cluster file system and exporting NFS file systems?**

  There are several differences:

  1. The cluster file system supports global devices. NFS does not support remote access to devices.
  2. The cluster file system has a global namespace. Only one mount command is required. With NFS, you must mount the file system on each node.
  3. The cluster file system caches files in more cases than does NFS. For example, when a file is being accessed from multiple nodes for read, write, file locks, async I/O.
  4. The cluster file system is built to exploit future fast cluster interconnects that provide remote DMA and zero-copy functions.
  5. If you change the attributes on a file (using `chmod(1M)`, for example) in a cluster file system, the change is reflected immediately on all nodes. With an exported NFS file system, this can take much longer.

- **The file system /global/.devices/node@<nodeID> appears on my cluster nodes. Can I use this file system to store data that I want to be highly available and global?**

  These file systems store the global device namespace. They are not intended for general use. While they are global, they are never accessed in a global manner--each node only accesses its own global device namespace. If a node is down, other nodes cannot access this namespace for the node that is down. These file systems are not highly available. They should not be used to store data that needs to be globally accessible or highly available.

Volume Management FAQs

- **Do I need to mirror all disk devices?**

  For a disk device to be considered highly available, it must be mirrored, or use RAID-5 hardware. All data services should use either highly available disk devices, or cluster file systems mounted on highly available disk devices. Such configurations can tolerate single disk failures.

- **Can I use one volume manager for the local disks (boot disk) and a different volume manager for the multihost disks?**

  SPARC: This configuration is supported with the Solaris Volume Manager software managing the local disks and VERITAS Volume Manager managing the multihost disks. No other combination is supported.

  x86: No, this configuration is not supported, as only Solaris Volume Manager is supported in x86 based clusters.

Data Services FAQs

- **What SunPlex data services are available?**

  The list of supported data services is included in "Supported Products" in *Sun Cluster 3.1 9/04 Release Notes for Solaris OS* .

- **What application versions are supported by SunPlex data services?**

  The list of supported application versions is included in "Supported Products" in *Sun Cluster 3.1 9/04 Release Notes for Solaris OS*.

- **Can I write my own data service?**

  Yes. See the "Data Service Development Library Reference" in *Sun Cluster Data Services Developer's Guide for Solaris OS* for more information.

- **When creating network resources, should I specify numeric IP addresses or hostnames?**

  The preferred method for specifying network resources is to use the UNIX hostname rather than the numeric IP address.

- **When creating network resources, what is the difference between using a logical hostname (a LogicalHostname resource) or a shared address (a SharedAddress resource)?**

  Except in the case of Sun Cluster HA for NFS, wherever the documentation calls for the use of a `LogicalHostname` resource in a `Failover` mode resource group, a `SharedAddress` resource or `LogicalHostname` resource may be used interchangeably. The use of a `SharedAddress` resource incurs some additional overhead because the cluster networking software is configured for a `SharedAddress` but not for a `LogicalHostname`.

  The advantage to using a `SharedAddress` is the case where you are configuring both scalable and failover data services, and want clients to be able to access both services using the same hostname. In this case, the `SharedAddress` resource(s) along with the failover application resource are contained in one resource group, while the scalable service resource is contained in a separate resource group and configured to use the `SharedAddress`. Both the scalable and failover services may then use the same set of hostnames/addresses which are configured in the `SharedAddress` resource.

Public Network FAQs

- **What public network adapters does the SunPlex system support?**

  Currently, the SunPlex system supports Ethernet (10/100BASE-T and 1000BASE-SX Gb) public network adapters. Because new interfaces might be supported in the future, check with your Sun sales representative for the most current information.

- **What is the role of the MAC address in failover?**

  When a failover occurs, new Address Resolution Protocol (ARP) packets are generated and broadcast to the world. These ARP packets contain the new MAC address (of the new physical adapter to which the node failed over) and the old IP address. When another machine on the network receives one of these packets, it flushes the old MAC-IP mapping from its ARP cache and uses the new one.

- **Does the SunPlex system support setting** `local-mac-address?=true`*?*

  Yes. In fact, IP Network Multipathing requires that `local-mac-address?` must be set to `true`.

  You can set `local-mac-address?` with `eeprom`(1M), at the OpenBoot PROM `ok` prompt in a SPARC based cluster, or with the SCSI utility that you optionally run after the BIOS boots in an x86 based cluster.

- **How much delay can I expect when IP Network Multipathing performs a switchover between adapters?**

  The delay could be several minutes. This is because when a IP Network Multipathing switchover is done, it involves sending out a gratuitous ARP. However, there is no guarantee that the router between the client and the cluster will use the gratuitous ARP. So, until the ARP cache entry for this IP address on the router times out, it is possible that it could use the stale MAC address.

- **How fast are failures of a network adapter detected?**

  The default failure detection time is 10 seconds. The algorithm tries to meet the failure detection time, but the actual time depends on the network load.

Cluster Member FAQs

- **Do all cluster members need to have the same root password?**

  You are not required to have the same root password on each cluster member. However, you can simplify administration of the cluster by using the same root password on all nodes.

- **Is the order in which nodes are booted significant?**

  In most cases, no. However, the boot order is important to prevent amnesia (refer to "About Failure Fencing" on page 51 for details on amnesia). For example, if node two was the owner of the quorum device and node one is down, and then you bring node two down, you must bring up node two before bringing back node one. This prevents you from accidentally bringing up a node with out of date cluster configuration information.

- **Do I need to mirror local disks in a cluster node?**

  Yes. Though this mirroring is not a requirement, mirroring the cluster node's disks precludes against a non-mirrored disk failure taking down the node. The downside to mirroring a cluster node's local disks is more system administration overhead.

- **What are the cluster member backup issues?**

  You can use several backup methods for a cluster. One method is to have a node as the backup node with a tape drive/library attached. Then use the cluster file system to back up the data. Do not connect this node to the shared disks.

  See the "Backing Up and Restoring a Cluster" in *Sun Cluster System Administration Guide for Solaris OS* for additional information about how to backup and restore data.

- **When is a node healthy enough to be used as a secondary node?**

  After a reboot, a node is healthy enough to be a secondary node when the node displays the login prompt.

Cluster Storage FAQs

- **What makes multihost storage highly available?**

  Multihost storage is highly available because it can survive the loss of a single disk, due to mirroring (or due to hardware-based RAID-5 controllers). Because a multihost storage device has more than one host connection, it can also withstand the loss of a single node to which it is connected. In addition, redundant paths from each node to the attached storage provide tolerance for the failure of a host bus adapter, cable, or disk controller.

Cluster Interconnect FAQs

- **What cluster interconnects does the SunPlex system support?**

  Currently, the SunPlex system supports Ethernet (100BASE-T Fast Ethernet and 1000BASE-SX Gb) cluster interconnects in both SPARC based and x86 based clusters. The SunPlex system supports the SCI network interface cluster interconnect in SPARC based clusters only.

- **What is the difference between a "cable" and a transport "path?"**

  Cluster transport cables are configured using transport adapters and switches. Cables join adapters and switches on a component-to-component basis. The cluster topology manager uses available cables to build end-to-end transport paths between nodes. A cable does not map directly to a transport path.

  Cables are statically "enabled" and "disabled" by an administrator. Cables have a "state," (enabled or disabled) but not a "status." If a cable is disabled, it is as if it were unconfigured. Cables that are disabled cannot be used as transport paths. They are not probed and therefore, it is not possible to know their status. The state of a cable can be viewed using `scconf -p`.

  Transport paths are dynamically established by the cluster topology manager. The "status" of a transport path is determined by the topology manager. A path can have a status of "online" or "offline." The status of a transport path can be viewed using `scstat(1M)`.

  Consider the following example of a two-node cluster with four cables.

  ```
  node1:adapter0     to switch1, port0
  node1:adapter1     to switch2, port0
  node2:adapter0     to switch1, port1
  node2:adapter1     to switch2, port1
  ```

  There are two possible transport paths that can be formed from these four cables.

  ```
  node1:adapter0    to node2:adapter0
  node2:adapter1    to node2:adapter1
  ```

Client Systems FAQs

- **Do I need to consider any special client needs or restrictions for use with a cluster?**

  Client systems connect to the cluster as they would any other server. In some instances, depending on the data service application, you might need to install client-side software or perform other configuration changes so that the client can connect to the data service application. See individual chapters in *Sun Cluster Data Services Planning and Administration Guide* for more information on client-side configuration requirements.

Administrative Console FAQs

- **Does the SunPlex system require an administrative console?**

  Yes.

- **Does the administrative console have to be dedicated to the cluster, or can it be used for other tasks?**

  The SunPlex system does not require a dedicated administrative console, but using one provides these benefits:

  - Enables centralized cluster management by grouping console and management tools on the same machine
  - Provides potentially quicker problem resolution by your hardware service provider

- **Does the administrative console need to be located "close" to the cluster itself, for example, in the same room?**

  Check with your hardware service provider. The provider might require that the console be located in close proximity to the cluster itself. No technical reason exists for the console to be located in the same room.

- **Can an administrative console serve more than one cluster, as long as any distance requirements are also first met?**

  Yes. You can control multiple clusters from a single administrative console. You can also share a single terminal concentrator between clusters.

Terminal Concentrator and System Service Processor FAQs

- **Does the SunPlex system require a terminal concentrator?**

  All software releases starting with Sun Cluster 3.0 do not require a terminal concentrator to run. Unlike the Sun Cluster 2.2 product, which required a terminal concentrator for failure fencing, later products do not depend on the terminal concentrator.

- **I see that most SunPlex servers use a terminal concentrator, but the Sun Enterprise E10000 server does not. Why is that?**

  The terminal concentrator is effectively a serial-to-Ethernet converter for most servers. Its console port is a serial port. The Sun Enterprise E10000 server doesn't have a serial console. The System Service Processor (SSP) is the console, either through an Ethernet or `jtag` port. For the Sun Enterprise E10000 server, you always use the SSP for consoles.

- **What are the benefits of using a terminal concentrator?**

  Using a terminal concentrator provides console-level access to each node from a remote workstation anywhere on the network, including when the node is at the OpenBoot PROM (OBP) on a SPARC based node or a boot subsystem on an x86 based node.

- **If I use a terminal concentrator not supported by Sun, what do I need to know to qualify the one that I want to use?**

  The main difference between the terminal concentrator supported by Sun and other console devices is that the Sun terminal concentrator has special firmware that prevents the terminal concentrator from sending a break to the console when it boots. Note that if you have a console device that can send a break, or a signal that might be interpreted as a break to the console, it shuts down the node.

- **Can I free a locked port on the terminal concentrator supported by Sun without rebooting it?**

  Yes. Note the port number that needs to be reset and type the following commands:

  ```
  telnet tc
  Enter Annex port name or number: cli
  annex: su -
  annex# admin
  admin : reset port_number
  admin : quit
  annex# hangup
  #
  ```

  Refer to the following manuals for more information about how to configure and administer the terminal concentrator supported by Sun.

- "Administering Sun Cluster Overview" in *Sun Cluster System Administration Guide for Solaris OS*
- "Installing and Configuring the Terminal Concentrator" in *Sun Cluster 3.x Hardware Administration Manual for Solaris OS*

- **What if the terminal concentrator itself fails? Must I have another one standing by?**

  No. You do not lose any cluster availability if the terminal concentrator fails. You do lose the ability to connect to the node consoles until the concentrator is back in service.

- **If I do use a terminal concentrator, what about security?**

  Generally, the terminal concentrator is attached to a small network used by system administrators, not a network that is used for other client access. You can control security by limiting access to that particular network.

- **SPARC: How do I use dynamic reconfiguration with a tape or disk drive?**

  - Determine whether the disk or tape drive is part of an active device group. If the drive is not part of an active device group, you can perform the DR remove operation on it.
  - If the DR remove-board operation would affect an active disk or tape drive, the system rejects the operation and identifies the drives that would be affected by the operation. If the drive is part of an active device group, go to "SPARC: DR Clustering Considerations for Disk and Tape Drives" on page 84.
  - Determine whether the drive is a component of the primary node or the secondary node. If the drive is a component of the secondary node, you can perform the DR remove operation on it.
  - If the drive is a component of the primary node, you must switch the primary and secondary nodes before performing the DR remove operation on the device.

Caution – If the current primary node fails while you are performing the DR operation on a secondary node, cluster availability is impacted. The primary node has no place to fail over until a new secondary node is provided.

Index

D

Q

R

S

T

U

V