Sun™ Cluster 3.1 Administration
ES-338

Student Guide

Sun Microsystems, Inc.
UBRM05-104
500 Eldorado Blvd.
Broomfield, CO 80021
U.S.A.

Revision B.1

Export Laws. Products, Services, and technical data delivered by Sun may be subject to U.S. export controls or the trade laws of other countries. You will comply with all such laws and obtain all licenses to export, re-export, or import as may be required after delivery to You. You will not export or re-export to entities on the most current U.S. export exclusions lists or to any country subject to U.S. embargo or terrorist controls as specified in the U.S. export laws. You will not use or provide Products, Services, or technical data for nuclear, missile, or chemical biological weaponry end uses.

DOCUMENTATION IS PROVIDED “AS IS” AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS, AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

THIS MANUAL IS DESIGNED TO SUPPORT AN INSTRUCTOR-LED TRAINING (ILT) COURSE AND IS INTENDED TO BE USED FOR REFERENCE PURPOSES IN CONJUNCTION WITH THE ILT COURSE. THE MANUAL IS NOT A STANDALONE TRAINING TOOL. USE OF THE MANUAL FOR SELF-STUDY WITHOUT CLASS ATTENDANCE IS NOT RECOMMENDED.

Export Commodity Classification Number (ECCN) assigned: 9 August 2005

Please
Recycle

Adobe PostScript

Législation en matière dexportations. Les Produits, Services et données techniques livrés par Sun peuvent être soumis aux contrôles américains sur les exportations, ou à la législation commerciale dautres pays. Nous nous conformerons à lensemble de ces textes et nous obtiendrons toutes licences dexportation, de ré-exportation ou dimportation susceptibles dêtre requises après livraison à Vous. Vous nexporterez, ni ne ré-exporterez en aucun cas à des entités figurant sur les listes américaines dinterdiction dexportation les plus courantes, ni vers un quelconque pays soumis à embargo par les Etats-Unis, ou à des contrôles anti-terroristes, comme prévu par la législation américaine en matière dexportations. Vous nutiliserez, ni ne fournirez les Produits, Services ou données techniques pour aucune utilisation finale liée aux armes nucléaires, chimiques ou biologiques ou aux missiles.

LA DOCUMENTATION EST FOURNIE "EN L'ETAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFAÇON.

CE MANUEL DE RÉFÉRENCE DOIT ÊTRE UTILISÉ DANS LE CADRE D'UN COURS DE FORMATION DIRIGÉ PAR UN INSTRUCTEUR (ILT). IL NE S'AGIT PAS D'UN OUTIL DE FORMATION INDÉPENDANT. NOUS VOUS DÉCONSEILLONS DE L'UTILISER DANS LE CADRE D'UNE AUTO-FORMATION.

Table of Contents

About This Course

Course Goals

Upon completion of this course, you should be able to:

- Describe the major Sun™ Cluster software components and functions
- Configure different methods of connecting to node consoles
- Configure a cluster administrative workstation
- Install and configure the Sun Cluster 3.1 software
- Configure Sun Cluster 3.1 software quorum devices
- Configure VERITAS Volume Manager in the Sun Cluster software environment
- Configure Solaris™ Volume Manager software in the Sun Cluster software environment
- Create Internet Protocol Multipathing (IPMP) failover groups in the Sun Cluster 3.1 software environment
- Understand resources and resource groups
- Configure a failover data service resource group (Network File System [NFS])
- Configure a scalable data service resource group (Apache)
- Configure failover Solaris 10 zones, failover ORACLE®, and ORACLE Real Application Clusters (RAC) in the Sun Cluster software environment

Course Map

The following course map enables you to see what you have accomplished and where you are going in reference to the course goals.

Product Introduction

Introducing Sun™ Cluster Hardware and Software

Installation

Exploring Node Console Connectivity and the Cluster Console Software

Preparing for Installation and Understanding Quorum Devices

Installing and Configuring the Sun Cluster Software Framework

Operation

Performing Basic Cluster Administration

Customization

Using VERITAS Volume Manager for Volume Management

Managing Volumes With Solaris™ Volume Manager (Solstice DiskSuite™ Software)

Managing the Public Network With IPMP

Introducing Data Services, Resource Groups, and HA-NFS

Configuring Scalable Services and Advanced Resource Group Relationships

Supplemental Exercises

Performing Supplemental Exercises for Sun Cluster 3.1 Software

Topics Not Covered

This course does not cover the topics shown on the overhead. Many of the topics listed on the overhead are described in other courses offered by Sun Services:

- Database installation and management – Described in database vendor courses
- Solaris™ 10 Operating System (OS) Advanced Features or Differences
- Network administration
- Solaris™ Operating System (Solaris OS) administration – Described in SA-239: *Intermediate System Administration for the Solaris™ 9 Operating System* and SA-299: *Advanced System Administration for the Solaris™ 9 Operating System,* and the Solaris 10 OS equivalents
- Disk storage management – Described in ES-222: *Solaris™ Volume Manager Administration* and ES-310: *Sun StorEdge™ Volume Manager Administration*

Refer to the Sun Services catalog for specific information and registration.

How Prepared Are You?

To be sure you are prepared to take this course, can you answer yes to the following questions?

- Can you explain virtual volume management terminology, such as mirroring, striping, concatenation, volumes, and mirror synchronization?
- Can you perform basic Solaris OS administration tasks, such as using `tar` and `ufsdump` commands, creating user accounts, formatting disk drives, using `vi`, installing the Solaris OS, installing patches, and adding packages?
- Do you have prior experience with Sun hardware and the OpenBoot™ programmable read-only memory (PROM) technology?
- Are you familiar with general computer hardware, electrostatic precautions, and safe handling practices?

Introductions

Now that you have been introduced to the course, introduce yourself to each other and the instructor, addressing the items shown in the following bullets.

- Name
- Company affiliation
- Title, function, and job responsibility
- Experience related to topics presented in this course
- Reasons for enrolling in this course
- Expectations for this course

How to Use Course Materials

To enable you to succeed in this course, these course materials use a learning model that is composed of the following components:

- Goals – You should be able to accomplish the goals after finishing this course and meeting all of its objectives.
- Objectives – You should be able to accomplish the objectives after completing a portion of instructional content. Objectives support goals and can support other higher-level objectives.
- Lecture – The instructor will present information specific to the objective of the module. This information should help you learn the knowledge and skills necessary to succeed with the activities.
- Activities – The activities take on various forms, such as an exercise, self-check, discussion, and demonstration. Activities help to facilitate mastery of an objective.
- Visual aids – The instructor might use several visual aids to convey a concept, such as a process, in a visual form. Visual aids commonly contain graphics, animation, and video.

Conventions

The following conventions are used in this course to represent various training elements and alternative learning resources.

Icons

Additional resources – Indicates other references that provide additional information on the topics described in the module.

Discussion – Indicates a small-group or class discussion on the current topic is recommended at this time.

Note – Indicates additional information that can help students but is not crucial to their understanding of the concept being described. Students should be able to understand the concept or complete the task without this information. Examples of notational information include keyword shortcuts and minor system adjustments.

Caution – Indicates that there is a risk of personal injury from a nonelectrical hazard, or risk of irreversible damage to data, software, or the operating system. A caution indicates that the possibility of a hazard (as opposed to certainty) might happen, depending on the action of the user.

Typographical Conventions

`Courier` is used for the names of commands, files, directories, programming code, and on-screen computer output; for example:

> Use `ls -al` to list all files.
> `system% You have mail.`

`Courier` is also used to indicate programming constructs, such as class names, methods, and keywords; for example:

> Use the `getServletInfo` method to get author information.
> The `java.awt.Dialog` class contains `Dialog` constructor.

`Courier bold` is used for characters and numbers that you type; for example:

> To list the files in this directory, type:
> `#` **`ls`**

`Courier bold` is also used for each line of programming code that is referenced in a textual description; for example:

```
1 import java.io.*;
2 import javax.servlet.*;
3 import javax.servlet.http.*;
```

> Notice the `javax.servlet` interface is imported to allow access to its life cycle methods (Line 2).

`Courier italic` is used for variables and command-line placeholders that are replaced with a real name or value; for example:

> To delete a file, use the `rm` *`filename`* command.

`Courier italic bold` is used to represent variables whose values are to be entered by the student as part of an activity; for example:

> Type **`chmod a+rwx`** ***`filename`*** to grant read, write, and execute rights for filename to world, group, and users.

Palatino italic is used for book titles, new words or terms, or words that you want to emphasize; for example:

> Read Chapter 6 in the *User's Guide*.
> These are called *class* options.

Module 1

Introducing Sun™ Cluster Hardware and Software

Objectives

Upon completion of this module, you should be able to:

- Define the concept of clustering
- Describe the Sun™ Cluster 3.1 hardware and software environment
- Explain the Sun Cluster 3.1 hardware environment
- View the Sun Cluster 3.1 software support
- Describe the types of applications in the Sun Cluster 3.1 software environment
- Identify the Sun Cluster 3.1 software data service support
- Explore the Sun Cluster 3.1 software high availability (HA) framework
- Define global storage services differences

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- Are there similarities between the redundant hardware components in a standalone server and the redundant hardware components in a cluster?
- Why is it impossible to achieve industry standards of high availability on a standalone server?
- Why are a variety of applications that run in the Sun Cluster software environment said to be *cluster-unaware*?
- How do cluster-unaware applications differ from cluster-aware applications?
- What services does the Sun Cluster software framework provide for all the applications running in the cluster?
- Do global devices and global file systems have usage other than for scalable applications in the cluster?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)

Defining Clustering

Clustering is a general terminology that describes a group of two or more separate servers operating as a "harmonious unit." Clusters generally have the following characteristics:

- Separate server nodes, each booting from its own, *non-shared*, copy of the operating system
- Dedicated hardware interconnect, providing private transport only between the nodes of the same cluster
- Multiported storage, providing paths from at least two nodes in the cluster to each physical storage device storing data for the applications running in the cluster
- Cluster software "framework," providing cluster-specific knowledge to the nodes in the cluster about the health of the hardware and the health and state of their peer nodes
- General goal of providing a platform for high availability and scalability for the applications running in the cluster
- Support for a variety of *cluster-unaware* applications and *cluster-aware* applications

High-Availability Platforms

Clusters are generally marketed as the only way to provide *high availability* (HA) for the applications that run on them.

HA can be defined as a minimization of downtime rather than the complete elimination of downtime. Many standalone servers themselves are marketed as "providing higher levels of availability than our competitors (or predecessors)." Most true standards of HA cannot be achieved in a standalone server environment.

HA Standards

HA standards are usually phrased with wording such as "provides 5 nines availability." This means 99.999 percent uptime for the application or about five minutes of downtime per year. One clean server reboot often already exceeds that amount of downtime.

How Clusters Provide HA (Inter-Node Failover)

Cluster environments provide an environment where, in the case of any single hardware or software failure in the cluster, application services and data are recovered automatically (without human intervention) and quickly (faster than a server reboot). This is done by taking advantage of the existence of the redundant servers in the cluster and redundant server-storage paths.

HA Benefits for Unplanned and Planned Outages

The HA benefit that cluster environments provide involves not only hardware and software failures, but also “planned outages.” Where a cluster has the ability to automatically relocate applications inside the cluster in the case of failures, it also provides the ability to manually relocate services for planned outages. As such, normal reboots or hardware maintenance in the cluster only affects the uptime of the applications for as much time as it takes to manually relocate them to different servers in the cluster.

Fault-Tolerant Servers Are *Not* an Alternative to HA Clusters

Many vendors provide servers that are marketed as *fault-tolerant*. These servers are designed to be able to tolerate any single hardware failure (including memory failure, central processing unit (CPU) failure, and the like) without any downtime whatsoever.

There is a common misconception that fault-tolerant servers are an alternative to HA Clusters, or that a fault-tolerant server supersedes HA in some way. In fact, while fault-tolerant servers can hide any hardware failure, they are not designed to provide especially fast recovery in the case of a software failure, such as a Solaris OS kernel panic or an application failure. Recovery in these circumstances on a single fault-tolerant server might still require a full OS reboot, which, as previously stated, might already exceed maximum downtime permitted by the HA standards to which you aspire.

Scalability Platforms

Clusters also provide an integrated hardware and software environment for *scalability*. Scalability is defined as the ability to increase application performance by supporting multiple instances of applications on different nodes in the cluster. These instances are generally accessing the *same* data as each other.

High Availability and Scalability Are *Not* Mutually Exclusive

Clusters generally do not require a choice between availability and performance. HA is generally built-in to scalable applications as well as non-scalable ones. In scalable applications, you might not need to relocate failed applications because other instances are already running on other nodes. You might still need to perform recovery on behalf of failed instances.

Sun Cluster 3.1 Hardware and Software Environment

The Sun Cluster 3.1 hardware and software environment is Sun's latest generation clustering product. The following features distinguish the Sun Cluster hardware and software product from competitors in the field:

- *Supports from two to sixteen nodes* – Nodes can be added, removed, and replaced from the cluster without any interruption of application service.
- *Global device implementation* – While data storage must be physically connected on paths from at least two different nodes in the Sun Cluster 3.1 hardware and software environment, all the storage in the cluster is logically available from every node in the cluster using standard device semantics.

  This provides a huge amount of flexibility in being able to run applications on nodes that use data that is not even physically connected to the nodes. More information about global device file naming and location can be found in "Global Storage Services" on page 1-34.
- *Global file system implementation* – The Sun Cluster software framework provides a global file service independent of any particular application running in the cluster, so that the same files can be accessed on every node of the cluster, regardless of the storage topology.
- *Cluster framework services implemented in the kernel* – The Sun Cluster software is tightly integrated with the Solaris 8 OS, Solaris 9 OS, and Solaris 10 OS kernels. Node monitoring capability, transport monitoring capability, and the global device and file system implementation among others, are all implemented in the kernel to provide higher reliability and performance.

- *Support for a large variety of "off-the-shelf" applications* – The Sun Cluster hardware and software product includes data service agents for a large variety of cluster-unaware applications. These are tested scripts and fault monitors that make applications run properly in the cluster environment with the inherent high availability and scalability that entails. A full list of data service agents is in "Sun Cluster 3.1 Software Data Service Support" on page 1-29.
- *Support for some "off-the-shelf "applications as scalable applications with built-in load balancing (global interfaces)* – The global interface feature provides a single Internet Protocol (IP) address and load-balancing service for some scalable services, for example, Apache Web Server and Sun Java™ System Web Server (formerly known as Sun ONE Web Server). Clients outside the cluster see the multiple node instances of the service as a single service with a single IP address.

Sun Cluster 3.1 Hardware Environment

The Sun Cluster 3.1 hardware environment supports a maximum of sixteen nodes. Figure 1-1 demonstrates the hardware components of a typical two-node cluster:

- Cluster nodes running Solaris 8 OS (Update 7 or higher), Solaris 9 OS, or Solaris 10 OS

  Each node must run the same revision and same update of the OS.
- Separate boot disks on each node (with a preference for boot disks mirrored on each node)
- One or more public network interfaces per system per subnet (with a preferred minimum of at least two public network interfaces)
- A redundant private cluster transport interface
- Dual-hosted, mirrored disk storage
- One terminal concentrator (or any other console access method)
- Administrative console

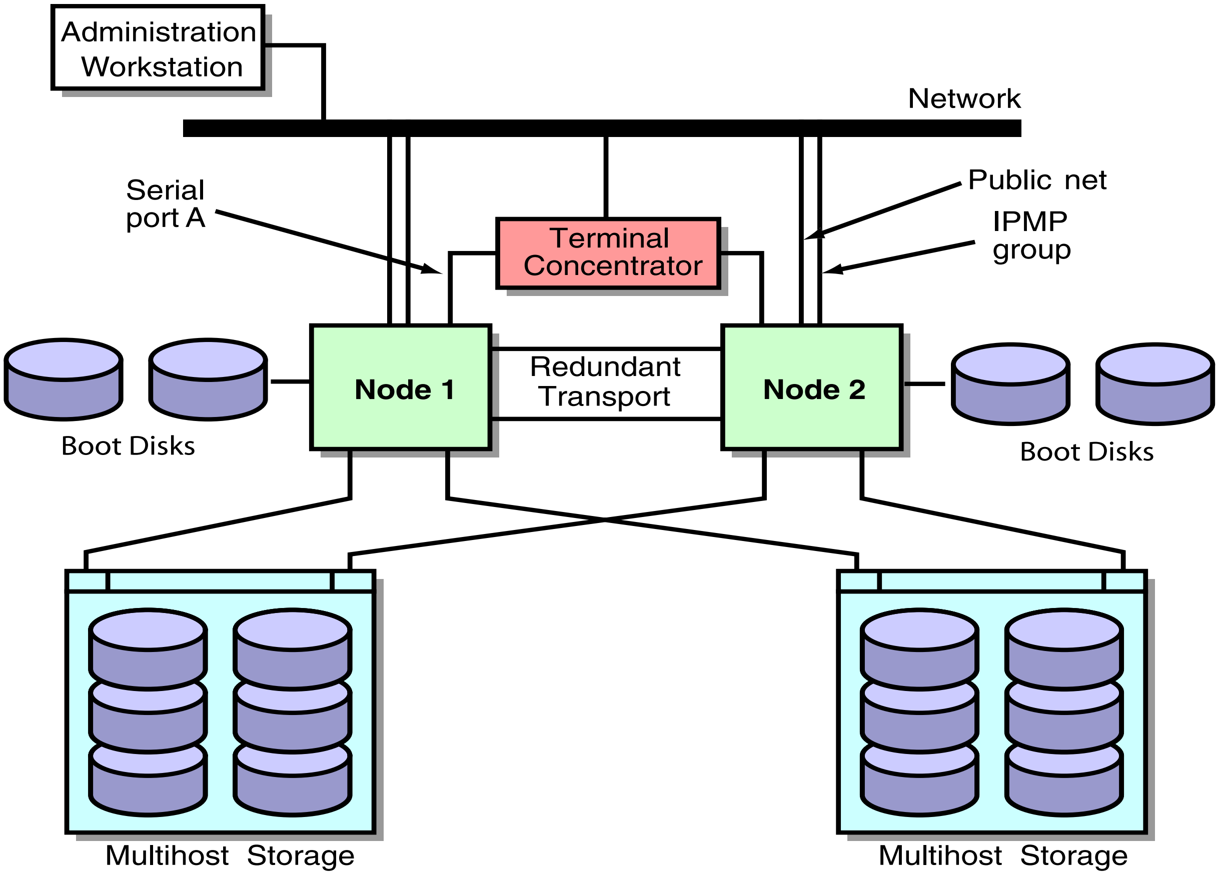


Figure 1-1 Minimal Sun Cluster 3.1 Hardware Environment

Cluster Host Systems

A wide range of Sun hardware platforms are supported for use in the clustered environment. These range from small rackmounted servers (SPARC and x86 servers) up to Sun's largest enterprise-level servers, including Sun Fire™ E25K and 15K servers.

Many heterogeneous environments are supported (that is, different types of servers used as nodes in the same cluster). It depends on the network and storage host adapters used, not on the servers themselves.

Cluster Transport Interface

All nodes in a cluster are linked by a private cluster transport. The transport is redundant and can be used for the following purposes:

- Cluster-wide monitoring and recovery
- Global data access (transparent to applications)
- Application-specific transport for cluster-aware applications (such as ORACLE® Real Application Clusters (RAC))

Clusters must be defined with a minimum of two separate "private networks" that form the cluster transport. You can have more than two (and you can add more later) that would be a performance benefit in certain circumstances, because global data access traffic is "striped" across all of the transports.

Crossover cables are often used in a two-node cluster. Switches are optional when you have two nodes and are required for more than two nodes.

Following are the types of cluster transport hardware supported:

- Fast Ethernet (100BASE-T). This is the most common type of transport.

- Gigabit Ethernet (1000BASE-SX). This is used when a higher cluster transport speed is required for proper functioning of a data service.
- Scalable coherent interface (SCI) intended for remote shared memory (RSM) applications. These applications are described in "Other Remote Shared Memory (RSM) Applications" on page 1-28. The SCI interconnect is supported only in clusters with a maximum of four nodes.
- Sun Fire Link hardware intended for RSM applications and available in the Sun Fire Midrange servers and Sun Fire High-End servers. It is supported only in clusters with a maximum of four nodes.
- Infiniband (Solaris 10 OS only)

  This is a relatively new industry standard interconnect used outside of the Sun Cluster environment for interconnecting a variety of hosts and storage devices. In the Sun Cluster environment it is supported only as an interconnect between hosts, not between hosts and storage devices.

Public Network Interfaces

Each node must have public network interfaces under control of the Solaris OS IP Multipathing (IPMP) software. It is *recommended*, though not required, that at least two interfaces in an IPMP group connect each node to each subnet used. The cluster requires that all nodes on which an application might run be on the same subnet.

You can have connectivity to as many subnets as you want, but a Sun Cluster hardware host is not allowed to act as a router. Not all available network interface cards are supported in the Sun Cluster 3.1 hardware environment.

Cluster Disk Storage

The Sun Cluster hardware environment can use several Sun storage models. They must all accept multihost connections. The Sun StorEdge™ T3 array has a single connection and must be used with a hub or a switch.

Some data storage arrays support only two physically-connected nodes. Many other storage configurations support more than two nodes connected to the storage.

VERITAS Volume Manager or Solaris Volume Manager (Solstice DiskSuite™) software is normally used to mirror the storage across controllers. You can choose not to use any Volume Manager if each node has multipathed access to highly-available hardware redundant array of independent disks (RAID) storage.

Boot Disks

The Sun Cluster 3.1 environment *requires* that boot disks for each node be local to the node. That is, not in any of the multiported storage arrays.

Having two local drives for disks and mirroring with VERITAS Volume Manager or Solaris Volume Manager software is *recommended.*

Terminal Concentrator

The server nodes in the cluster typically run without a graphics console, although the Sun graphics console (keyboard and monitor) is allowed when supported by the base server configuration. If the server nodes do not use a Sun graphics console, you can use any method you want to connect to the serial (`ttyA`) consoles of the nodes.

A terminal concentrator (TC) is a device that provides data translation from the network to serial port interfaces. Each of the serial port outputs connects to a separate node in the cluster through serial port A.

Another way of getting the convenience of *remote console access* is to just use a workstation that has serial ports connected to serial port A of each node (which is perfect for workstations with two serial ports in a two-node cluster). You can access this workstation remotely and then use the `tip` command to access the node consoles.

There is always a trade-off between convenience and security. You might prefer to have only "dumb-terminal" console access to the cluster nodes, and keep these terminals behind locked doors requiring stringent security checks to open them. This is acceptable (although less convenient to administer) for Sun Cluster 3.1 hardware as well.

Administrative Workstation

Included with the Sun Cluster software is the administration console software, which can be installed on any SPARC or x86 Solaris OS workstation. The software can be a convenience in managing the multiple nodes of the cluster from a centralized location. It does not affect the cluster in any other way.

Clusters With More Than Two Nodes

In clusters with more than two nodes:

- Switches are required for the transport. There must be at least two switches. Each node is connected to each switch to form a redundant transport.
- A variety of storage topologies are supported. Figure 1-2 on page 1-14 shows an example of a "Pair + N" topology. All the nodes in the cluster have access to the data in the multihost storage through the global devices and global file system features of the Sun Cluster environment.

See Module 3, “Preparing for Installation and Understanding Quorum Devices” for more information on these subjects.

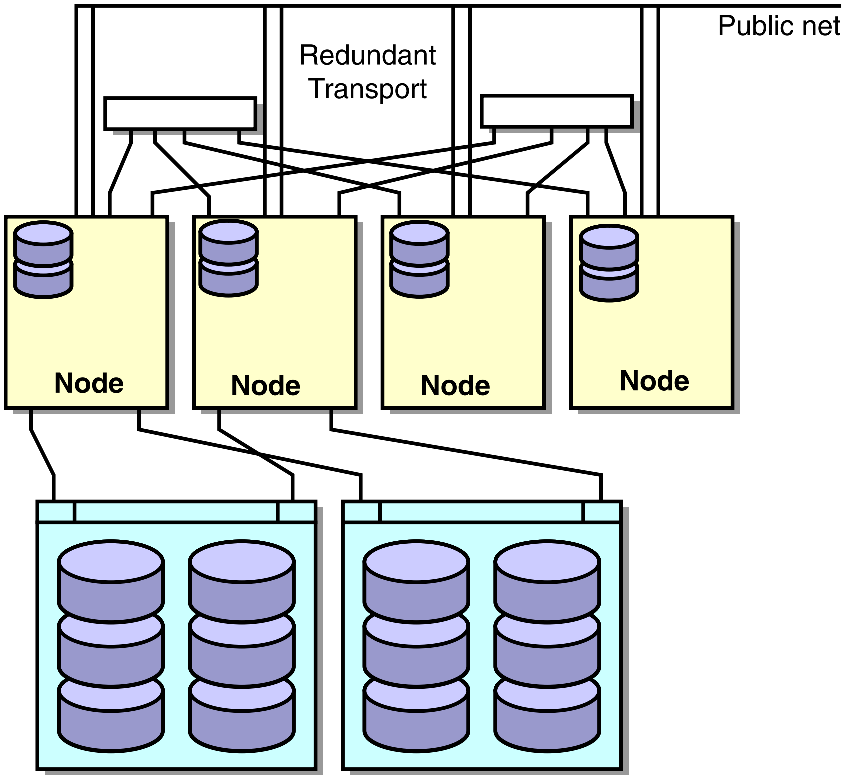


Figure 1-2 “Pair + N” Topology

Cluster With Network Attached Storage (NAS) Devices

Sun Cluster 3.1 8/05 (Update 4) provides a framework for supporting network-attached storage (NAS) devices as the data storage available for the Sun Cluster nodes without requiring any other shared storage devices (Figure 1-3).

A NAS device provides file services to all of the cluster nodes through the Network File System (NFS) or any other network file-sharing protocol. File services can run between the cluster nodes and the network storage server on a dedicated subnet or—less likely— on the same public subnet providing access to the clients of the cluster services. In other words, file traffic is supported on any network *except* those that make up the cluster interconnect.

Sun Cluster 3.1 Update 4 software provides a general *drop-in* type of architecture to serve as an interface for managing NAS in the cluster.

In the current release, the only specific implementation supported is the Network Appliance (NetApp) Filer product, which uses NFS to provide file services to the clients which, in this case, are the cluster nodes. All Sun Cluster applications are supported with their data on the NAS device *except* NFS itself.

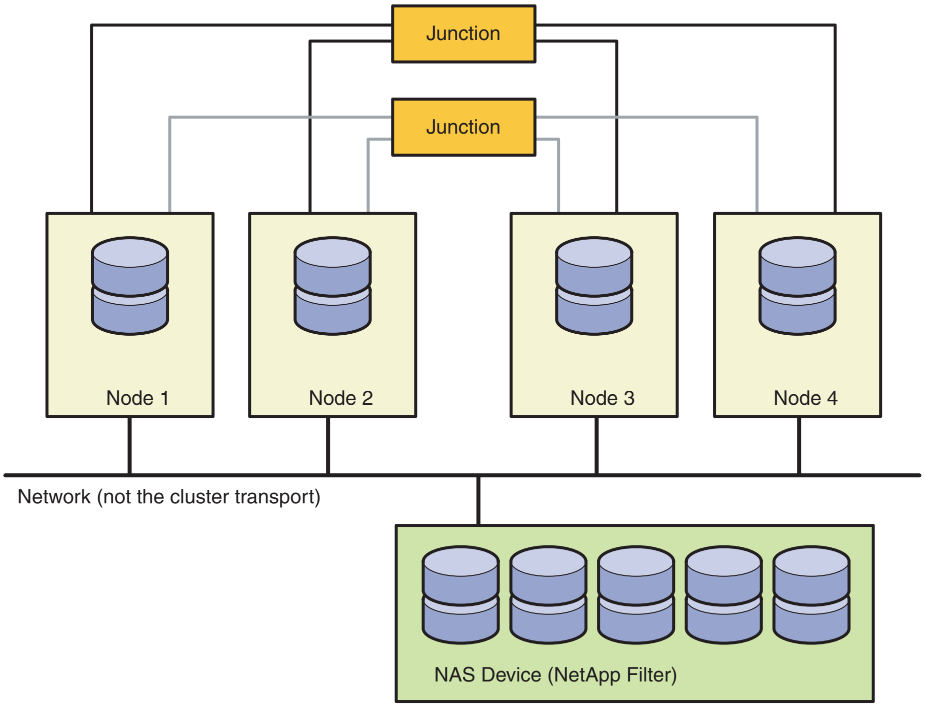

Junction
Junction
Node 1
Node 2
Node 3
Node 4
Network (not the cluster transport)
NAS Device (NetApp Filter)

Figure 1-3 Cluster With NAS Device as Only Shared Storage

Cluster With Shared Physical Interconnects

Starting in Sun Cluster 3.1 8/05 (Update 4) certain types of transport adapters can be used as public network adapters and private network transport adapters simultaneously. The purpose of this feature is to allow certain types of servers that may never have more than two physical network adapters—such as servers in a blade architecture—to be used as Sun Cluster nodes.

Using this feature, such servers could use each physical adapter both as a single transport adapter and a single public network adapter.

Tagged Virtual Local Area Networks (VLANs) and Adapter Restrictions

This feature makes use of network device drivers that support a specification called *tagged VLANs*. In the tagged VLAN specification the virtual network identity of an adapter is known by the adapter itself in the form of a VLAN identifier (VLAN-ID). This VLAN-ID is encapsulated as part of the header information at the media access control (MAC) level and interpreted by the switches. Adapters configured with a particular value for the VLAN-ID only accept packets containing the same VLAN-ID.

The only adapters that support the tagged VLAN device driver and that are also supported in the Sun Cluster environment are the Cassini Ethernet (ce) adapters and the Broadcom Gigabit Ethernet (bge) adapters. Thus the shared physical interconnects feature is only available with those adapters.

Illustration of Shared Private and Public Networks

Figure 1-4 is a diagram of a two-node cluster where each node has only two physical network adapters that are capable of tagged VLANs:

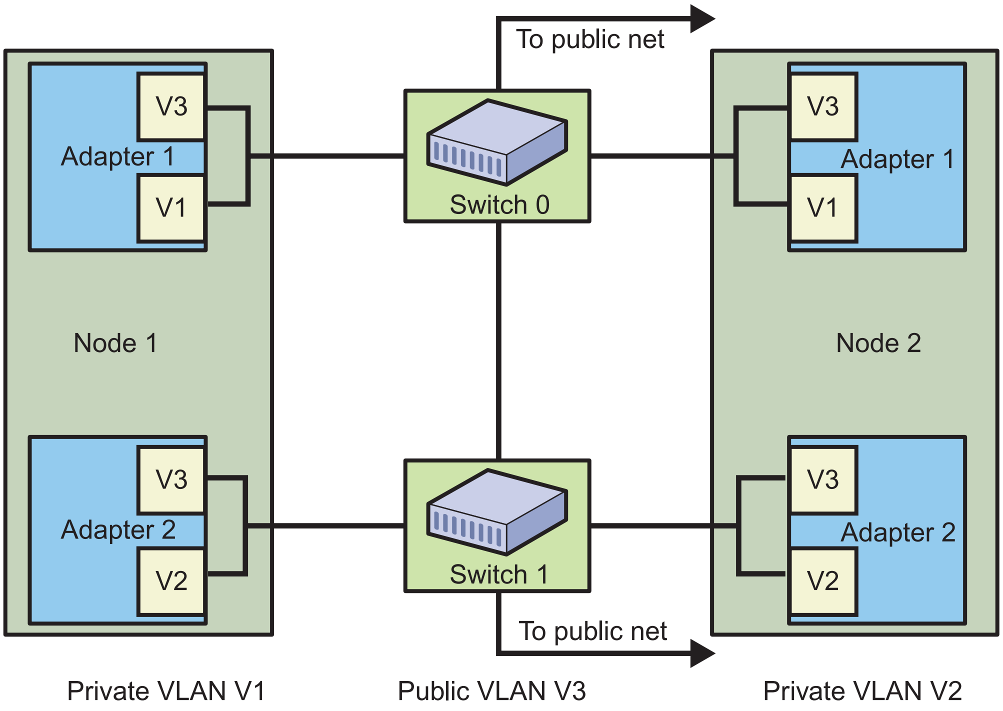


Figure 1-4 Adapters Being Shared Between Public and Private Nets

As shown in Figure 1-4, the switches are interconnected in order to support the two adapters on each node existing in the same IPMP group for public network address failover. In addition, each switch is being used as a private network interconnect. The isolation of traffic between the private network and public network adapters is controlled by the VLAN-ID that is assigned to the adapters themselves, which pass the information to the network fabric.

Sun Cluster Hardware Redundancy Features

The following items summarize the generally required and optional hardware redundancy features in the Sun Cluster hardware environment:

- Redundant server nodes are *required.*
- Redundant transport is *required.*
- High availability access to and from *each* node to data storage is *required*. That is, at least one of the following (you could have both):
  - Mirroring across controllers for "Just a Bunch Of Disks" (JBOD) or for hardware RAID devices without multipathing (such as single-brick T3's, or 3310 RAID boxes each with only a single RAID controller)
  - Multipathing from each connected node to highly-available hardware RAID devices
- Redundant public network interfaces per subnet are *recommended.*
- Redundant boot disks are *recommended.*

It is always recommended, although not required, to have redundant components as far apart as possible. For example, on a system with multiple input/output (I/O) boards, you want to put the redundant transport interfaces, the redundant public nets, and the redundant storage array controllers on two different I/O boards.

Cluster in a Box

The Sun Cluster 3.1 hardware environment supports clusters with both (or all) nodes in the same physical box for the following domain-based servers (current at the time of writing of this course):

- Sun Fire E25K/E20K/15K/12K servers (High-end Servers)
- Sun Fire 3800–6800 and E4900/E6900 servers (Midframe Servers)
- Sun Enterprise™ 10000 servers

You should take as many redundancy precautions as possible, for example, running multiple domains in different segments on Sun Fire 3x00–6x00 servers and segmenting across the power plane on Sun Fire 6x00 servers.

If a customer chooses to configure the Sun Cluster 3.1 hardware environment in this way, there must be a realization that there is a greater chance of a single hardware failure causing downtime for the entire cluster than there is when clustering independent standalone servers, or domains from domain-based servers across different physical platforms.

Sun Cluster 3.1 Software Support

To function as a cluster member, the following types of software must be installed on every Sun Cluster hardware node:

- Solaris OS software
- Sun Cluster software
- Data service application software
- Logical volume management

An exception is a configuration that uses hardware RAID. This configuration may not require a software volume manager.

Figure 1-5 provides a high-level overview of the software components that work together to create the Sun Cluster software environment.

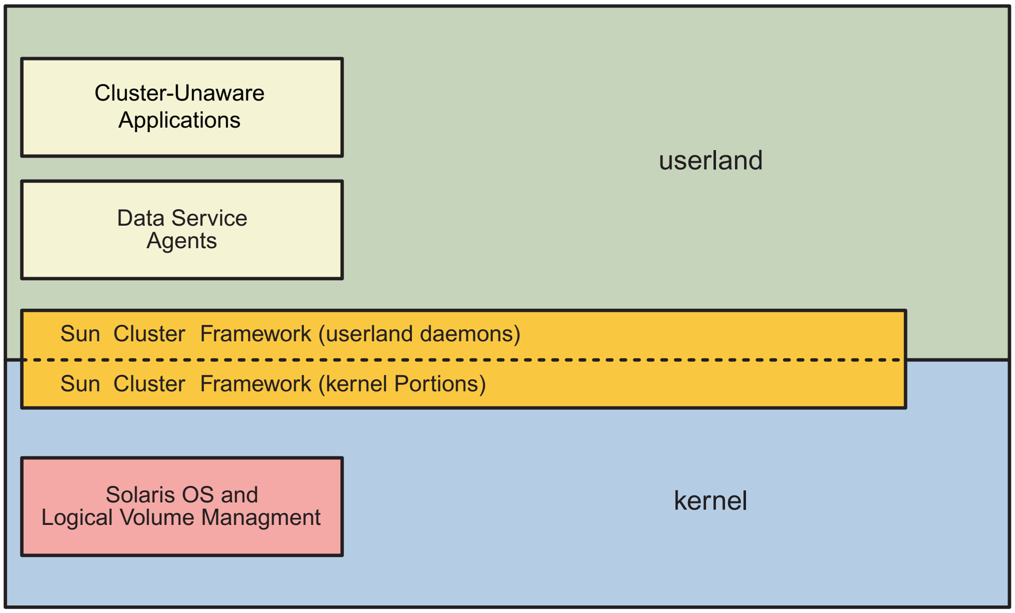

Cluster-Unaware Applications
userland
Data Service Agents
Sun Cluster Framework (userland daemons)
Sun Cluster Framework (kernel Portions)
Solaris OS and Logical Volume Managment
kernel

Figure 1-5 Sun Cluster Software Layered Model

Software Revisions

The following software revisions are supported by Sun Cluster 3.1 software:

- Solaris OS versions (each node must run the same version)
  - Solaris 8 OS Update 7 (02/02) and later
  - Solaris 9 OS (all updates)
  - Solaris 10 OS
- VERITAS Volume Manager versions
  - VERITAS Volume Manager 3.5 (for the Solaris 8 or Solaris 9 OS)
  - VERITAS Volume Manager 4.0 (for the Solaris 8 or Solaris 9 OS)
  - VERITAS Volume Manager 4.1 (for Solaris 8, Solaris 9, and Solaris 10 OS)
- Solstice DiskSuite software revisions
  - Solstice DiskSuite 4.2.1 software for the Solaris 8 OS
  - Soft partitions supported with patch 108693-06 or greater
- Solaris Volume Manager software revisions – Solaris Volume Manager software (part of base OS in the Solaris 9 OS and Solaris 10 OS)

Note – New Solaris 10 OS updates may have some lag before being officially supported in the cluster. If there is any question, consult Sun support personnel who have access to the latest support matrices.

Solaris 10 OS Differences

While Solaris 10 OS has many advanced features, the Sun Cluster 3.1 8/05 (update 4) release, which is the first to support Solaris 10 OS, has as a goal Solaris 10 "co-existence." There are very few differences when administering the Sun Cluster software on Solaris 10 OS as opposed to Solaris 8 OS or Solaris 9 OS.

Solaris 10 OS Service Management Facility (SMF)

While the Sun Cluster software makes use of traditional boot scripts to launch Sun cluster framework dameons in Solaris 8 OS and Solaris 9 OS, the software makes use of the new Service Management Facility (SMF) in Solaris 10 OS. The daemons in question are detailed later in the course, in Module 5.

The fact that cluster daemons are defined as SMF services in Solaris 10 OS makes very little difference in how you will manage the cluster. In no case (Solaris 8 OS, Solaris 9 OS, or Solaris 10 OS) will you ever be launching or killing Sun Cluster framework daemons by hand.

There will be a slight difference in what you will experience at boot time in the Solaris 10 OS. In Solaris 8 OS and 9 OS, by the time you get the console login prompt on a node, you can be guaranteed that all the Sun cluster framework daemons are already running. In Solaris 10 OS, on the other hand, you will get the login prompt much more quickly, while some of the cluster framework daemons are still being launched, and others may still be waiting to be launched.

Solaris 10 OS Zones

The Solaris 10 OS zones feature is a powerful new facility that allows you to create multiple, virtual instances of the Solaris 10 OS, or local zones, each running inside a physical instance of the Solaris OS. Each configured *local zone* (the main Solaris 10 OS instance is known as the *global zone*), appears to be a complete OS instance isolated from each other and from the global zone. Each zone has its own *zonepath*, that is, its own completely separate directory structure for all of its files starting with the root directory. When configuring a local zone, its zonepath is expressed as a subdirectory somewhere inside a global zone mounted filesystem. When you log in to a local zone, however, you see the file systems belonging only to that local zone, users and processes belonging only to that local zone, and so on. Root users logged into one local zone have superuser privileges only in that zone.

Sun Cluster 3.1 Update 4 does *not* provide full zone integration. A full zone integration feature would allow you to essentially have clustered Solaris 10 OS zones, with cluster framework software installed in the zones, and cluster configuration and agents working together inside the zones to provide resource failover and scalability in zones. There could be clustered relationships between zones running on different physical machines or even between different local zones running on the same physical machine.

What Sun Cluster 3.1 Update 4 *does* provide is an agent that allows you to boot and control *cluster-unaware local zones*, that live inside physical machines (global zones) which are, of course, running the cluster framework.

The official name of the agent is *Sun Cluster Data Service for Solaris Containers*. A *Solaris Container* is just a zone that is managed with the Solaris Resource Manager (SRM). The agent actually does not provide any automated management of SRM.

You have a chance to exercise the co-existence of Solaris 10 zones in the cluster in one of the lab exercises of Module 11.

Types of Applications in the Sun Cluster Software Environment

The Sun Cluster software environment supports applications with High Availability (HA) and scalability.

Cluster-Unaware (Off-the-Shelf) Applications

The majority of applications in the cluster are in the cluster-unaware category and are part of the main focus of this course.

Two main categories of cluster-unaware applications can run in the Sun Cluster 3.1 environment. These are described on the following pages and in detail in Module 9, "Introducing Data Services, Resource Groups, and HA-NFS," and Module 10, "Configuring Scalable Services and Advanced Resource Group Relationships." The applications include the following:

- Failover applications
- Scalable applications

The common elements of all of these applications are the following:

- The cluster's *resource group manager* (RGM) coordinates all stopping and starting of the applications. They are never started and stopped by traditional Solaris OS run-control (rc) scripts.
- A *data service agent* for the application provides the "glue pieces" to make it work properly in the Sun Cluster software environment. This includes methods to start and stop the application appropriately in the cluster, as well as fault monitors specific for that application in the cluster.

Failover Applications

The failover model is the easiest to provide for in the cluster. Failover applications run on only one node of the cluster at a time. The cluster provides high-availability by providing automatic restart on the same or a different node of the cluster.

Failover services are usually paired with an *application IP address*. This is an IP address that always fails over from node to node along with the application. In this way, clients outside the cluster see a *logical host name* with no knowledge on which node a service is running or even knowledge that a service is running in the cluster.

Note – Beginning in Sun Cluster 3.1 9/04, both IPV4 addresses and IPV6 addresses are supported.

Multiple failover applications in the same r*esource group* can share an IP address, with the restriction that they must all fail over to the same node together as shown in Figure 1-6.

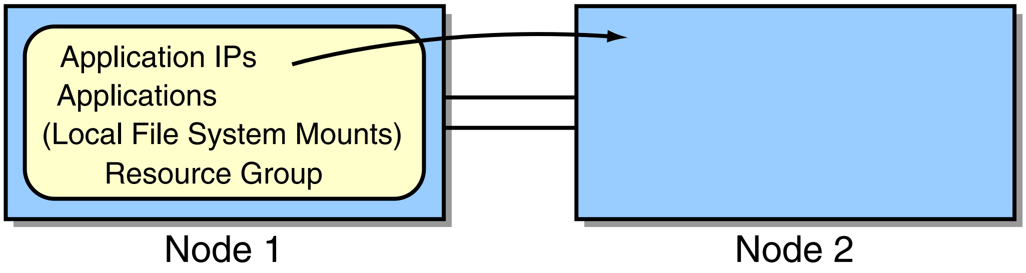


Figure 1-6 Multiple Failover Applications in the Same Resource Group

Scalable Applications

Scalable applications involve running multiple instances of an application in the same cluster and making it look like a single service by means of a *global interface* that provides a single IP address and load balancing.

While scalable applications (Figure 1-7) are still off-the-shelf, not every application can be made to run as a scalable application in the Sun Cluster 3.1 software environment. Applications that write data without any type of locking mechanism might work as failover applications but do not work as scalable applications.

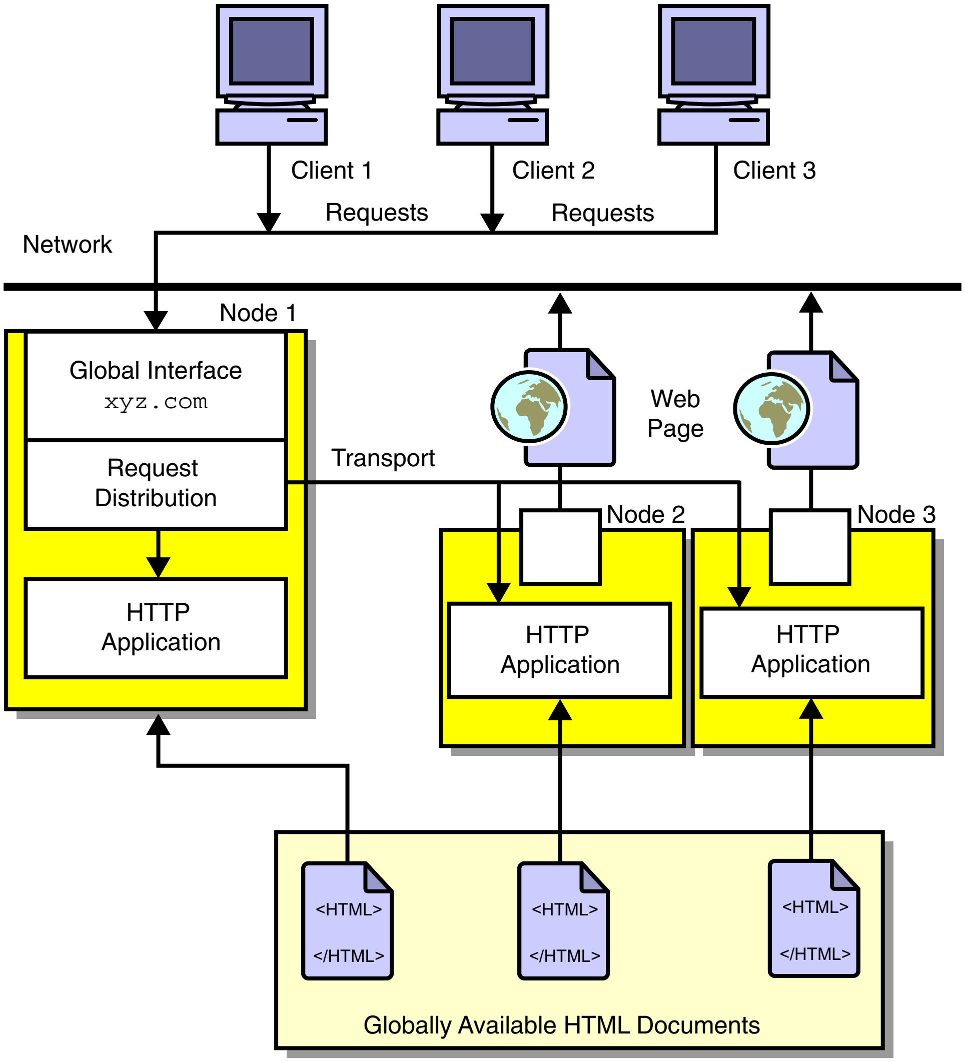


Figure 1-7 Scalable Application Work Flow

Cluster-Aware Applications

Cluster-aware applications are applications where knowledge of the cluster is *built-in* to the software. They differ from *cluster-unaware* applications in the following ways:

- Multiple instances of the application running on different nodes are aware of each other and communicate across the private transport.
- It is *not required* that the Sun Cluster software framework RGM start and stop these applications. Because these applications are cluster-aware, they can be started in their own independent scripts, or by hand.
- Applications are *not necessarily* logically grouped with external *application IP addresses*. If they are, the network connections can be monitored by cluster commands. It is also possible to monitor these cluster-aware applications with Sun Cluster 3.1 software framework resource types.

Parallel Database Applications

Parallel databases are a special type of cluster application. Multiple instances of the database server cooperate in the cluster, handling different queries on the same database and even providing parallel query capability on large queries. The following are supported in the Sun Cluster 3.1 software environment:

- ORACLE 8i Parallel Server (ORACLE 8i OPS)
- ORACLE 9i Real Application Clusters (ORACLE 9i RAC)
- ORACLE 10g Real Application Clusters (ORACLE 10g RAC)

Other Remote Shared Memory (RSM) Applications

Applications that run on Sun Cluster 3.1 hardware, and also contain the SCI or Sun Fire Link special interconnects, can make use of an application programming interface (API) called RSM. This maps data from an application instance running on one node into the address space of an instance running on another node. This can be a highly efficient way for cluster-aware applications to share large amounts of data across the transport.

ORACLE RAC is the only application that is supported in the Sun Cluster 3.1 software environment that can make use of RSM (if you have the right interconnect).

Note – This course focuses on the cluster-unaware applications rather than the types presented on this page. Module 11, "Performing Supplemental Exercises for Sun Cluster 3.1 Software," contains procedures and an optional exercise for running ORACLE 10g RAC (without RSM).

Sun Cluster 3.1 Software Data Service Support

This section contains a list of Sun-supported data service agents that make cluster-unaware applications highly available, either in failover or scalable configurations.

HA and Scalable Data Service Support

The Sun Cluster software provides preconfigured components that support the following HA data services. The release of new data service agents does not correspond exactly with updates of the Sun Cluster software framework. Some agents are available from various divisions within Sun, rather than being on the Sun Cluster software data services CD-ROM. These are the components available at the time of the writing of this course:

- Sun Cluster HA for ORACLE Server (failover)
- Sun Cluster HA for ORACLE E-Business Suite (failover)
- Sun Cluster HA for Sun Java System Web Server (failover or scalable)
- Sun Cluster HA for Sun Java System Web Proxy Server (failover)
- Sun Cluster HA for Sun Java System Application Server SE/PE (failover)
- Sun Cluster HA for Sun Java System Directory Server (failover)
- Sun Cluster HA for Sun Java System Message Queue software (failover)
- Sun Cluster HA for Sun StorEdge Availability Suite (failover)
- Sun Cluster HA for Sun StorEdge Enterprise Backup Software (EBS) (failover)
- Sun Cluster HA for Apache Web Server (failover or scalable)
- Sun Cluster HA for Apache Proxy Server (failover)
- Sun Cluster HA for Apache Tomcat (failover or scalable)
- Sun Cluster HA for Domain Name System (DNS) (failover)
- Sun Cluster HA for Network File System (NFS) (failover)

- Sun Cluster HA for Dynamic Host Configuration Protocol (DHCP) (failover)
- Sun Cluster HA for SAP (failover or scalable)
- Sun Cluster HA for SAP LiveCache (failover)
- Sun Cluster HA for Sybase ASE (failover)
- Sun Cluster HA for VERITAS NetBackup (failover)
- Sun Cluster HA for BroadVision (scalable)
- Sun Cluster HA for Siebel (failover)
- Sun Cluster HA for Samba (failover)
- Sun Cluster HA for BEA WebLogic Application Server (failover)
- Sun Cluster HA for IBM Websphere MQ (failover)
- Sun Cluster HA for IBM Websphere MQ Integrator (failover)
- Sun Cluster HA for MySQL (failover)
- Sun Cluster HA for SWIFTalliance (failover)
- Sun Cluster HA for Solaris Containers (Solaris 10 OS zones – failover)

Exploring the Sun Cluster Software HA Framework

The Sun Cluster software framework is the software layer that provides generic cluster services to the nodes in the cluster, regardless of which applications are running in the cluster. The Sun Cluster software framework is implemented as a series of daemons and kernel modules. One of the advantages of the Sun Cluster software environment is that so much of the framework is in the kernel, where it is fast, always memory-resident, and reliable. Some of the services provided by the framework are described in the following sections.

Node Fault Monitoring and Cluster Membership

The cluster membership monitor (CMM) is kernel-resident on each node and detects major cluster status changes, such as loss of communication between one or more nodes. The CMM relies on the transport kernel module to generate heartbeats across the transport medium to other nodes in the cluster. If the heartbeat from any node is not detected within a defined time-out period, it is considered as having failed, and a cluster reconfiguration is initiated to renegotiate cluster membership.

Network Fault Monitoring

Both the public network interfaces and the cluster transport interfaces are monitored for potential failures.

Public Network Management

The Sun Cluster 3.1 software environment requires the use of IPMP, a standard Solaris OS feature, to control interface failures on a node. The Sun Cluster software adds a layer of monitoring to detect total network failure on one node and drive possible failover of applications to another node.

Cluster Transport Monitoring

The cluster transport interfaces are monitored on each node. If an active cluster transport interface on any node is determined to be inoperative, all nodes route interconnect traffic to functional transport interfaces. The failure is transparent to Sun Cluster software applications.

Disk Path Monitoring

Disk path monitoring (DPM) provides the capability to monitor disk paths with a new cluster daemon process and a command line interface (CLI) command. DPM is not supported on nodes that run versions that were released prior to Sun Cluster 3.1 5/03 software.

DPM improves the overall reliability of failover and switchover by monitoring the secondary disk-path availability. It is possible to monitor disk paths to a single node or to all nodes in the cluster. The disk path failure detection mechanism generates an event through the Cluster Event Framework and allows manual intervention.

Application Traffic Striping

Applications written correctly can use the transport for data transfer. This feature stripes IP traffic sent to the per-node logical IP addresses across all private interconnects. Transmission Control Protocol (TCP) traffic is striped on a per connection granularity. User Datagram Protocol (UDP) traffic is striped on a per packet basis. The cluster framework uses the virtual network device `clprivnet0` for these transactions. This network interface is visible with `ifconfig`. No configuration is required as part of the Sun Cluster 3.1 software framework.

Figure 1-8 shows the benefit of application traffic striping to a cluster aware application. The application receives the benefit of striping across all of the physical private interconnects, but only needs to be aware of a single IP address on each node, configured on that nodes `clprivnet0` adapter.

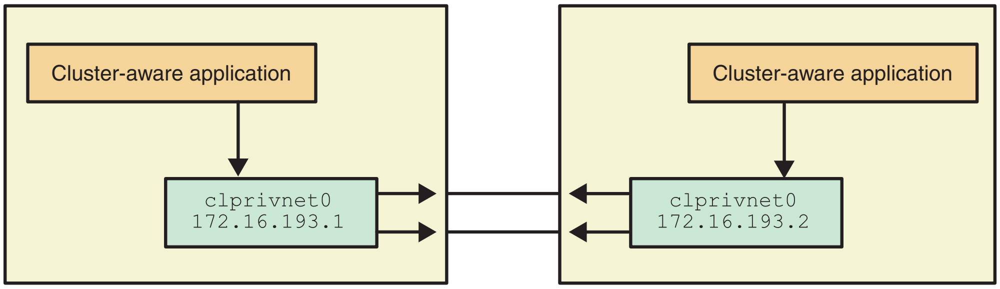


Figure 1-8 Application Traffic Striping

Cluster Configuration Repository

General cluster configuration information is stored in global configuration files collectively referred to as the cluster configuration repository (CCR). The CCR must be kept consistent between all nodes and is a critical element that enables each node to be aware of its potential role as a designated backup system.

Caution – Never attempt to modify any of the CCR-related files. The files contain generation number information that is critical to the operation of the cluster software. The CCR information is automatically modified as the result of administrative command execution and cluster status changes.

The CCR structures contain the following types of information:

- Cluster and node names
- Cluster transport configuration
- The names of registered VERITAS disk groups or Solaris Volume Manager software disksets
- A list of nodes that can master each disk group
- Information about NAS devices
- Data service operational parameter values (timeouts)
- Paths to data service callback methods
- Disk ID (DID) device configuration
- Current cluster status

The CCR is accessed when error or recovery situations occur or when there has been a general cluster status change, such as a node leaving or joining the cluster.

Global Storage Services

The Sun Cluster software framework provides global storage services, a feature which greatly distinguishes the Sun Cluster software product. Not only do these features enable scalable applications to run in the cluster, they also provide a much more flexible environment for failover services by freeing applications to run on nodes that are not physically connected to the data.

It is important to understand the differences and relationships between the following services:

- Global naming (DID devices)
- Global devices
- Global file system

Disk ID Devices (DIDs)

The DID feature provides a unique device name for every disk drive, compact disk, read-only-memory (CD-ROM) drive, or tape drive in the cluster. Multiported disks that might have different logical names on different nodes (different controller numbers) are given a cluster-wide unique DID instance number. Different disks that may, each on its own node, use the same logical name (for example, `c0t0d0` for each node's root disk) are each given different unique DID instance numbers.

Figure 1-9 demonstrates the relationship between normal Solaris OS logical path names and DID instances.

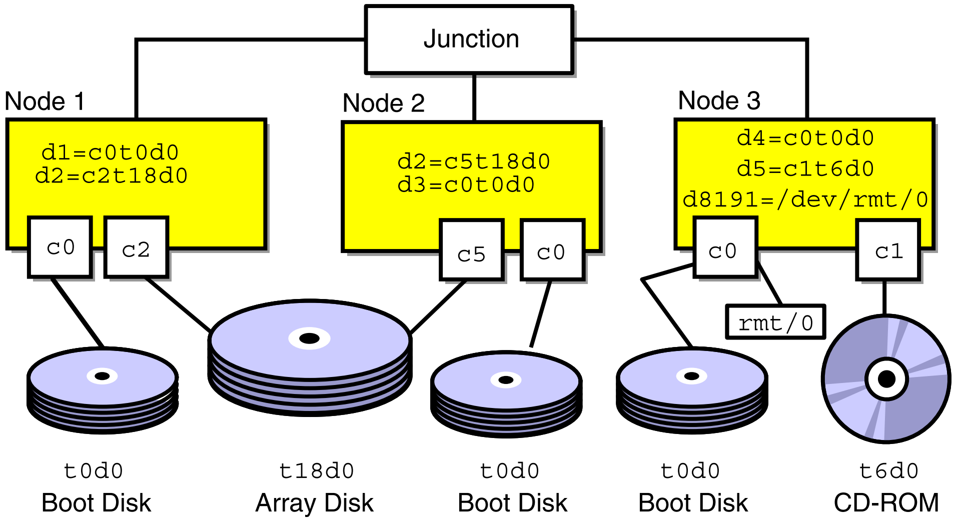


Figure 1-9 DID Driver Devices

Device files are created for each of the normal eight Solaris OS disk partitions in both the /dev/did/dsk and /dev/did/rdsk directories; for example: /dev/did/dsk/d2s3 and /dev/did/rdsk/d2s3.

It is important to note that DIDs themselves are just a global naming scheme and not a global access scheme.

DIDs are used as components of Solaris Volume Manager software volumes and in choosing cluster quorum devices, as described in Module 3, “Preparing for Installation and Understanding Quorum Devices.”

DIDs are *not* used as components of VERITAS Volume Manager volumes.

Global Devices

The *global devices* feature of Sun Cluster 3.1 software provides simultaneous access to the raw (character) device associated with storage devices from all nodes, regardless of where the storage is physically attached.

This includes individual DID disk devices, CD-ROMs and tapes, as well as VERITAS Volume Manager volumes and Solaris Volume Manager volumes.

Shared global devices are used most often in the cluster with VERITAS Volume Manager and Solaris Volume Manager software devices. The volume management software is unaware of the global device service implemented on it.

The Sun Cluster 3.1 software framework manages automatic failover of the *primary node* for global device groups. All nodes use the same device path, but only the primary node for a particular device actually talks through the storage medium to the disk device. All other nodes access the device by communicating with the primary node through the cluster transport. In Figure 1-10, all nodes have simultaneous access to the device `/dev/vx/rdsk/nfsdg/nfsvol`. Node 2 becomes the primary node if Node 1 fails.

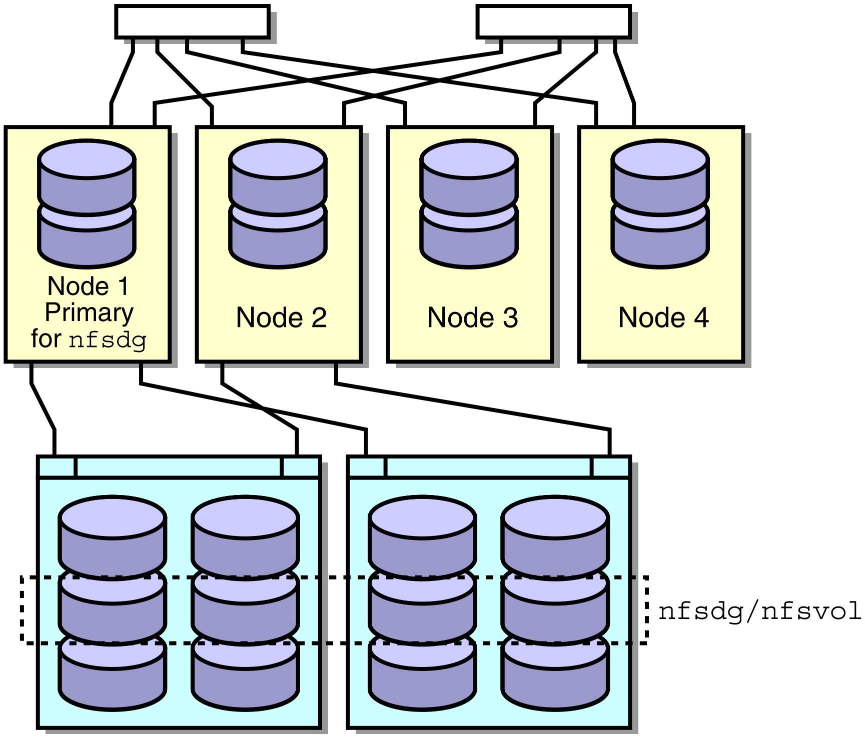


Figure 1-10 Node Access Diagram

Device Files for Global Devices

The Sun Cluster 3.1 software maintains a special file system on each node, completely dedicated to storing the device files for global devices. This file system has the mount point `/global/.devices/node@`*`nodeID`*, where *`nodeID`* is an integer representing a node in the cluster. This requires a dedicated disk partition on each node, usually on the boot disk.

All of the `/global/.devices` file systems—one for each node—are visible from each node. In other words, they are examples of global file systems. The global file system feature is described more on the following pages.

```
# df -k
/dev/vx/dsk/rootdisk_25vol
    95702    5026   81106     6%    /global/.devices/node@2
/dev/vx/dsk/rootdisk_15vol
    95702    5026   81106     6%    /global/.devices/node@1
```

The device names under the `/global/.devices/node@`*`nodeID`* arena can be used directly, but, because they are clumsy, the Sun Cluster 3.1 environment provides symbolic links into this namespace.

For VERITAS Volume Manager and Solaris Volume Manager software, the Sun Cluster software links the standard device access directories into the global namespace:

```
proto192:/dev/vx# ls -l /dev/vx/rdsk/nfsdg
lrwxrwxrwx   1 root     root          40 Nov 25 03:57
         /dev/vx/rdsk/nfsdg ->/global/.devices/node@1/dev/vx/rdsk/nfsdg/
```

For individual DID devices, the standard directories `/dev/did/dsk`, `/dev/did/rdsk`, and `/dev/did/rmt` are *not* global access paths. Instead, Sun Cluster software creates alternate path names under the `/dev/global` directory that link into the global device space:

```
proto192:/dev/md/nfsds# ls -l /dev/global
lrwxrwxrwx   1 root     other         34 Nov  6 13:05
         /dev/global -> /global/.devices/node@1/dev/global/
proto192:/dev/md/nfsds# ls -l /dev/global/rdsk/d3s0
lrwxrwxrwx   1 root     root          39 Nov  4 17:43
         /dev/global/rdsk/d3s0 -> ../../../devices/pseudo/did@0:3,3s0,raw
proto192:/dev/md/nfsds# ls -l /dev/global/rmt/1
lrwxrwxrwx   1 root     root          39 Nov  4 17:43
         /dev/global/rmt/1 -> ../../../devices/pseudo/did@8191,1,tp
```

Note – You *do* have raw (character) device access from one node, through the /dev/global device paths, to the boot disks of other nodes. In other words, while you can not mount one node's root disk from another node, you *can* overwrite it with newfs or dd. It is not necessarily advisable to take advantage of this feature.

Global File Systems

The global file system feature makes file systems simultaneously available on all nodes, regardless of their physical location.

The global file system capability is independent of the structure of the actual file system layout on disk. The UNIX® File System (ufs), VERITAS File System (VxFS) and High Sierra File System (hsfs) are supported.

The Sun Cluster software makes a file system global with a global mount option. This is normally in the /etc/vfstab file but can be put on the command line of a standard mount command:

```
# mount -o global,logging /dev/vx/dsk/nfs-dg/vol-01 /global/nfs
```

The equivalent mount entry in the /etc/vfstab file is:

```
/dev/vx/dsk/nfs-dg/vol-01 /dev/vx/rdsk/nfs-dg/vol-01 \
/global/nfs  ufs 2 yes global,logging
```

The global file system works on the same principle as the global device feature. That is, only one node at a time is the primary and actually talks to the underlying file system. All other nodes use normal file semantics but actually communicate with the primary over the cluster transport. The primary for a global file system built on a global device is always the same as the primary for the global device.

The global file system is also known by the following names:

- Cluster file system
- Proxy file system

Failover (Non-Global) File Systems in the Cluster

Sun Cluster 3.1 software also has support in the cluster for *failover file system access*. That is, file systems that are not globally, simultaneously, available from every node, but available only one node at a time, on a node which is running a service and has a physical connection to the storage in question.

Failover File System Access Is *Not* for Scalable Services

Failover file system access is appropriate for failover services that will run only on the nodes physically connected to storage devices.

Failover file system access is *not* usable for scalable services that require global file system access.

Failover file system access, when used appropriately, can have a performance benefit over global file system access. There is overhead in the global file system infrastructure of maintaining replicated state information on multiple nodes simultaneously.

Exercise: Guided Tour of the Training Lab

At the end of this exercise you should be able to identify Sun Cluster hardware components located in the training lab.

Preparation

No special preparation is required for this lab.

Task

You will participate in a guided tour of the training lab. While participating in the guided tour, you will identify the Sun Cluster hardware components including the cluster nodes, terminal concentrator, and administrative workstation.

If this course is being run without local access to the equipment, take this opportunity to review the essentials of the Sun Cluster hardware and software, or to familiarize yourself with the remote lab environment.

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 2

Exploring Node Console Connectivity and the Cluster Console Software

Objectives

Upon completion of this module, you should be able to:

- Describe the different methods for accessing a console
- Access the Node consoles on domain-based servers
- Configure the Sun Cluster console software on the administration workstation
- Use the Cluster console tools

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- What is the trade-off between convenience and security in different console access methods?
- How do you reach the node console on domain-based clusters?
- What benefits does using a terminal concentrator give you?
- Is installation of the administration workstation software essential for proper cluster operation?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)

Accessing the Cluster Node Consoles

This section describes different methods for achieving access to the Sun Cluster 3.1 node consoles. It is expected that a Sun Cluster 3.1 environment administrator:

- Does *not* require node console access for most operations described during the duration of the course. Most cluster operations require only that you be logged in on a cluster node as `root` or as a user with cluster authorizations in the Role-Based Access Control (RBAC) subsystem. It is acceptable to have direct `telnet`, `rlogin`, or `ssh` access to the node.
- *Must* have console node access for certain emergency and informational purposes. If a node is failing to boot, the cluster administrator will have to access the node console to figure out why. The cluster administrator might like to observe boot messages even in normal, functioning clusters.

The following pages describe the essential difference between console connectivity on "traditional" nodes that are not domain-based and console connectivity on the domain-based nodes (Sun Enterprise 10000 servers, Sun Fire Midrange Servers (E4900, E6900 and 3800–6800 servers), and Sun Fire High-End Servers E25K/E20K/15K/12K servers).

Accessing Serial Port Consoles on Traditional Nodes

Traditional Sun Cluster 3.1 nodes usually use serial port `ttyA` as the console. Using a graphics console with a Sun keyboard and monitor is supported, as long as it is supported on the base server platform.

The rule for console connectivity is simple. You can connect to the node `ttyA` interfaces any way you like, as long as whatever device you have connected directly to them does not spuriously issue `BREAK` signals on the serial line.

`BREAK` signals on the serial port will bring a cluster node to the `OK` prompt, killing all cluster operations on that node.

It is possible to disable node recognition of a BREAK signal, either by a hardware keyswitch position (on some nodes), a software keyswitch position (on the Sun Fire 3800–6800 servers and Sun Fire 15K/12K servers), or a file setting (on all nodes).

For those servers with a hardware keyswitch, turn the key to the third position to power the server on and disable the BREAK signal.

For those servers with a software keyswitch, issue the setkeyswitch command with the secure option to power the server on and disable the BREAK signal.

For all servers, while running Solaris OS, uncomment the line KEYBOARD_ABORT=alternate in /etc/default/kbd to disable receipt of the normal BREAK signal through the serial port. This setting takes effect on boot, or by running the kbd -i command as root.

The Alternate Break signal is defined by the particular serial port driver you happen to have on your system. You can use the prtconf command to figure out the name of your serial port driver, and then use man *serial-driver* to figure out the sequence. For example, for the zs driver, the sequence is carriage return, tilde (~), and control-B: CR ~ CTRL-B. When the Alternate Break sequence is in effect, only serial console devices are affected.

Access Serial Port Node Consoles Using a Terminal Concentrator

One of the popular ways of accessing traditional node consoles is through a TC, a device which listens for connections on the network and passes through traffic (unencapsulating and reencapsulating all the TCP/IP headers) to the various serial ports.

A TC is also known as a Network Terminal Server (NTS).

Figure 2-1 shows a terminal concentrator network and serial port interfaces. The node public network interfaces are not shown. While you can attach the TC to the public net, most security-conscious administrators would attach it to a private management network.

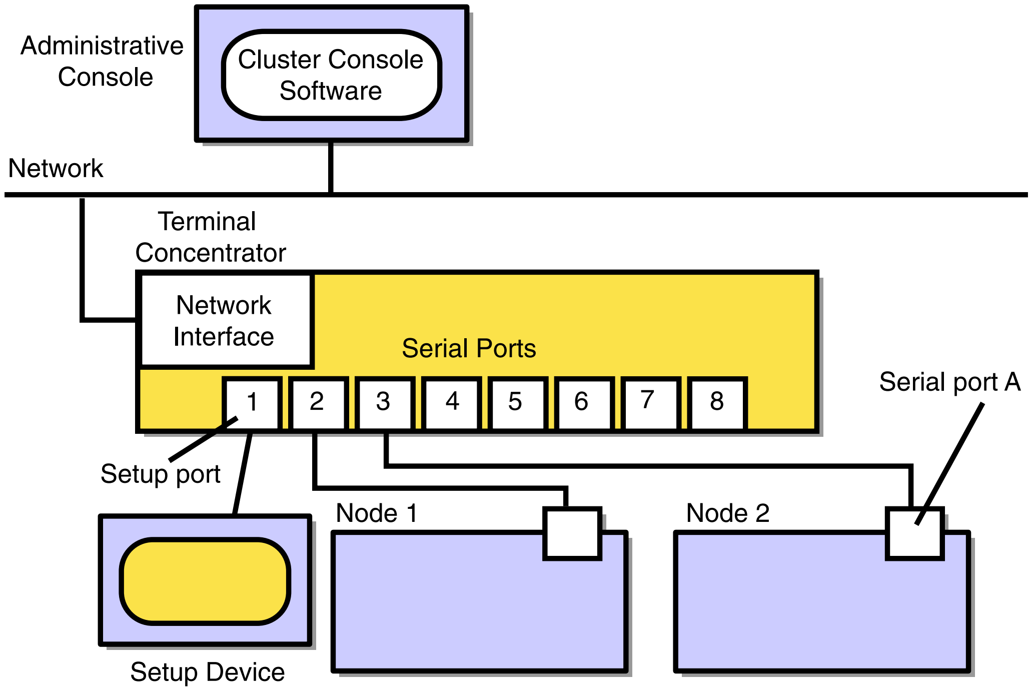


Figure 2-1 TC Network and Serial Port Interfaces

Most TCs enable you to administer TCP *pass-through* ports on the TC. When you connect with telnet to the TC's IP address and pass through port, the TC transfers traffic directly to the appropriate serial port (perhaps with an additional password challenge).

Sun Terminal Concentrator (Sun NTS)

Sun rebrands a terminal concentrator, the Xylogics™ NTS. Sun guarantees that this TC does not spuriously issue `BREAK` signals when it is powered on and off, for example.

The Sun TC supports `telnet` and `rlogin` access. The TCP ports it uses as pass-through ports for `telnet` access are 5002 for serial port 2, 5003 for serial port 3, and so forth.

When you use telnet to connect to the Sun TC from an administrative workstation using, for example, the following command, the Sun TC passes you through directly to serial port 2, connected to your first cluster node.

```
# telnet tc-ipname 5002
```

The Sun TC supports an extra level of password challenge, a per-port password which you have to enter before going through to the serial port.

Full instructions on installing and configuring the Sun TC are in Appendix A, “Terminal Concentrator.”

Other Terminal Concentrators

You can choose any type of TC as long as it does not issue BREAK signals on the serial ports when it is powered on, powered off, reset, or any other time that might be considered spurious.

If your TC cannot meet that requirement, you can still disable recognition of the BREAK signal or enable an alternate abort signal for your node.

Some terminal concentrators support Secure Shell (the Sun NTS does not support Secure Shell). This might influence your choice, if you are concerned about passing TC traffic in the clear on the network.

Alternatives to a Terminal Concentrator

Some possible alternatives to the TC include the following:

- Use a Sun graphics console. This requires physical access to the node for console work. This is most secure, but least convenient.
- Use dumb terminals for each node console. If these are in a secure physical environment, this is certainly your most secure but least convenient method.
- Use a workstation that has two serial ports as a "`tip` launchpad," especially for a cluster with only two nodes.

  You can attach a workstation on the network exactly as you would place a TC, and attach its serial ports to the node consoles. You then add lines to the `/etc/remote` file of the Solaris OS workstation as follows:

  ```
  node1:\
     :dv=/dev/term/a:br#9600:el=^C^S^Q^U^D:ie=%$:oe=^D:
  node2:\
     :dv=/dev/term/b:br#9600:el=^C^S^Q^U^D:ie=%$:oe=^D:
  ```

  This allows you to access node consoles by accessing the launchpad workstation and manually typing `tip node1` or `tip node2`.

  One advantage of using a Solaris OS workstation instead of a TC is that it is easier to tighten the security therein. You could easily, for example, disable `telnet` and `rlogin` access and require that administrators access the `tip` launchpad by means of Secure Shell.
- Use network access to the console on servers that specifically support the following types of remote control:
  - Advanced Lights Out Manager (ALOM)
  - Remote System control (RSC)
  - Service Processor (on Sun Fire V40z x86 nodes)

  These servers provide a combination of firmware and hardware that provides direct access to the server's hardware and console. Dedicated Ethernet ports and dedicated IP addresses are used for remote console access and server management.

Accessing the Node Consoles on Domain-Based Servers

Domain-based servers have no serial port console access for the domains. Instead, you access the main system support processor (SSP), for Sun Enterprise 10000 servers or system controller (SC), for Sun Fire servers and use a *virtual console* protocol to access the console for each node.

Sun Enterprise™ 10000 Servers

The SSP and its backup sit physically outside the server frame and run the Solaris OS and SSP software.

To access the node console, you will always have to log in manually to the SSP as the user named `ssp` and manually invoke the `netcon` command for the appropriate domain.

Sun Fire™ High-End Servers

The SC and its backup sit physically inside the server frame, but act much like the Sun Enterprise 10000 system SSP. They run the Solaris OS with System Management Services (SMS) software.

To access the node console, you have to log in manually to the SC as the domain administrator for the node in question (this can be a different user for each domain) and manually invoke the `console` command for the appropriate domain.

Sun Fire Midrange Servers

The SC and its backup sit physically inside the server frame. They run a unique, firmware-based operating environment known as VxWorks. The SCs run an application, based on Java technology, known as ScApp that is dedicated to managing the server.

You can configure ScApp to provide remote access through `telnet`, through `ssh`, or neither. If you configure `telnet` access, the SC runs a *TC-like* emulation where it listens on TCP ports 5001, 5002, and so forth. Accessing the SC with one of these ports passes you through (perhaps with an extra password challenge) to a domain console or domain shell.

Describing Sun Cluster Console Software for an Administration Workstation

The Sun Cluster 3.1 software includes a small amount of software to be installed on an administrative workstation. This software can be installed on any Solaris OS workstation running Solaris OS version 2.6 and above.

The console software is a convenience which displays new windows for you to the cluster nodes and issues `telnet` or `rlogin` commands to connect you either directly to the nodes or to the node console access point.

The console software includes a *common window* feature which lets you issue keystrokes simultaneously to all the nodes in the cluster (or even to all the nodes in multiple clusters).

The Sun Cluster 3.1 software environment itself has no dependency on the administration workstation's console software. It is just an administrative convenience.

Console Software Installation

The administrative console software is contained in a single package, `SUNWccon`. The `SUNWccon` package is installed manually from the Sun Cluster 3.1 software distribution CD-ROM.

Cluster Console Window Variations

The three variations of the cluster console tool each use a different method to access the cluster hosts. They all look and behave the same way. These are the following:

- Cluster console (`cconsole`)

  The `cconsole` program accesses the node consoles through the TC or other remote console access method. Depending on your access method, you might be directly connected to the console node, or you might have to respond to additional password challenges, or you might have to login to an SSP or SC and manually connect to the node console.

- Cluster console (`crlogin`)

  The `crlogin` program accesses the nodes directly using the `rlogin` command.

- Cluster console (`ctelnet`)

  The `ctelnet` program accesses the nodes directly using the `telnet` command.

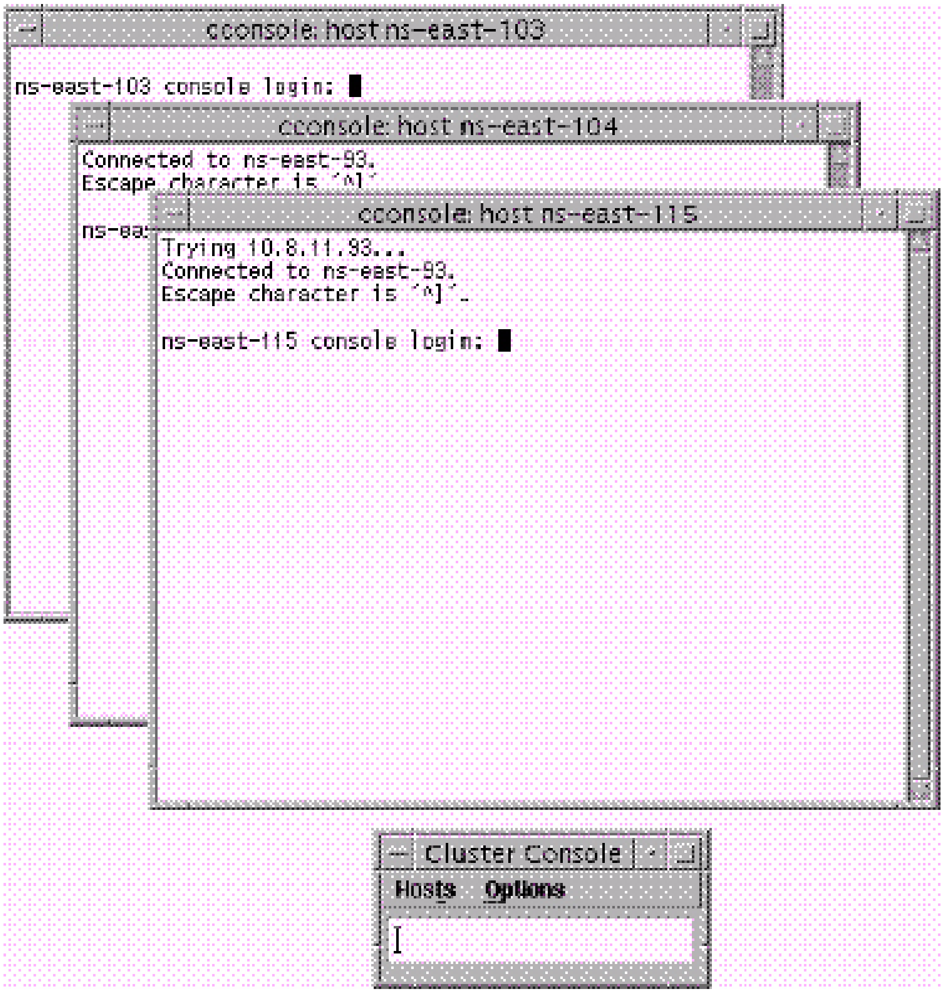
Note that it is possible in certain scenarios that some tools might be unusable, but you might want to install this software to use other tools.

For example, if you are using dumb terminals to attach to the node consoles, you will not be able to use the `cconsole` variation, but you can use the other two after the nodes are booted.

Alternatively, you might have, for example, a Sun Fire 6800 server where `telnet` and `rlogin` are disabled in the domains, and only `ssh` is working as a remote login method. You might be able to use *only* the `cconsole` variation, but not the other two variations.

Cluster Console Tools Look and Feel

All the tools have the same general look and feel. The tool automatically shows one new window for each node, and a small common keystroke window (Figure 2-2). You can type in each individual window as desired. Input directed to the common window is automatically replicated to all the other windows.


cconsole: host ns-east-103
cconsole: host ns-east-104
cconsole: host ns-east-115
Cluster Console
Hosts
Options

```
ns-east-103 console login: █
```

```
Connected to ns-east-93.
Escape character is '^]'.

ns-ea
```

```
Trying 10.8.11.93...
Connected to ns-east-93.
Escape character is '^]'.

ns-east-115 console login: █
```

Figure 2-2 Cluster Console Windows

Start the Tools Manually

As shown, you can use the tools manually to connect to a single cluster node or to the entire cluster.

```
# /opt/SUNWcluster/bin/cconsole node1 &
# /opt/SUNWcluster/bin/ctelnet my-cluster &
# /opt/SUNWcluster/bin/crlogin node3 &
```

Cluster Console Host Windows

Each node in the cluster has a host window. You can enter commands in each host window separately.

Set the `TERM` environment variable to `vt100` or `dtterm` for best operation.

Cluster Console Common Window

The common window, shown in Figure 2-3, allows you to enter commands to all host system windows at the same time. All of the windows are tied together, so when you move the common window, the host windows follow. The Options menu allows you to ungroup the windows, move them into a new arrangement, and group them again.

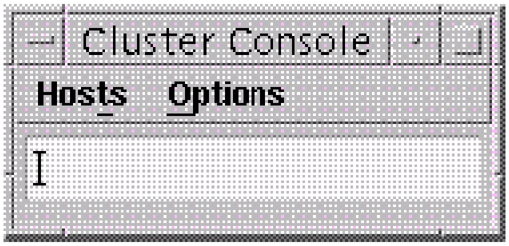


Figure 2-3 Cluster Console Common Window

Cluster Control Panel

As shown in Figure 2-4, the Cluster Control Panel provides centralized access to three variations of the cluster console tool.

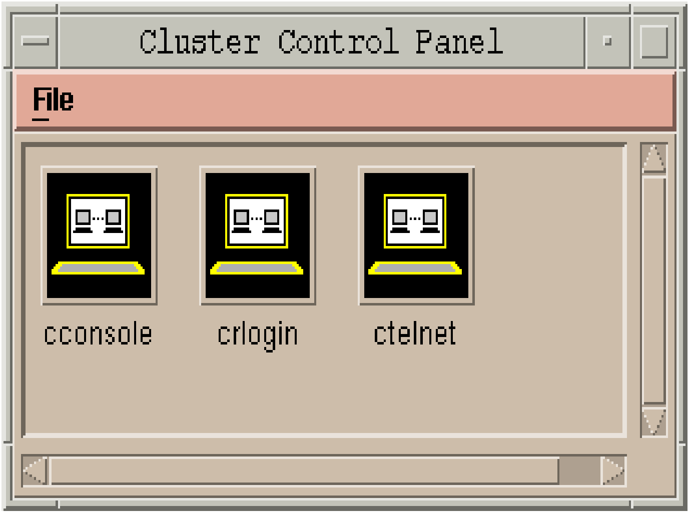


Figure 2-4 Cluster Control Panel

Starting the Cluster Control Panel

To start the Cluster Control Panel, type the following command:

```
# /opt/SUNWcluster/bin/ccp [clustername] &
```

Configuring Cluster Console Tools

All of the necessary information needed for the cluster administration tools that run on the administrative console is configured in two files. The files are:

- The /etc/clusters file.
- The /etc/serialports file (for cconsole variation only).

When you install the Sun Cluster console software on the administrative console, you must manually create the clusters and serialports files and populate them with the necessary information.

Configuring the /etc/clusters File

The /etc/clusters file contains the name of a cluster followed by the names of the nodes that are part of the cluster.

The following is a typical entry in the /etc/clusters file:

```
sc-cluster sc-node1 sc-node2
```

The single-line entry defines a cluster named sc-cluster which has two nodes named sc-node1 and sc-node2.

Note – The cluster name is purely arbitrary. The Sun Cluster 3.1 software itself will have you define a cluster name, which will likely agree with this one, although nothing will break if it does not agree.

You can define many different clusters in a single /etc/clusters file, so you can administer several clusters from a single administrative console.

Configuring the /etc/serialports File

The /etc/serialports file defines the remote path to the console for each node defined in the /etc/clusters file. You must enter the paths to all nodes in all of your clusters in this file. This file uses three columns: the node name listed in /etc/inet/hosts, the host name of the workstation or device providing the connection as listed in /etc/inet/hosts, and the port for telnet to use for the connection to or through this device.

The following are typical entries in the /etc/serialports file when utilizing a terminal concentrator to pass through to the physical serial connection (ttyA) on each cluster node:

```
sc-node1 sc-tc 5002
sc-node2 sc-tc 5003
```

For the Sun Enterprise 10000 server, the columns are the name of the domain, the name of the SSP workstation, and port 23. When the cconsole *cluster* command is executed, you are connected to port 23 on the SSP. This opens up a telnet session for each domain listed. The actual login to the SSP is done manually in each window. Once logged in, a netcon session is manually started for the domain in each window.

```
sc-10knode1 sc10k-ssp 23
sc-10knode2 sc10k-ssp 23
```

For the Sun Fire E25K/E20K/15K/12K server, the columns are the name of the domain, the name of the main system controller, and port 23. When the cconsole *cluster* command is executed, you are connected to port 23 on the main system controller. This opens up a telnet session for each domain listed. The actual login to the main system controller is done manually in each window using the login of the domain administrator for that domain. Once logged in, a console session is manually started for that domain.

```
sc-15knode1 sf15k-mainsc 23
sc-15knode2 sf15k-mainsc 23
```

For a system using ALOM or RSC access instead of a TC, the columns display the name of the node, the name corresponding to the dedicated IP for ALOM/RSC for that node, and port 23. When the `cconsole` *`cluster`* command is executed, you are connected to the ALOM or RSC for that node. The actual login to the node console is done manually in each window.

```
node1  node1-sc  23
node2  node2-sc  23
```

If you are using a `tip` launchpad, the columns display the name of the node, the name of the launchpad, and port 23. The actual login to the launchpad is done manually in each window. The actual connection to each node console is made by executing the `tip` command and specifying the serial port used for that node in each window.

```
node1 sc-tip-ws 23
node2 sc-tip-ws 23
```

For the Sun Fire 3800–6800 servers and E4900/E6900 servers, the columns display the domain name, the Sun Fire server main system controller name, and a TCP port number similar to those used with a terminal concentrator. Port 5000 connects you to the platform shell, ports 5001, 5002, 5003, or 5004 connect you to the domain shell or domain console for domains A, B, C, or D, respectively. Connection to the domain shell or console is dependent upon the current state of the domain.

```
sf1_node1 sf1_mainsc 5001
sf1_node2 sf1_mainsc 5002
```

Using the Tools When ssh Is Required or Preferred to Access the Console

Certain environments may require ssh access to get to the console, or you might prefer to use ssh access even if telnet is also available. Examples are:

- Terminal Concentrators which have ssh enabled
- The tip launchpad on which you have enabled ssh
- Sun High-end Servers (ssh to the SC) and E10000 (ssh to the SSP)
- Sun Fire Midrange System Controllers on which you have enabled ssh (telnet will be automatically disabled)
- Sun Fire v40z Service Processor (supports only ssh, not telnet)

In order to use the tools in these scenarios, you can create an /etc/serialports file like the following:

```
# example for a 6800 (cluster in a box)
node1 my6800sc  22
node2 my6800sc  22

# example for a 15K (cluster in a box)
node1 my15kmainsc 22
node2 my15kmainsc 22

# example for two v40z's (each has its own sp)
v40node1 node1-sp 22
v40node2 node2-sp 22
```

Now create a file called, for example, /etc/consolewithssh. This is used by the customized telnet script below to figure out which console access devices require ssh access, and what the ssh user name should be. Note that the ssh server on the Sun Fire Midrange system controller ignores the user name, so the file just includes anything as a place holder. A Sun Fire 15K system controller would have a user defined on the system controller (in this example, mysmsuser), that is defined as a domain administrator of both domains in question).

```
# console access device / user
my6800sc       anything

node1-sp       admin
node2-sp       admin

my15kmainsc    mysmsuser
```

Now you can make a variation of the regular `telnet` executable:

```
# mv /usr/bin/telnet /usr/bin/telnet.real
# vi /usr/bin/telnet
#!/bin/ksh

HOST=$1
PORT=$2
LOGINFORSSH=$(cat /etc/consolewithssh |awk '$1 == "'$HOST'" {print $2}')
if [[ $LOGINFORSSH != "" ]]
then
        /usr/bin/ssh -p $PORT -l $LOGINFORSSH $HOST
else
        /usr/bin/telnet.real "$@"
fi

# chmod a+x /usr/bin/telnet
```

Altering the `finger` Executable

The `cconsole` variation of the cluster console tools calls the command `finger @`*`console-access-device`*. If the console access device happens to be a Sun Terminal Concentrator, the `cconsole` can report using this command if the particular serial ports on the TC are busy.

The `cconsole` is unaffected if a console access device immediately returns a `Connection Refused` error code from the `finger` command. However, certain devices, namely the V40z System Processor, do not return *anything*: the command will freeze up. In this case, you may need to alter the `finger` command so that `cconsole` can proceed:

```
# mv /usr/bin/finger /usr/bin/finger.real
# vi /usr/bin/finger
#!/bin/ksh

if [[ $1 = @node1-sp || $1 = @node2-sp ]]
then
        exit 0
else
        /usr/bin/finger.real "$@"
fi
#chmod a+x /usr/bin/finger
```

Using the Tools When `ssh` Is Required or Preferred for Direct Access to the Nodes

If you want to use the cluster access tools for direct access to nodes using `ssh`, you can make the same sort of variations to the `/usr/bin/rlogin` command, and then use the `crlogin` variation of the tools. For example, the following globally substitutes `ssh` instead of the real `rlogin`:

```
# mv /usr/bin/rlogin /usr/bin/rlogin.real
# vi /usr/bin/rlogin
#!/bin/ksh

/usr/bin/ssh "$@"

# chmod a+x /usr/bin/rlogin
```

Exercise: Configuring the Administrative Console

In this exercise, you complete the following tasks:

- Task 1 – Updating Host Name Resolution
- Task 2 – Installing the Cluster Console Software
- Task 3 – Verifying the Administrative Console Environment
- Task 4 – Configuring the `/etc/clusters` File
- Task 5 – Configuring the `/etc/serialports` File
- Task 6 – Starting the `cconsole` Tool
- Task 7 – Using the `ccp` Control Panel

Preparation

This exercise assumes that the Solaris 10 OS software is already installed on all of the cluster systems. Perform the following steps to prepare for the lab:

1. Ask your instructor for the name assigned to your cluster.

   Cluster name: _______________________________

2. Record the information in Table 2-1 about your assigned cluster before proceeding with this exercise.

Table 2-1 Cluster Names and Addresses

System	Name	IP Address
Administrative console		
TC		
Node 1		
Node 2		
Node 3 (if any)		

3. Ask your instructor for the location of the Sun Cluster software.

   Software location: _______________________________

Task 1 – Updating Host Name Resolution

Even though your site might use Network Information Service (NIS) or DNS to resolve host names, it can be beneficial to resolve the names locally on the administrative console and cluster hosts. This can be valuable in the case of naming service failures. The `cconsole` program does not start unless it can first resolve the host names in the `/etc/clusters` file.

Perform the following steps:

1. If necessary, edit the `/etc/hosts` file on your administrative console, and add the IP addresses and names of the TC and the host systems in your cluster.
2. Verify that the `/etc/nsswitch.conf` file entry for `hosts` has `files` listed first.

   ```
   hosts: files nis
   ```

Task 2 – Installing the Cluster Console Software

Perform the following steps to install the cluster console software:

1. Log in to your administrative console as user `root`.

Note – If your administrative console is in a remote lab, your instructor will assign you a user login. In this case the cluster console software is already installed.

2. Check to see if the cluster console software is already installed:

   ```
   (# or $) pkginfo SUNWccon
   ```

   If it is already installed, you can skip the rest of the steps of this task.
3. Move to the Sun Cluster 3.1 packages directory.
4. Verify that you are in the correct location:

   ```
   # ls SUNWccon
   SUNWccon
   ```

5. Install the cluster console software package:

   ```
   # pkgadd -d . SUNWccon
   ```

Task 3 – Verifying the Administrative Console Environment

Perform the following steps to verify the administrative console:

1. Verify that the following search paths and variables are present in the `.profile` file in your home directory:

```
PATH=$PATH:/opt/SUNWcluster/bin
MANPATH=$MANPATH:/opt/SUNWcluster/man
EDITOR=/usr/bin/vi
export PATH MANPATH EDITOR
```

Note – Create the `.profile` file in your home directory if necessary, and add the changes.

2. Execute the `.profile` file to verify changes that have been made:

```
(# or $) $HOME/.profile
```

Note – You can also log out and log in again to set the new variables.

Task 4 – Configuring the `/etc/clusters` File

The `/etc/clusters` file has a single line entry for each cluster you intend to monitor. The entries are in the form:

```
clustername host1name host2name host3name host4name
```

Sample `/etc/clusters` File

```
sc-cluster pnode1 pnode2 pnode3
```

Perform the following to configure the /etc/clusters file:

> Edit the /etc/clusters file, and add a line using the cluster and node names assigned to your system.

Note – If you are using a remote lab environment, the /etc/clusters file may already be set up for you. Examine the file.

Task 5 – Configuring the /etc/serialports File

The /etc/serialports file has an entry for each cluster host describing the connection path. This example is for a three-node cluster using a Sun NTS.

hostname tcname tcport

Sample /etc/serialports File

```
pnode1     cluster-tc     5002
pnode2     cluster-tc     5003
pnode3     cluster-tc     5004
```

Perform the following to configure the /etc/serialports file:

> Edit the /etc/serialports file and add lines using the node and TC names assigned to your system.

Note – In the remote lab environment, this file may already be set up for you. Examine the file.

Task 6 – Starting the `cconsole` Tool

This section provides a good functional verification of the TC in addition to the environment configuration.

Perform the following steps to start the `cconsole` tool:

1. Make sure power is on for the TC and all of the cluster hosts.
2. Start the `cconsole` tool on the administrative console:

   ```
   (# or $) cconsole clustername &
   ```

Note – Substitute the name of your cluster for *`clustername`*.

3. Place the cursor in the `cconsole` Common window, and press the Return key several times. You should see a response on all of the cluster host windows. If not, ask your instructor for assistance.
4. If your console device is an ALOM dedicated interface, you may have to enter an ALOM command to access the actual node consoles. Consult your instructor.
5. If the cluster host systems are not booted, boot them now.

   ```
   ok boot
   ```

6. After all cluster host systems have completed their boot, log in as user `root`.
7. Practice using the Common window Group Term Windows feature under the Options menu. You can ungroup the `cconsole` windows, rearrange them, and then group them together again.

Task 7 – Using the `ccp` Control Panel

The `ccp` control panel can be useful if you must use the console tool variations `crlogin` and `ctelnet`.

Perform the following steps to use the `ccp` control panel:

1. Start the `ccp` tool (**`ccp`** ***`clustername`*** **`&`**).
2. Practice using the `crlogin` and `ctelnet` console tool variations.
3. Quit the `crlogin`, `ctelnet`, and `ccp` tools.

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 3

Preparing for Installation and Understanding Quorum Devices

Objectives

Upon completion of this module, you should be able to:

- List the Sun Cluster software boot disk requirements and hardware restrictions
- Identify typical cluster storage topologies
- Describe quorum Vvotes and quorum devices
- Describe persistent quorum reservations and cluster amnesia
- Describe data fencing
- Configure a supported cluster interconnect system
- Identify public network adapters
- Configure shared physical adapters

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- Is cluster planning required even before installation of the Solaris OS on a node?
- Do certain cluster topologies *enforce* which applications are going to run on which nodes, or do they just *suggest* this relationship?
- Why is a quorum device absolutely required in a two-node cluster?
- What is meant by a cluster *amnesia* problem?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)
- *Sun System Handbook* at `http://sunsolve.sun.com/handbook_pub`

Configuring Cluster Servers

The servers you use in a Sun Cluster 3.1 hardware configuration must conform to a number of general software and hardware requirements to qualify for support. The requirements include both hardware and software in the following areas:

- Boot device restrictions
- Server hardware restrictions

Boot Device Restrictions

With the Sun Cluster 3.1 software release, there are several restrictions on boot devices including the following:

- You cannot use a shared storage device as a boot device. If a storage device is connected to more than one host, it is shared.
- The Solaris 8 OS (Update 7 or greater), Solaris 9 OS, and Solaris 10 OS are supported with the following restrictions:
  - The same version of the Solaris OS, including the update revision (for example, Solaris 9 9/04 or Solaris 10 3/05), must be installed on all nodes in the cluster.
  - VERITAS Dynamic Multipathing (DMP) is *not* supported.

    See Module 6, "Using VERITAS Volume Manager for Volume Management," for more details about this restriction.

    The supported versions of VERITAS Volume Manager *must* have the DMP device drivers enabled, but Sun Cluster 3.1 software still does not support an actual multipathing topology under control of Volume Manager DMP.
- The boot disk partitions have the following requirements:
  - Sun Cluster 3.1 software requires a minimum swap partition of 750 megabytes (Mbytes).
  - There must be a minimum 512 Mbytes per `globaldevices` file system.
  - Solaris Volume Manager software requires a 20 Mbyte partition for its meta state databases.
  - VERITAS Volume Manager software requires two unused partitions for encapsulation of the boot disk.

Note – The /globaldevices file system is modified during the Sun Cluster 3.1 software installation. It is automatically renamed to /global/.devices/node@*nodeid*, where *nodeid* represents the number that is assigned to a node when it becomes a cluster member. The original /globaldevices mount point is removed. The /globaldevices file system must have ample space and ample inode capacity for creating both block special devices and character special devices. This is especially important if a large number of disks are in the cluster. A file system size of 512 Mbytes should suffice for even the largest cluster configurations.

Boot Disk JumpStart™ Software Profile

If you decide to configure your cluster servers using the JumpStart™ software, the following JumpStart software profile represents a starting point for boot disk configuration:

```
install_type initial_install
system_type standalone
partitioning explicit
cluster SUNWCXall
filesys c0t0d0s0 free / logging
filesys c0t0d0s1 1024 swap
filesys c0t0d0s5 512 /globaldevices
filesys c0t0d0s7 20
```

Note – The logging option is the *default* starting in Solaris 9 9/04 (Update 7), and including all updates of Solaris 10 OS.

Server Hardware Restrictions

To be a supported Sun Cluster configuration, the configuration of hardware components in a cluster must first be supported by the corresponding base product groups for each hardware component. For example, for a Sun Cluster configuration composed of two Sun Fire V880 servers connected to two Sun StorEdge 3510 FC Arrays to be supported, the server and storage base product groups must support connecting a Sun StorEdge 3510 FC Array to a Sun Fire V880 server in a standalone configuration.

All cluster servers must meet the following minimum hardware requirements:

- Each server must have a minimum of 512 Mbytes of memory.
- Each server must have a minimum of 750 Mbytes of swap space. If you are running applications on the cluster nodes that require swap space, you should allocate at least 512 Mbytes over that requirement for the cluster.
- Servers in a cluster can be heterogeneous with certain restrictions on mixing certain types of host adapters in the same cluster. For example, Peripheral Component Interconnect (PCI) and Sbus adapters *cannot* be mixed in the same cluster if attached to small computer system interface (SCSI) storage.

Configuring Cluster Storage Connections

While previous versions of the Sun Cluster software had strict rules regarding how many nodes were supported in various disk topologies, the only rules in the Sun Cluster 3.1 software regarding the data storage for the cluster are the following:

- Sun Cluster software never supports more than sixteen nodes. Some storage configurations have restrictions on the total number of nodes supported.
- 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.

Cluster Topologies

Cluster topologies describe typical ways in which cluster nodes can be connected to data storage devices. While 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. The following are some typical topologies:

- Clustered pairs topology
- Pair+N topology
- N+1 topology
- Multiported (more than two node) N*N scalable topology
- NAS device-only topology

Clustered Pairs Topology

As shown in Figure 3-1, a clustered pairs topology is two or more pairs of nodes with each pair physically connected to some storage. Because of the global device and global file system infrastructure, this does not restrict where applications can fail over to and run. Still, it is likely you will configure applications to fail over in pairs of nodes attached to the same storage.

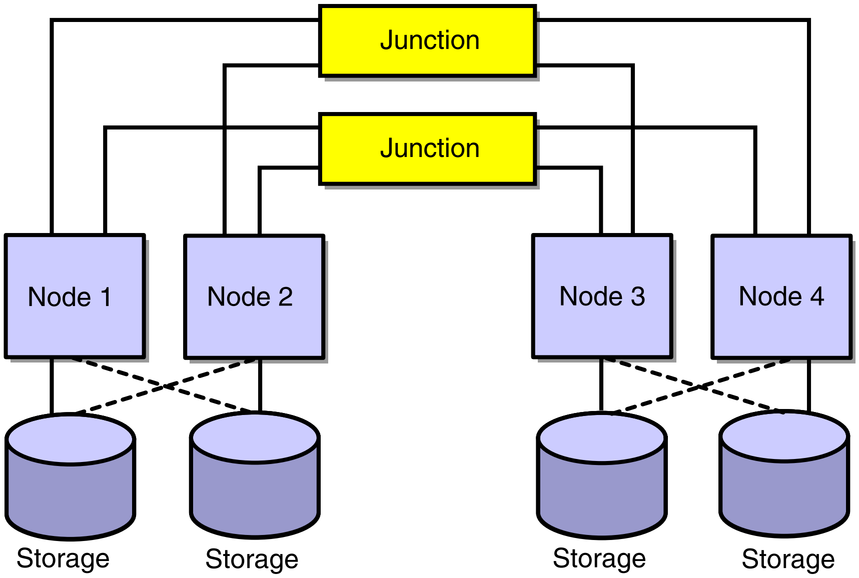


Figure 3-1 Clustered Pairs Topology Configuration

The features of clustered pairs configurations are as follows:

- Nodes are configured in pairs. You can have any even number of nodes from two to 16.
- Each pair of nodes shares storage. Storage is connected to both nodes in the pair.
- All nodes are part of the same cluster.

  You are likely to design applications that run on the same pair of nodes physically connected to the data storage for that application, but you are not restricted to this design.
- Because each pair has its own storage, no one node must have a significantly higher storage capacity than the others.
- The cost of the cluster interconnect is spread across all the nodes.
- This configuration is well suited for failover data services.

Pair+N Topology

As shown in Figure 3-2, the Pair+N topology includes a pair of nodes directly connected to shared storage and nodes that must use the cluster interconnect to access shared storage because they have no direct connection themselves.

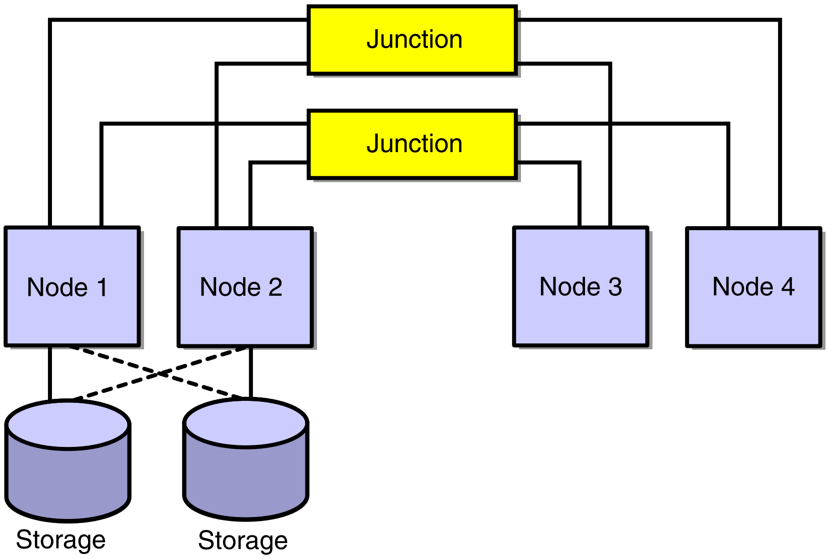


Figure 3-2 Pair+N Topology

The features of the Pair+N configurations are as follows:

- All shared storage is connected to a single pair.
- Additional cluster nodes support scalable data services or failover data services with the global device and file system infrastructure.
- A maximum of sixteen nodes are supported.
- There are common redundant interconnects between all nodes.
- The Pair+N configuration is well suited for scalable data services.

The limitations of a Pair+N configuration is that there can be heavy data traffic on the cluster interconnects. You can increase bandwidth by adding more cluster transports.

N+1 Topology

The N+1 topology, shown in Figure 3-3, enables one system to act as the storage backup for every other system in the cluster. All of the secondary paths to the storage devices are connected to the redundant or secondary system, which can be running a normal workload of its own.

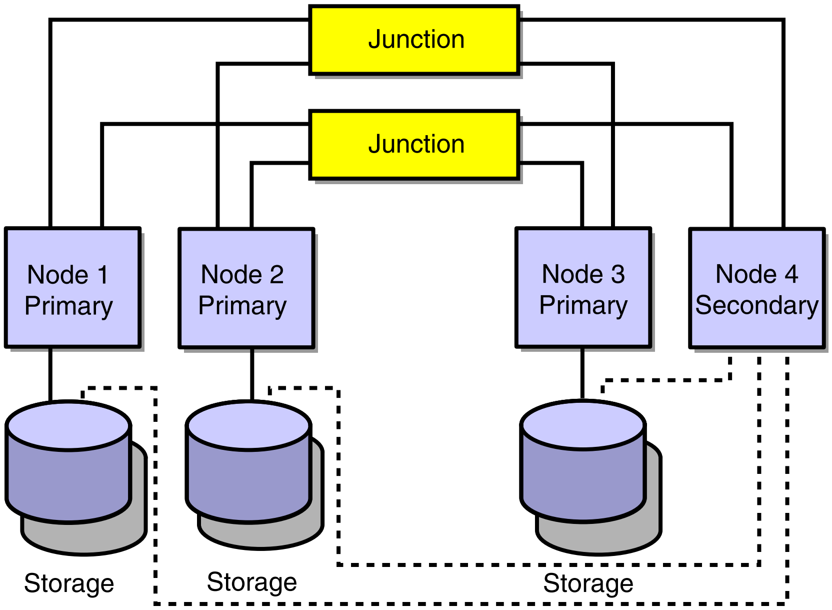


Figure 3-3 N+1 Topology Configuration

The features of the N+1 configurations are as follows:

- The secondary node is the only node in the configuration that is physically connected to all the multihost storage.
- The backup node can take over without any performance degradation.
- The backup node is more cost effective because it does not require additional data storage.
- This configuration is best suited for failover data services.

A limitation of the N+1 configuration is that if there is more than one primary node failure, you can overload the secondary node.

Scalable Storage Topology

In a scalable, or *N*N topology*, shown in Figure 3-4, more than two nodes can be physically connected to the same storage.

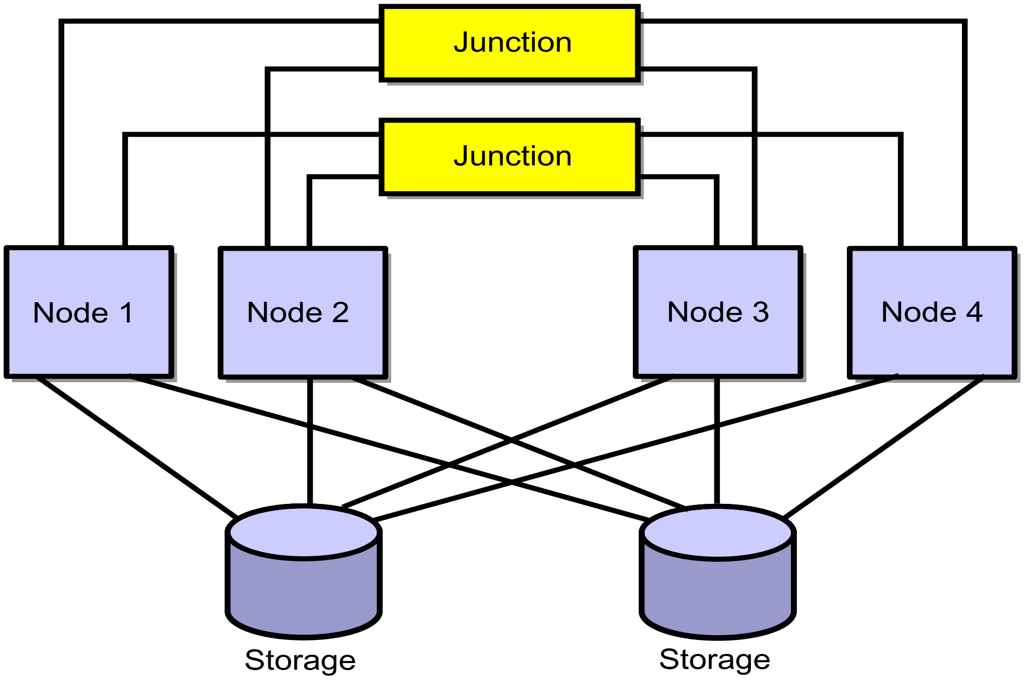


Figure 3-4 Scalable Storage Topology

This configuration is required for running the ORACLE Parallel Server/Real Application Clusters (OPS/RAC) across more than two nodes.

For ordinary, *cluster-unaware* applications, each particular disk group or diskset in the shared storage will still only support physical traffic from one node at a time. Still, having more than two nodes physically connected to the storage adds flexibility and reliability to the cluster

NAS Device-Only Topology

In the NAS Device-Only topology, shown in Figure 3-5, the only cluster storage is a supported Network Attached Storage device. Sun Cluster 3.1 8/05 (Update 4) is the first revision to support this topology, and specifically supports only the Network Appliance (NetApp) Filer NAS product.

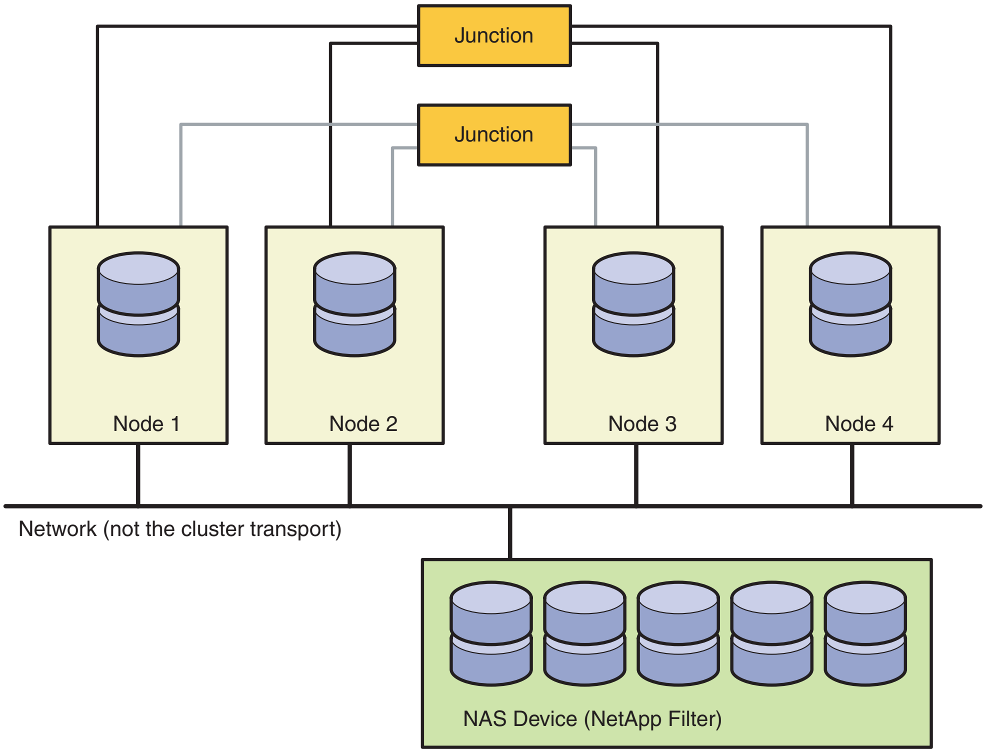

Junction
Junction
Node 1
Node 2
Node 3
Node 4
Network (not the cluster transport)
NAS Device (NetApp Filter)

Figure 3-5 NAS Device-Only Topology

Note – Starting in Sun Cluster 3.1 8/05 there is support for using a NAS device as a *quorum device*, and support for NAS *data fencing*. These concepts are discussed in more detail later in this module.

In order to support the specific NetApp Filer implementation (which is the only NAS device supported in Sun Cluster 3.1 8/05), you must install a package called `NTAPclnas` on the cluster nodes. This package is available for NetApp customers with a support contract from `http://now.netapp.com`.

Non-Storage Topology

In clusters with *more than two nodes*, it is not required to have any shared storage at all. This can be suitable for an application that is purely computer-based and requires no data storage. Two-node clusters *require* shared storage because they require a quorum device.

Single-Node Cluster Topology

In this configuration, one node or domain forms the entire cluster. This configuration allows for a single node to run as a functioning cluster. It offers users the benefits of having application management functionality and application restart functionality. The cluster will start and be fully functional with just one node.

Single node clusters are ideal for users learning how to manage a cluster, to observe cluster behavior (possibly for agent development purposes), or to begin a cluster with the intention of adding nodes to it as time goes on.

Single node clusters can also be useful in the Sun Cluster Geographic Edition product, which manages a *partnership* of two clusters with data replication across a wide area. Each member of such a partnership must be a full Sun Cluster installation, and a one-node cluster on either or both ends is acceptable.

Describing Quorum Votes and Quorum Devices

The cluster membership subsystem of the Sun Cluster 3.1 software framework operates on a "voting system." Following is an explanation of how this voting system operates:

- Each node is assigned exactly one vote.
- Certain disks can be identified as "quorum devices" and are assigned votes.
- There must be a majority (*more* than 50 percent of all possible votes present) to form a cluster or remain in the cluster.

Why Have Quorum Voting at All?

Given the rules present in the bulleted items listed in the previous section, it is clear by looking at a simple two-node cluster why you need extra quorum disk votes. If a two-node cluster had only "node votes," then you need to have both nodes booted to run the cluster. This defeats one of the major goals of the cluster, which is to be able to survive node failure.

But why have quorum voting at all? If there were no quorum rules, you could happily run as many nodes in the cluster as were able to boot at any point in time. However, the quorum vote and quorum devices solve the following two major problems:

- Failure fencing
- Amnesia prevention

These are two distinct problems and it is actually quite clever that they are solved by the same quorum mechanism in the Sun Cluster 3.*x* software. Other vendors' cluster implementations have two distinct mechanisms for solving these problems, making cluster management more complicated.

Failure Fencing

As shown in Figure 3-6, if interconnect communication between nodes ceases, either because of a complete interconnect failure or a node crashing, each node must assume the other is still functional. This is called *split-brain* operation. Two separate clusters cannot be allowed to exist because of the potential for data corruption. Each node tries to establish a cluster by gaining another quorum vote. Both nodes attempt to reserve the designated quorum device. The first node to reserve the quorum disk establishes a majority and remains as a cluster member. The node that fails the race to reserve the quorum device aborts the Sun Cluster software because it does not have a majority of votes.

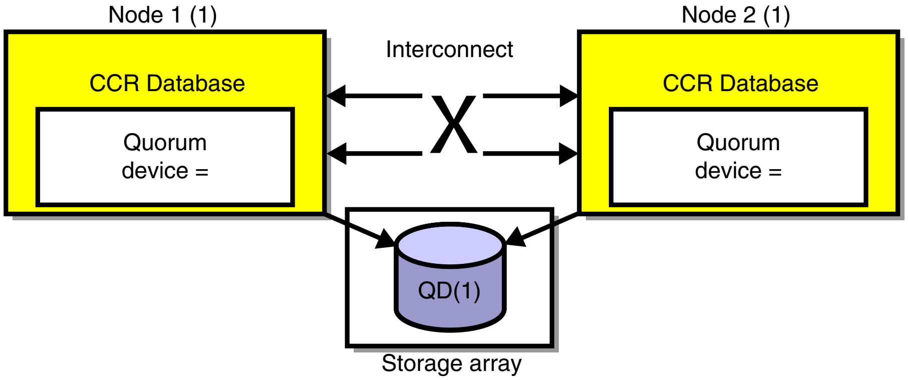


Figure 3-6 Failure Fencing

Amnesia Prevention

A "cluster amnesia" scenario involves one or more nodes being able to form the cluster (boot first in the cluster) with a "stale" copy of the cluster configuration. Imagine the following scenario:

1. In a two node cluster (Node 1 and Node 2), Node 2 is halted for maintenance or crashes.
2. Cluster configuration changes are made on Node 1.
3. Node 1 is shut down.
4. You try to boot Node 2 to form a new cluster.

There is more information about how quorum votes and quorum devices prevent amnesia in "Preventing Cluster Amnesia With Persistent Reservations" on page 3-23.

Quorum Device Rules

The general rules for quorum devices are as follows:

- A quorum device must be available to both nodes in a two-node cluster.
- Quorum device information is maintained globally in the Cluster Configuration Repository (CCR) database.
- A quorum device *should* contain user data.
- The *maximum* and *optimal* number of votes contributed by quorum devices should be the number of node votes minus one (N-1).

  If the number of quorum devices equals or exceeds the number of nodes, the cluster cannot come up if too many quorum devices fail, even if all nodes are available. Clearly, this is unacceptable.
- Quorum devices are not required in clusters with greater than two nodes, but they are recommended for higher cluster availability.
- A single quorum device can be automatically configured by `scinstall`, for a two-node cluster only.
- All other quorum devices are manually configured after the Sun Cluster software installation is complete.
- Quorum devices are configured (specified) using DID devices.

Quorum Mathematics and Consequences

When the cluster is running, it is always aware of the following:

- The total *possible* quorum votes (number of nodes plus the number of disk quorum votes defined in the cluster)
- The total *present* quorum votes (number of nodes booted in the cluster plus the number of disk quorum votes physically accessible by those nodes)
- The total *needed* quorum votes equal greater than 50 percent of the *possible* votes

The consequences are quite simple:

- A node that cannot find the *needed* number of votes at boot time freezes waiting for other nodes to join to up the vote count.
- A node that is booted in the cluster but can no longer find the *needed* number of votes kernel panics.

Two-Node Cluster Quorum Devices

As shown in Figure 3-7, a two-node cluster *requires* a single quorum disk. The total votes are three. With the quorum disk, a single node can start clustered operation with a majority of votes (two votes, in this example).

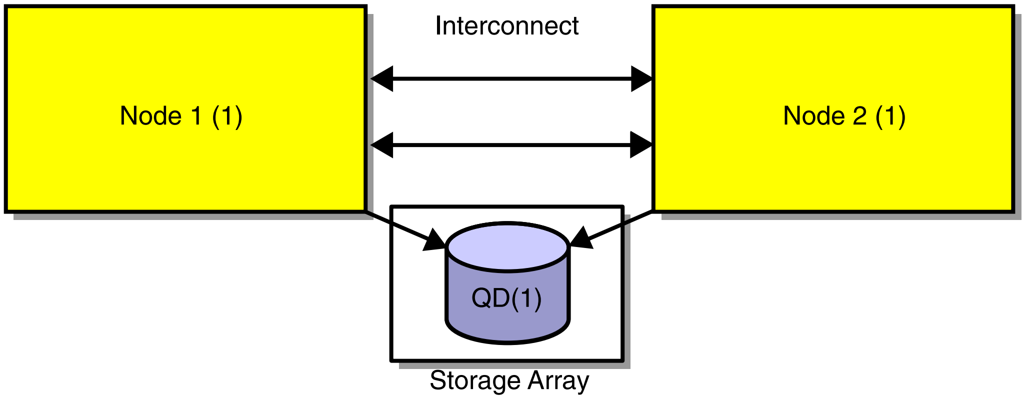


Figure 3-7 Two-Node Cluster Quorum Devices

Clustered-Pair Quorum Disks

A clustered-pairs configuration, shown in Figure 3-8, always has an even number of cluster nodes. The nodes in each pair usually provide data service failover backup for one another.

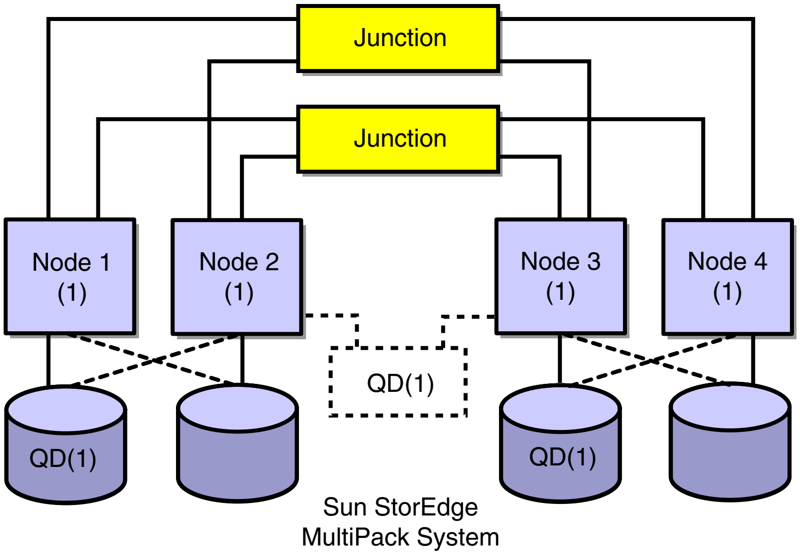


Figure 3-8 Clustered-Pair Quorum Devices

There are many possible split-brain scenarios. Not all of the possible split-brain combinations allow the continuation of clustered operation. The following is true for a clustered pair configuration *without* the "extra" quorum device shown in Figure 3-8.

- There are six possible votes.
- A quorum is four votes.
- If both quorum devices fail, the cluster can still come up.

  The nodes wait until all are present (booted).
- If Nodes 1 and 2 fail, there are not enough votes for Nodes 3 and 4 to continue running.

  A token quorum device between Nodes 2 and 3 can eliminate this problem. A Sun StorEdge MultiPack desktop array could be used for this purpose.
- An entire pair of nodes can fail, and there are still four votes out of seven.

Pair+N Quorum Disks

Figure 3-9 shows a typical quorum disk configuration in a Pair+2 configuration. Three quorum disks are used.

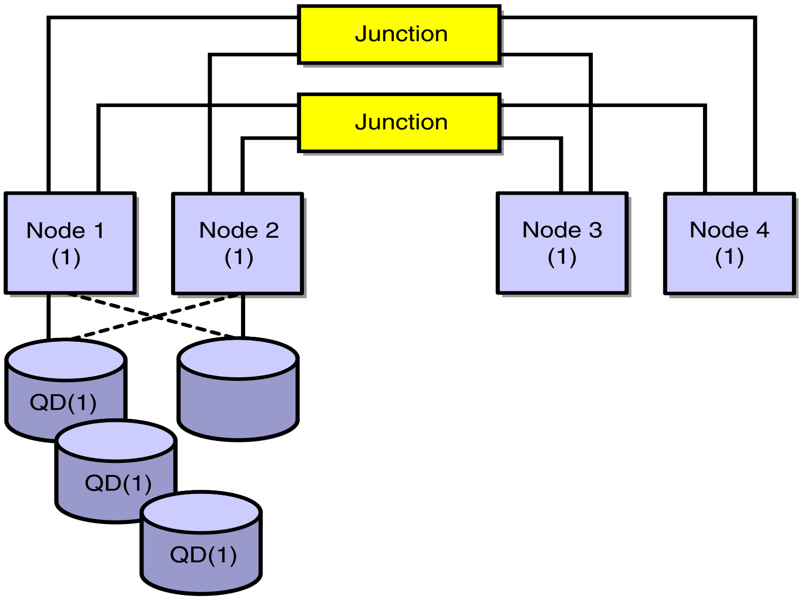


Figure 3-9 Pair+N Quorum Devices

The following is true for the Pair+N configuration shown in Figure 3-9:

- There are three quorum disks.
- There are seven possible votes.
- A quorum is four votes.
- Nodes 3 and 4 do not have access to any quorum devices.
- Nodes 1 or 2 can start clustered operation by themselves.
- Up to three nodes can fail (Nodes 1, 3, and 4 or Nodes 2, 3, and 4), and clustered operation can continue.

N+1 Quorum Disks

The N+1 configuration shown in Figure 3-10 requires a different approach. Node 3 is the failover backup for both Node 1 and Node 2.

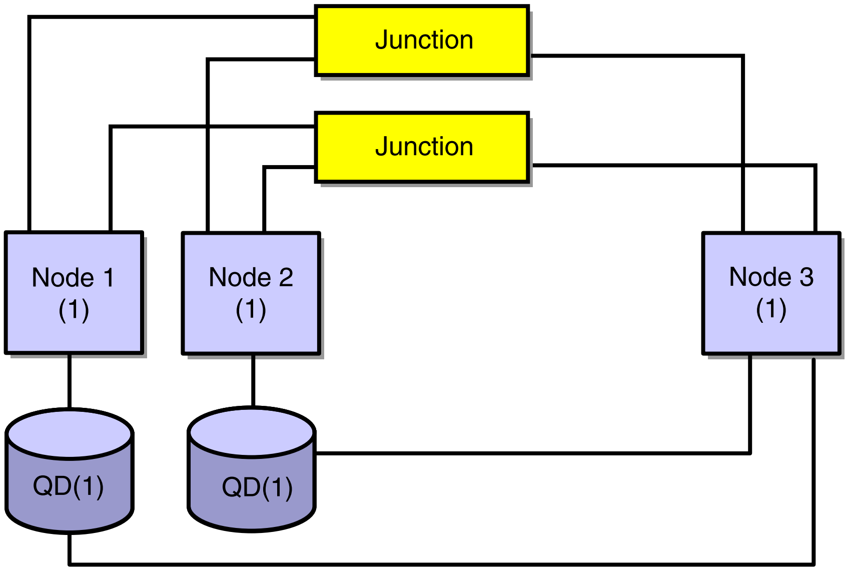


Figure 3-10 N+1 Quorum Devices

The following is true for the N+1 configuration shown in Figure 3-10:

- There are five possible votes.
- A quorum is three votes.
- If Nodes 1 and 2 fail, Node 3 can continue.

Quorum Devices in the Scalable Storage Topology

Quorum Devices in the scalable storage topology, as shown in Figure 3-11, differ significantly from any other topology.

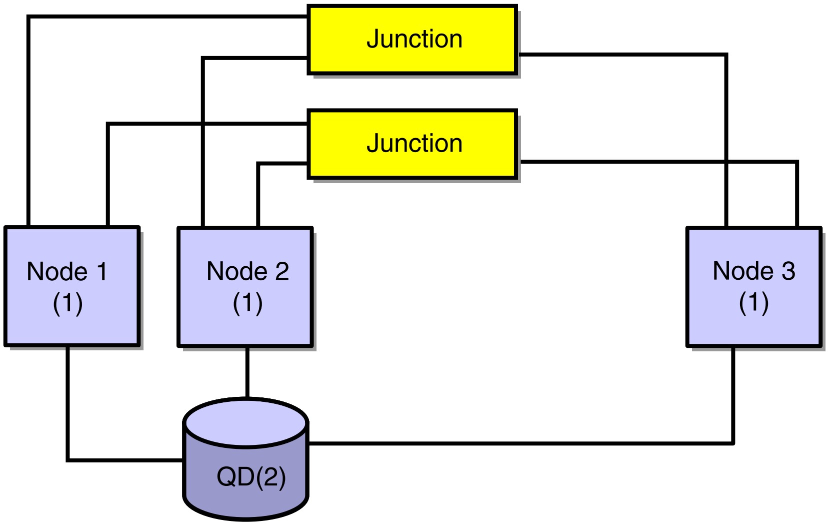


Figure 3-11 Quorum Devices in the Scalable Storage Topology

The following is true for the quorum devices shown in Figure 3-11:

- The single quorum device has a vote count equal to the votes of the nodes directly attached to it minus one.

Note – This rule is universal. In all the *previous* examples, there were two nodes (with one vote each) directly to the quorum device, so that the quorum device had one vote.

- The mathematics and consequences still apply.
- The reservation is done using a SCSI-3 *Persistent Group Reservation* (see “SCSI-3 Persistent Group Reservation” on page 3-26).

  If, for example, Nodes 1 and 3 can intercommunicate but Node 2 is isolated, Node 1 or 3 can reserve the quorum device on behalf of both of them.

Note – It would seem that in the same race, Node 2 could “win” and eliminate both Nodes 2 and 3. “Intentional Reservation Delays for Partitions With Fewer Than Half of the Nodes” on page 3-29 shows why this is unlikely.

Using a NAS Device as a Quorum Device

You can use a NAS device as a quorum device. In a two-node cluster, it can be your single-vote, SCSI-2 quorum device. In a cluster with more than two nodes, it will be configured as a SCSI-3 quorum device and given a number of votes that is one fewer than the number of nodes in the cluster. This architecture is illustrated in Figure 3-12.

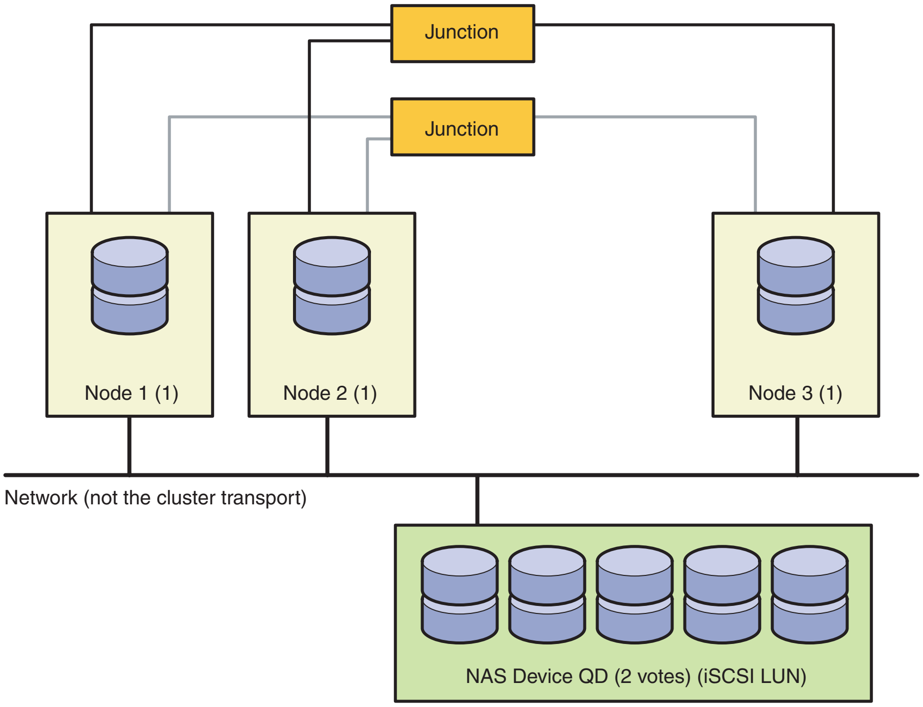

Junction
Junction
Node 1 (1)
Node 2 (1)
Node 3 (1)
Network (not the cluster transport)
NAS Device QD (2 votes) (iSCSI LUN)

Figure 3-12 NAS Device Used as a Quorum Device

NAS Quorum Device Implementation

The NAS quorum architecture, like the rest of the NAS architecture, is a general architecture with specific support in Sun Cluster 3.1 8/05 software only for the NetApp Filer.

The specific implementation for quorum on the NetApp filer requires use of the Internet SCSI (iSCSI) protocol, which is an implementation of SCSI over TCP/IP.

Note – The iSCSI protocol is *not* used by Sun Cluster nodes to actually access the data on a NetApp Filer NAS device. It is used only to implement the correct behavior for using the NAS device as a quorum device.

On the NetApp Filer side, the requirements for operation as a Sun Cluster quorum device are as follows:

- You must install the iSCSI license from your NAS device vendor.
- You must configure an iSCSI Logical Unit (LUN) for use as the quorum device.
- When booting the cluster, you must always boot the NAS device *before* you boot the cluster nodes.

On the cluster side, the requirements and restrictions are as follows:

- The iSCSI functionality is built into the `NTAPclnas` package that must be installed on each cluster node.
- A cluster can use a NAS device for only a single quorum device. There should be no need for other quorum devices. This is true because, in a cluster of more than two nodes, the quorum acts like a SCSI-3 quorum device attached to all the nodes.
- Multiple clusters using the same NAS device *can* use separate iSCSI LUN's on that device as their quorum devices.

Preventing Cluster Amnesia With Persistent Reservations

Quorum devices in the Sun Cluster software environment are used not only as a means of failure fencing but also as a means to prevent *cluster amnesia*.

Earlier you reviewed the following scenario:

1. In a two-node cluster (Node 1 and Node 2), Node 2 is halted for maintenance.
2. Meanwhile Node 1, running fine in the cluster, makes all sorts of cluster configuration changes (new device groups, resource groups).
3. Now Node 1 is shut down.
4. You try to boot Node 2 to form a new cluster.

In this simple scenario, the problem is that you boot Node 2 at the end. It does not have the "correct" copy of the CCR. But, if it were allowed to boot, Node 2 would have to use the copy that it has (as there is no other copy available) and you would "lose" the changes to the cluster configuration made in Step 2.

The Sun Cluster software quorum involves *persistent reservations* that prevent Node 2 from booting into the cluster. It is not able to count the quorum device as a vote. Node 2 will, therefore, wait until the other node boots to achieve the correct number of quorum votes.

Persistent Reservations and Reservation Keys

A *persistent reservation* means that reservation information on a quorum device:

- Survives even if all nodes connected to the device are reset
- Survives even after the quorum device itself is powered on and off

Clearly this involves writing some type of information on the disk itself. The information is called a reservation key and is as follows:

- Each node is assigned a unique 64-bit reservation key value.
- Every node that is physically connected to a quorum device has its reservation key physically written onto the device.

Figure 3-13 and Figure 3-14 on page 3-25 consider two nodes booted into a cluster connected to a quorum device:

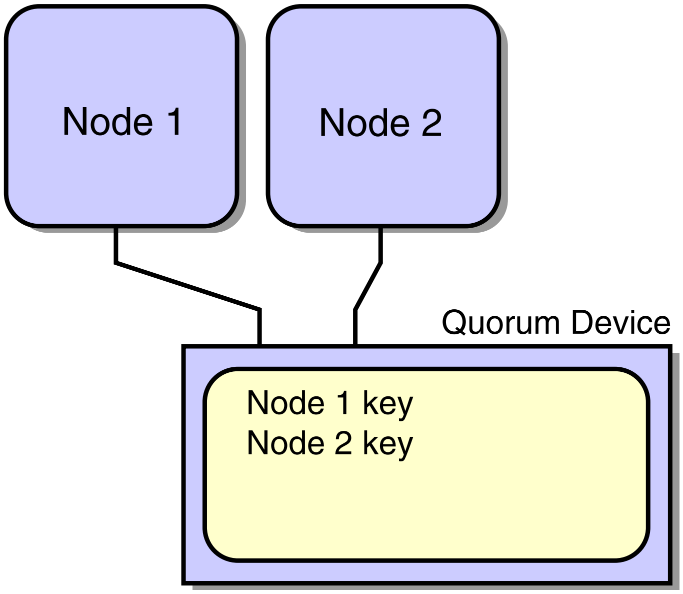


Figure 3-13 Two Nodes Booted and Connected to a Quorum Device

If Node 2 leaves the cluster for any reason, Node 1 will remove or *preempt* Node 2's key off of the quorum device. This would include if there were a split brain and Node 1 won the race to the quorum device.

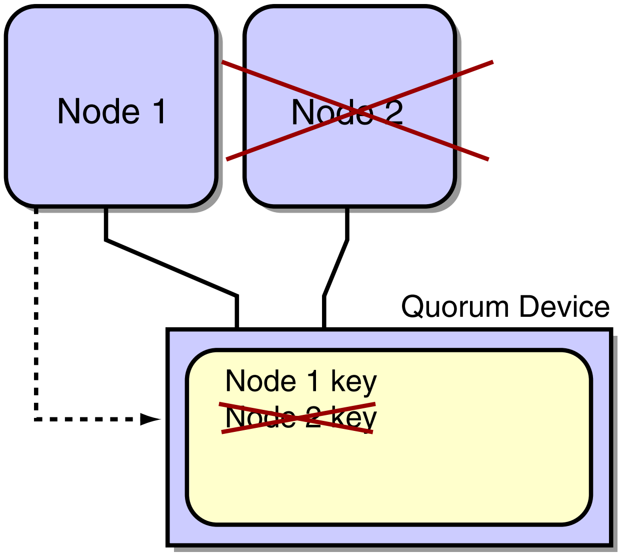


Figure 3-14 Node 2 Leaves the Cluster

Now the rest of the equation is clear. If you are booting into the cluster, a node cannot count the quorum device vote unless its reservation key is *already* on the quorum device. Thus, in the scenario illustrated in the previous paragraph, if Node 2 tries to boot first into the cluster, it will not be able to count the quorum vote, and must wait for Node 1 to boot.

After Node 1 joins the cluster, it can detect Node 2 across the transport and add Node 2's reservation key back to the quorum device so that everything is equal again.

A reservation key only gets added back to a quorum device by another node in the cluster whose key is already there.

SCSI-2 Persistent Reservation Emulation

The scenario presented in the previous section was actually using a method called SCSI-2 Persistent Reservation Emulation (PRE) to implement the reservation keys. PREs have the following characteristics:

- The persistent reservations are *not* supported directly by the SCSI command set.
- Reservation keys are written on *private cylinders* of the disk (cylinders that are not visible in the `format` command, but are still directly writable by the Solaris OS).

  The reservation keys have no impact on using the disk as a regular data disk, where you will not see the private cylinders.
- The *race* (for example, in a split brain scenario) is still decided by a normal SCSI-2 disk reservation.

When you add a quorum device, the Sun Cluster software automatically will use SCSI-2 PRE with that quorum device if there are exactly two sub-paths for the DID associated with that device.

SCSI-3 Persistent Group Reservation

When there are more than two nodes physically connected to the disk, SCSI-3 Persistent Group Reservations (PGR) are used. PGR have the following characteristics:

- The persistent reservations *are* implemented directly by the SCSI-3 command set. Disk firmware itself must be fully SCSI-3 compliant.
- "Racing" to remove another node's reservation key is not a separate step from physical reservation of the disk, as it is in SCSI-2. With SCSI-3, the removal of the other node's key *is* the reservation.

SCSI-3 PGR Scenario

In Figure 3-15, four nodes are all physically connected to a quorum drive. Remember the single quorum drive will have three quorum votes.

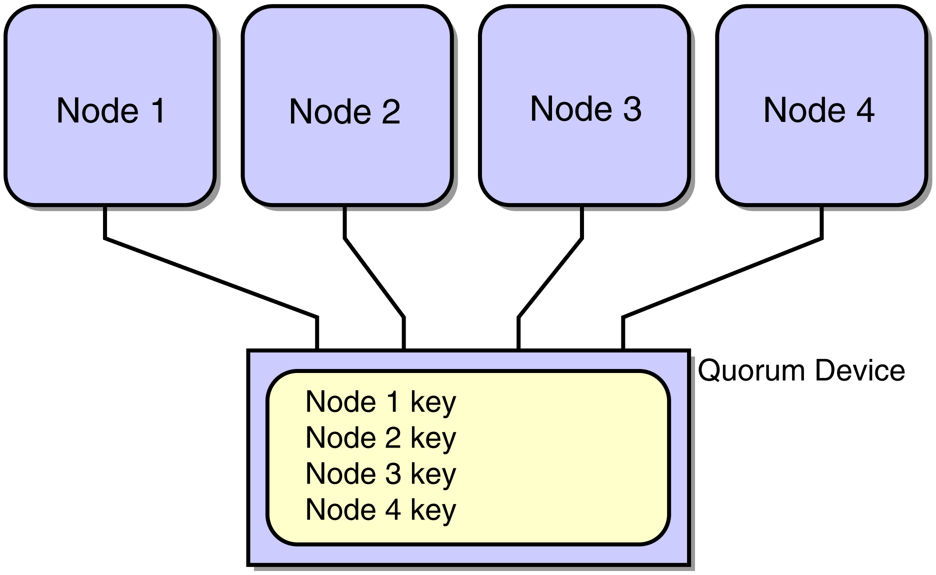


Figure 3-15 Four Nodes Physically Connected to a Quorum Drive

Now imagine because of multiple transport failures there is a partitioning where Nodes 1 and 3 can see each other over the transport and Nodes 2 and 4 can see each other over the transport.

In each pair, the node with the lower reservation key will try to eliminate the reservation key of the other pair. The SCSI-3 protocol assures that only one pair will win. This is shown in Figure 3-16.

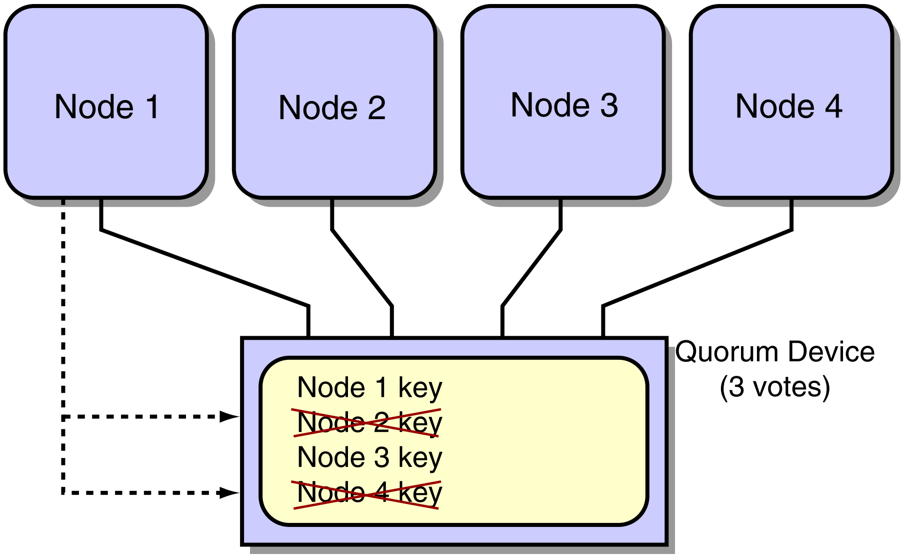


Figure 3-16 One Pair of Nodes Eliminating the Other

Because Nodes 2 and 4 have their reservation key eliminated, they cannot count the three votes of the quorum device. Because they fall below the needed quorum, they will kernel panic.

Cluster amnesia is avoided in the exact same way as in a two-node quorum device. If you now shut down the whole cluster, Node 2 and Node 4 can not count the quorum device because their reservation key is eliminated. They would have to wait for either Node 1 or Node 3 to join. One of those nodes can then add back reservation keys for Node 2 and Node 4.

Intentional Reservation Delays for Partitions With Fewer Than Half of the Nodes

Imagine the same scenario just presented, but three nodes can talk to each other while the fourth is "isolated" on the cluster transport, as shown in Figure 3-17.

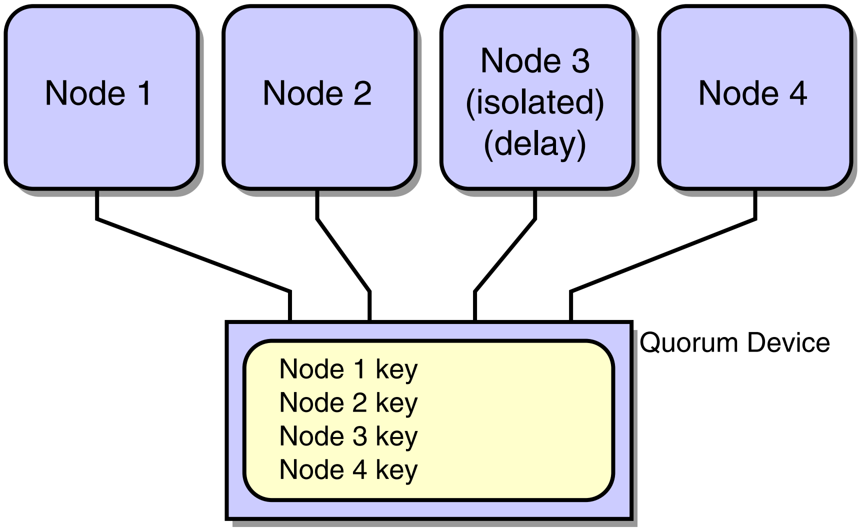


Figure 3-17 Node 3 Is Isolated on the Cluster Transport

Is there anything to prevent the lone node from eliminating the cluster keys of the other three and making them all kernel panic?

In this configuration, the lone node will intentionally delay before racing for the quorum device. The only way it can "win" is if the other three nodes are really dead (or each of them is also isolated, which would make them all delay the same amount).

The delay is implemented when the number of nodes that a node can see on the transport (including itself) is fewer than half the total nodes.

An example demonstration of the PGR process is located in "Task 4 – Preventing Cluster Amnesia" in Module 5, "Performing Basic Cluster Administration" on page 5-29.

Data Fencing

As an extra precaution, nodes that are eliminated from the cluster because of quorum problems also lose access to *all* shared data devices.

The reason for this is to eliminate a potential timing problem. The node or nodes that *remain* in the cluster have no idea of knowing whether the nodes being eliminated from the cluster are actually still running or not. If they are running, they *will* have a kernel panic (once they recognize they have fallen beneath the required quorum votes). However the surviving node or nodes have no way to "wait" for the other nodes to kernel panic before taking over the data. The whole reason nodes are being eliminated is that there has been a communication failure with them.

In order to eliminate this potential timing problem, which otherwise could lead to data corruption, before a surviving node or nodes reconfigures the data and applications, it "fences" the eliminated node or nodes from all shared data devices, in the following manner:

- For all shared dual-ported devices with an eliminated node, the surviving node does a standard SCSI-2 reservation on the device.
- For all shared multiported (more than two nodes physically attached) devices with an eliminated node or nodes, the surviving nodes do a standard SCSI-3 persistent group reservation on the device.
- For NetApp Filer NAS devices, a surviving node informs the NAS device to eliminate the NFS share from the eliminated node or nodes.

Data fencing is released when a fenced node is able to boot successfully into the cluster again.

Note – This data fencing is the reason you *absolutely can not put any boot device in a shared storage array*. If you did so, a node that is eliminated from the cluster would be fenced off from its own boot device.

Configuring a Cluster Interconnect

There are two variations of cluster interconnects: *point-to-point* and *junction-based*. In a junction-based interconnect, the junctions must be switches and not hubs.

Point-to-Point Cluster Interconnect

In a two-node cluster, you can directly connect interconnect interfaces using crossover cables. Figure 3-18 shows a point-to-point interconnect configuration using 100BASE-T interfaces.

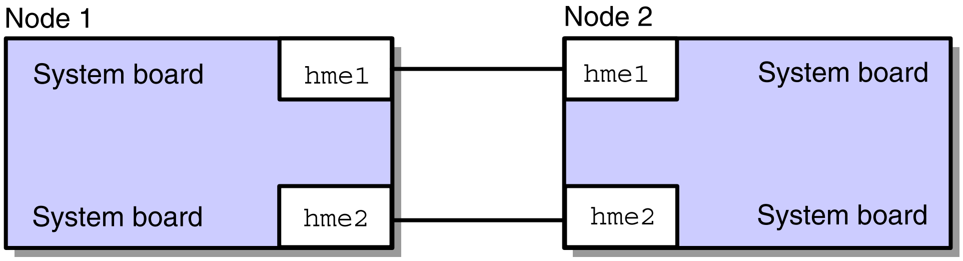


Figure 3-18 Point-to-Point Cluster Interconnect

During the Sun Cluster 3.1 software installation, you must provide the names of the end-point interfaces for each cable.

Caution – If you provide the wrong interconnect interface names during the initial Sun Cluster software installation, the first node is installed without errors, but when you try to manually install the second node, the installation hangs. You have to correct the cluster configuration error on the first node and then restart the installation on the second node.

Junction-Based Cluster Interconnect

In cluster configurations greater than two nodes, you must join the interconnect interfaces using switches. You can also use switches to join two-node cluster interconnects to prepare for the expansion of the number of nodes at a later time. A typical junction-based interconnect is shown in Figure 3-19.

During the Sun Cluster 3.1 software installation, you are asked whether the interconnect system uses junctions. If you answer yes, you must provide names for each of the switches.

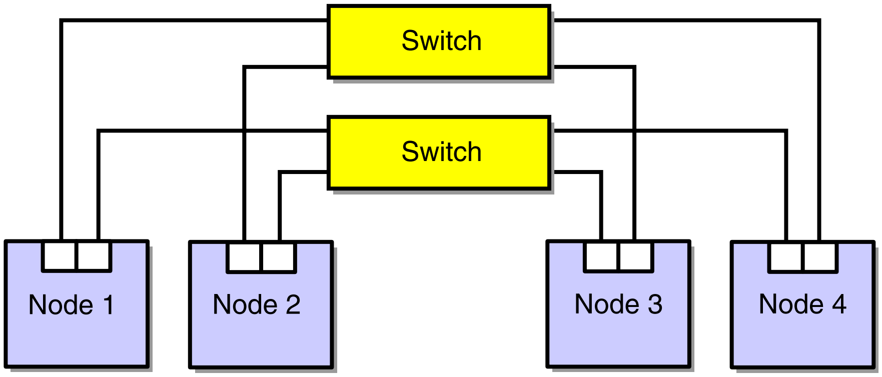


Figure 3-19 Junction-Based Cluster Interconnect

Note – If you specify more than two nodes during the initial portion of the Sun Cluster software installation, the use of junctions is assumed.

Cluster Transport Interface Addresses

During the Sun Cluster software installation, the cluster interconnects are assigned IP addresses based on a base address of `172.16.0.0`. If necessary, you can override the default address, but this is not recommended. Uniform addresses can be a benefit during problem isolation.

You are *required* to reserve at least a class B-like namespace for the cluster interconnect IP addresses. This is documented in the *Sun™ Cluster 3.1 System Administration Guide* but not well presented or enforced by the installation script.

Identify Cluster Transport Interfaces

Identifying network interfaces is not an easy task. To accurately determine the logical name of each interface on a system, use the following steps:

1. Look for network interfaces in the `/etc/path_to_inst` file. Look for entries that have possible types of interfaces in double quotes.

   The following example shows a machine with a single `hme` interface and four `qfe` ports (a single card). The instance names show up just to the *left* of the interface type, near the end of the line:

   ```
   # egrep '"qfe"|"hme"|"ce"|"ge"|"bge"|"eri"' \
   /etc/path_to_inst
   "/pci@1f,4000/pci@2/SUNW,qfe@0,1" 0 "qfe"
   "/pci@1f,4000/pci@2/SUNW,qfe@1,1" 1 "qfe"
   "/pci@1f,4000/pci@2/SUNW,qfe@2,1" 2 "qfe"
   "/pci@1f,4000/pci@2/SUNW,qfe@3,1" 3 "qfe"
   "/pci@1f,4000/network@1,1" 0 "hme"
   ```

2. Verify which interfaces are already up, with a public network address. These are *not* candidates for a cluster transport interface.

```
    # ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu
8232 index 1 inet 127.0.0.1 netmask ff000000
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4>
mtu 1500 index 2 inet 129.200.9.2 netmask ffffff00
broadcast 129.200.9.255
ether 8:0:20:96:3:86
```

3. Bring up a candidate interface for testing. Make up some unused subnet address just to test out interconnectivity across a private network.

   ```
   # ifconfig qfe1 plumb
   # ifconfig qfe1 192.168.1.1 up
   ```

4. Perform Step 3 on the other node. Choose a corresponding IP address, for example **`192.168.1.2.`**

5. Test that the nodes can ping across each private network, for example:

   ```
   # ping 192.168.1.2
   192.168.1.2 is alive
   ```

6. Make sure that the interface you are looking for is *not* actually on the public net:

   a. In one window run the `snoop` command for the interface in question:

   ```
   # snoop -d qfe1
   hope to see no output here
   ```

   b. In another window, ping the public broadcast address, and make sure no traffic is seen by the "candidate" interface.

   ```
   # ping public_net_broadcast_address
   ```

Note – You should create configuration worksheets and drawings of your cluster configuration, as in lab exercises at the end of this module.

7. After you have identified the new network interfaces, bring them down again. Cluster installation fails if your transport network interfaces are still up from testing.

   ```
   # ifconfig qfe1 down unplumb
   ```

8. Repeat steps 3-7 with transport adapter candidates for the *second* cluster transport. Repeat again if you will be configuring more than two cluster transports.

Identifying Public Network Adapters

You will not be asked about public network configuration at the time you are installing the cluster.

The public network interfaces must be managed by IPMP, which can be administered either before or after cluster installation. In this course, you will do it after cluster installation in Module 8.

Because you are identifying your private transport interfaces before cluster installation, it can be useful to identify your public network interfaces at the same time, so as to avoid confusion.

Your primary public network adapter should be the only one currently configured on the public network. You can verify this with the following command:

```
# ls -l /etc/hostname.*
# ifconfig -a
```

You can verify your secondary public network adapter, if applicable, by:

- Making sure it is not one of those you identified to be used as the private transport
- Making sure it *can* snoop public network broadcast traffic

```
# ifconfig ifname plumb
# snoop -d ifname
(other window or node)# ping -s pubnet_broadcast_addr
```

Configuring Shared Physical Adapters

Recall that certain adapters are capable of participating in tagged VLANs, and can be used as both private and public network adapters assuming that the switches are also capable of tagged VLANs. This allows blade architecture servers that have only two physical network adapters to be clustered.

Note – At the time of writing of this course only the Broadcom Gigabit Ethernet (bge) adapters and Cassini Ethernet (ce0) adapters are capable of serving as shared physical adapters.

An adapter that is participating in a tagged VLAN configuration is assigned an instance number `1000*(Vlan_identifer) + physical_instance_number`.

For example, if you have a physical adapter ce1, and it is participating in a tagged VLAN with ID 3 as its "public network personality", and a tagged VLAN with ID 5 as its "private network personality", then it will appear as if it were two separate adapters `ce3001` and `ce5001`.

Configuring the Public Network

In order to configure a shared adapter's public network personality, all you have to do is configure the adapter instance according to the mathematical formula above. Following the example, since VLAN ID3 is going to be used for the public network identity of what would otherwise be `ce1`, you would just configure the adapter instance `ce3001` by creating a file `/etc/hostname.ce3001`. When instance `ce3001` is plumbed, the adapter driver understands that it is using tagged VLAN ID 3 on physical instance number 1.

Allocating a Different VLAN ID for the Private Network

You will never configure the private network ID manually. The initial configuration will be done using the `scinstall` utility, as we will see in Module 4. All you have to do is ensure you have a different VLAN ID for the public and private networks. The `scinstall` utility will automatically detect a tagged VLAN-capable adapter and query for the private VLAN ID.

Exercise: Preparing for Installation

In this exercise, you complete the following tasks:

- Task 1 – Verifying the Solaris OS
- Task 2 – Identifying a Cluster Topology
- Task 3 – Selecting Quorum Devices
- Task 4 – Verifying the Cluster Interconnect Configuration
- Task 5 – Selecting Public Network Interfaces

Preparation

To begin this exercise, you must be connected to the cluster hosts through the `cconsole` tool, and you are logged into them as user `root`.

Note – During this exercise, when you see italicized variable names, such as ***IPaddress***, ***enclosure_name***, ***node1***, or ***clustername*** embedded in a command string, substitute the names appropriate for your cluster.

Task 1 – Verifying the Solaris OS

In this section, you verify that the boot disk is correctly partitioned on all nodes.

Perform the following steps:

1. Type the **/etc/prtconf** command on each node and record the size of physical memory (**/etc/prtconf | grep Memory**).

   Node 1 memory: _______________

   Node 2 memory: _______________

2. Type the **df -kl** command on each node, and verify that there is a `globaldevices` file system mounted.The recommended size is 512 Mbytes, which is sufficient for even huge numbers of devices.

3. Type **swap -l** on each node and verify it has at least 750 Mbytes of swap space.

Task 2 – Identifying a Cluster Topology

Perform the following steps:

1. Record the desired topology configuration of your cluster in Table 3-1.

Table 3-1 Topology Configuration

Number of nodes	
Number of storage arrays	
Types of storage arrays	

2. Verify that the storage arrays in your cluster are properly connected for your target topology. Recable the storage arrays if necessary.

Task 3 – Selecting Quorum Devices

Perform the following steps:

1. Record the number of quorum devices you must configure after the cluster host software installation.

   Number of quorum devices: __________

Note – Consult with your instructor if you are not sure about your quorum device configuration.

2. Decide whether the `scinstall` utility will automatically be able to choose a quorum device (it can for a single quorum device for a two-node cluster).

   Automatic Quorum Configuration: (yes/no)?

3. If there can be no automatic quorum configuration (for a three-node cluster, for example), type the **`format`** command and record the logical path to the disks that you want to use as quorum disk drives in your storage arrays.

   Quorum disks: ________________________________

   Type **`Control-D`** to cleanly exit the `format` utility.

Task 4 – Verifying the Cluster Interconnect Configuration

This task describes how to verify the cluster interconnect configuration.

Skip this section if your cluster interconnect is not point-to-point.

Perform the following steps to configure a point-to-point Ethernet interconnect:

1. Determine the names of your cluster interconnect adapters.

Note – You can use the strategy presented on page 3-33, if you are remote from the cluster equipment, you want to pretend you are remote, or your instructor does not want to tell you so that you gain experience doing it yourself.

2. Complete the form in Figure 3-20 if your cluster uses an Ethernet-based point-to-point interconnect configuration.

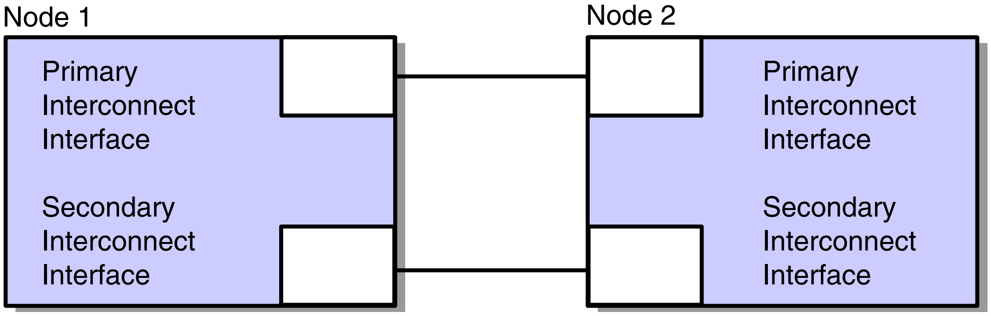

Node 1
Primary Interconnect Interface
Secondary Interconnect Interface
Node 2
Primary Interconnect Interface
Secondary Interconnect Interface

Figure 3-20 Ethernet Interconnect Point-to-Point Form

Perform the following steps to configure a switch-based Ethernet interconnect:

3. Complete the form in Figure 3-21 if your cluster uses an Ethernet-based cluster interconnect with switches.
   a. Record the logical names of the cluster interconnect interfaces (`hme2`, `qfe1`, and so forth).

Note – You can use the strategy presented on page 3-33, if you are remote from the cluster equipment, you want to pretend you are remote, or your instructor does not want to tell you so that you gain experience doing it yourself.

b. Add or delete nodes to the diagram as appropriate.

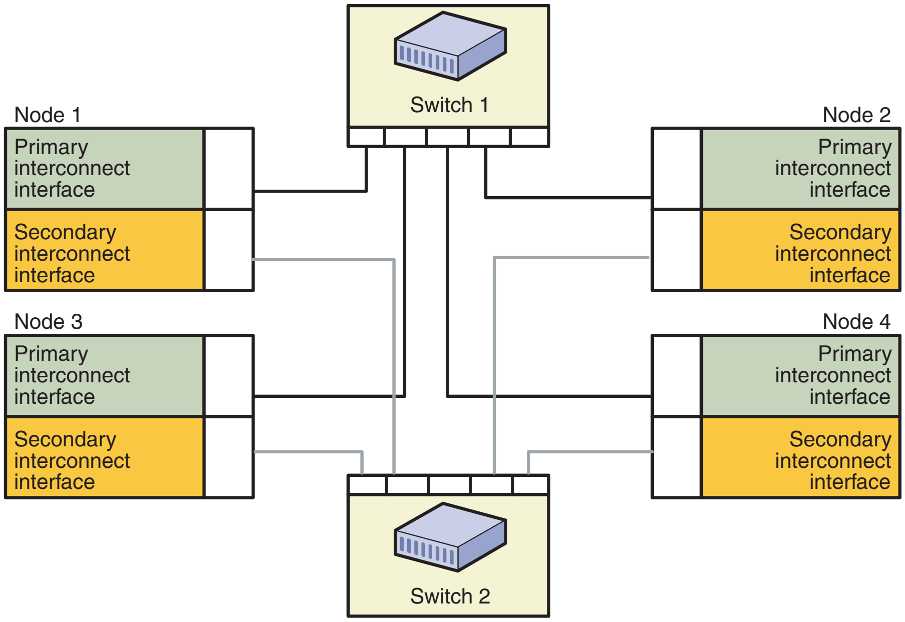

Switch 1
Node 1
Primary interconnect interface
Secondary interconnect interface
Node 2
Primary interconnect interface
Secondary interconnect interface
Node 3
Primary interconnect interface
Secondary interconnect interface
Node 4
Primary interconnect interface
Secondary interconnect interface
Switch 2

Figure 3-21 Ethernet Interconnect With Switches Form

4. Verify that each Ethernet interconnect interface is connected to the correct switch. If you are remote, and you do not want to take your instructor's word for it, you can verify that all nodes can ping each other across the private switches by using the strategy presented on page 3-33.

Note – If you have any doubt about the interconnect cabling, consult with your instructor now. Do not continue this exercise until you are confident that your cluster interconnect system is cabled correctly, and that you know the names of the cluster transport adapters.

Task 5 – Selecting Public Network Interfaces

Ask for help from your instructor in identifying public network interfaces on each node that can be used in IPMP groups.

Perform the following steps to select public network interfaces:

1. Record the logical names of potential IPMP Ethernet interfaces on each node in Table 3-2.

Note – You can use the strategy presented on page 3-35, if you are remote from the cluster equipment, you want to pretend you are remote, or your instructor does not want to tell you so that you gain experience doing it yourself.

Table 3-2 Logical Names of Potential IPMP Ethernet Interfaces

System	**Primary IPMP interface**	**Backup IPMP interface**
Node 1		
Node 2		
Node 3 (if any)		
Node 4 (if any)		

Note – It is important that you are sure about the logical name of each public network interface (`hme2`, `qfe3`, and so on).

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 4

Installing and Configuring the Sun Cluster Software Framework

Objectives

Upon completion of this module, you should be able to:

- Understand the Sun Cluster installation steps and configuration steps
- Install the Sun Cluster packages using the Sun Java Enterprise System (Java ES) installer
- Describe the Sun Cluster Framework configuration
- Configure a cluster installation using all-at-once and typical modes
- Configure a cluster installation using one-at-a-time and custom modes
- Configure additional nodes for the one-at-a-time method
- Describe the Solaris OS files and settings that are automatically configured by `scinstall`
- Perform automatic quorum configuration
- Describe the manual quorum selection
- Perform post-installation configuration

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- What configuration issues might control how the Sun Cluster software is installed?
- What type of post-installation tasks might be necessary?
- What other software might you need to finish the installation?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)

Sun Cluster Software Installation and Configuration

There are two distinct steps to follow in order to successfully initialize the Sun Cluster 3.1 framework and boot into the cluster:

1. Install the Sun Cluster 3.1 framework packages.
2. Configure the Sun Cluster 3.1 Software with `scinstall`.

Beginning with Sun Cluster 3.1 8/05 (Update 4), it is *required* that these be done as separate steps, unless you are upgrading the Sun Cluster framework. Only the upgrade procedure supports using `scinstall` as the method of installing the packages.

In earlier revisions of the Sun Cluster software, you may do the installation and configuration separately, or they can be combined using the `scinstall` utility.

Introduction to Sun Cluster Package Installation

Starting with Sun Cluster 3.1 8/05 an initial (non-upgrade) installation of the Sun Cluster packages is done one of the following two ways:

- Using the Java Enterprise System (Java ES) installer utility that is always co-packaged with the Sun Cluster framework
- Using a flash image that was created on a system where the Sun Cluster packages have been installed using the Java ES installer. The flash image must have been created *before* the cluster was actually configured with `scinstall`.

Sun Cluster Packaging

Starting with Sun Cluster 3.1 8/05, the distribution of Sun Cluster will look identical regardless of whether the Sun Cluster software is bundled with the full Java ES distribution (including the Sun Java System Web Server, Application Server, Directory Server, and other products) or as a standalone release. There are *separate* distributions for SPARC and x86. The distribution looks like the following:

- CDROM 1 of 2

  This CDROM contains the `installer` utility and some of the prerequisite components for the Sun Cluster software.

In a full Java ES distribution, the other Java System products are also included on CDROM 1 of 2.

- CDROM 2 of 2

  This CDROM contains the Sun Cluster framework product. It also contains the data service agents specifically only for the Java ES applications.

- Sun Cluster Agents CDROM

  A *separate* CDROM contains the Sun Cluster data service agents for all other applications that are *not* Sun Java System Applications.

Spooling Together CDROM 1of 2 and CDROM 2 of 2

If you intend to install the Sun Cluster framework (or a Java ES entire release containing the Sun Cluster framework) from a Network File System (NFS) server you must spool together the contents of the CDROM 1 of 2 and CDROM 2 of 2 into the same arena, as in the following example:

```
# mkdir /jes_sc31u4
(insert CDROM 1 of 2)
# cd /cdrom/cdrom0
# find . -print|cpio -pdmuv /jes_sc31u4
# cd /

(eject CDROM 1 of 2)
(insert CDROM 2 of 2)

# cd /cdrom/cdrom0
# find . -print|cpio -pdmuv /jes_sc31u4
# cd /
```

This looks slightly unnatural, but it is the only way to end up with a correct spooled version of the software.

Note – Do *not* spool the Sun Cluster agents CD (the one containing agents for the *non*-Java System applications), together with the other two in a single arena. The agents CD can be spooled to a separate arena, if desired.

Sun Cluster 3.1 Software Licensing

No license keys are required for the Sun Cluster software. You must, however, furnish paper license numbers to obtain service.

Patches for OS and Patches for Sun Cluster Software

It is important to manage consistently the following types of patches:

- OS and hardware patches

  These must be installed *before* the Sun Cluster framework packages are installed. OS patches should be consistent between cluster nodes, although when you add new ones *after* cluster configuration it is almost always possible to do so in a rolling fashion, patching and rebooting one node at a time.

- Sun Cluster framework patches

  Available patches should be added after the Sun cluster framework packages are installed but before it is configured.

  You can add these patches manually before you configure the Sun Cluster framework with `scinstall`.

  Alternatively, `scinstall` itself can add these (or any) patch before it actually performs configuration steps.

  Almost all *new* patches will be able to be installed in a rolling fashion.

- Sun Cluster Data Service Agent patches

  You will not be able to add these until you install the particular agent being patched.

Installing the Sun Cluster Packages With the Java ES Installer

The Java ES installer provides both graphical and terminal-based interfaces for installing the Sun Cluster software. Other Java System applications can be installed along with Sun Cluster when Sun Cluster is included in a full Java ES release.

Prerequisites for Installing Sun Cluster Software

Before you can use the Java ES installer to install the Sun Cluster software framework on a cluster node, you must meet the following prerequisites:

1. Boot disks must be configured according to Sun Cluster standards, as defined in Module 3 of this course.
2. The Solaris OS and OS patches must be installed.

Auxiliary Software Automatically Installed by Java ES Installer

Sun Cluster 3.1 8/05 software (update 4) has several required auxiliary components. All of these components are automatically installed or upgraded as needed when you use the Java ES installer to install the Sun Cluster software.

The required auxiliary components are the following:

- The Common Agent Container (CACAO)

  This is a Java application that serves as a container for a Sun Cluster remote management interface that is implemented as a series of Java objects. In the standard Sun Cluster implementation, this interface is used only by the SunPlex Manager web interface.

- The Java Management Development Kit (JDMK) Runtime Libraries

  These Java Libraries are required by the management objects.

- Java Runtime Environment 1.5

  This is already included in a full installation of Solaris 10 OS, and the Java ES installer will recognize if the version present is already sufficient.

- The Sun Java Web Console Application 2.2

  Sun Java Web Console serves as a single point of entry and single sign-on for a variety of web-based applications, including the SunPlex Manager used in the Sun Cluster software. When you install the Sun Cluster software packages, the Java ES installer automatically detects a preexisting Sun Java Web Console installation and upgrades the software to the correct version if necessary. The Solaris 10 OS full installation already includes the correct version and it will be recognized by the Java ES installer

Running the Java ES Installer

If installing directly from CD, the Java ES installer is run from CDROM 1 of 2. If installing from spooled software, recall that CDROM 1 of 2 and CDROM 2 of 2 must have previously been spooled together into the same arena.

The installer will run as a graphical utility if you have a correct X-Windows DISPLAY variable set in your environment. It will run as a terminal-based utility if you do not have DISPLAY set, or if you explicitly run the installer with the -nodisplay option.

In the following subsections, not every screen is shown from the installer.

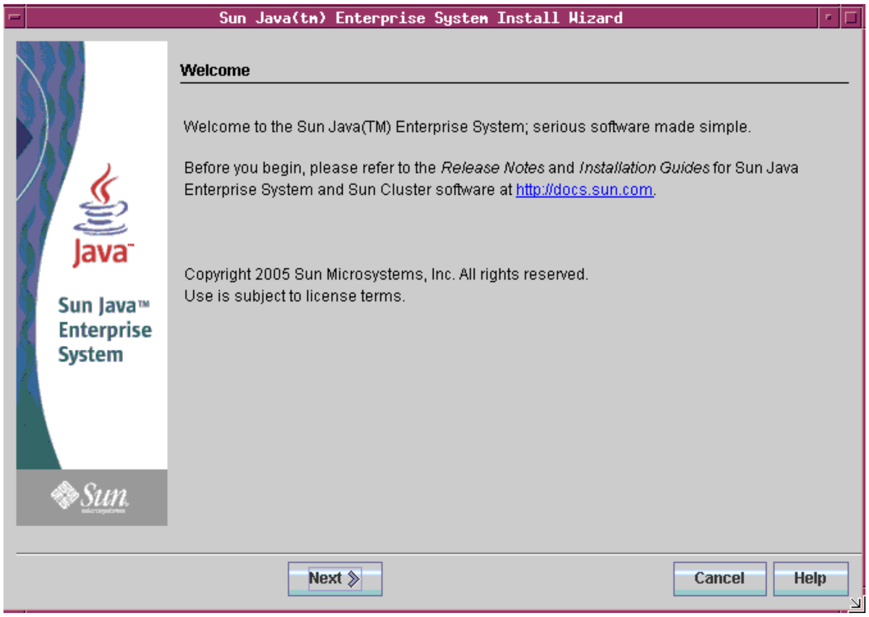
Launching the Java ES Installer

You run the installer from the Solaris_sparc or Solaris_x86 subdirectory of CDROM 1 of 2 (or of the combined spooled arena).

```
# DISPLAY=display-name-or-IP:display-#
# export DISPLAY

# cd cd-or-spooled-arena-path/Solaris_sparc
# ./installer
```

The screen shown in Figure 4-1 shows the initial screen for the graphical version of the Java ES installer (the splash screen is not shown).


Sun Java(tm) Enterprise System Install Wizard
Java
Sun Java™ Enterprise System
Sun
Next
Cancel
Help

Welcome

Welcome to the Sun Java(TM) Enterprise System; serious software made simple.

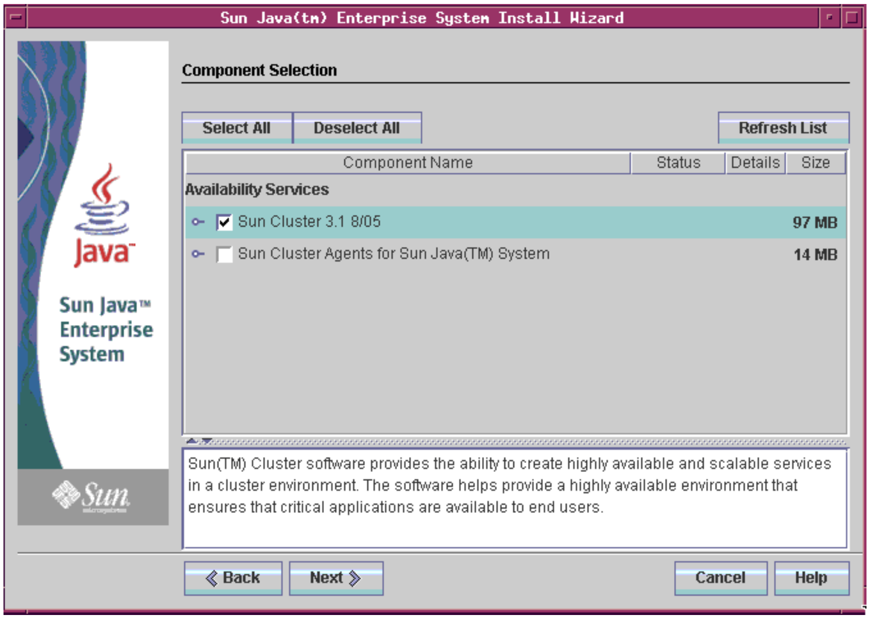
Before you begin, please refer to the *Release Notes* and *Installation Guides* for Sun Java Enterprise System and Sun Cluster software at http://docs.sun.com.

Copyright 2005 Sun Microsystems, Inc. All rights reserved.
Use is subject to license terms.

Figure 4-1 Java ES Installer Initial Screen

Java ES Component Selection

After the license and language screens (not shown), you can choose the Java ES components that you want. The screen shown in Figure 4-2 shows the possible selections when Sun Cluster is distributed as a standalone release without the other Java System components. Note that besides the Sun Cluster framework itself (shown checked in the figure), you can choose to install all or some of the agents for Sun Java System applications.


Sun Java(tm) Enterprise System Install Wizard
Java
Sun Java™ Enterprise System
Sun
Select All
Deselect All
Refresh List
Back
Next
Cancel
Help

Component Selection

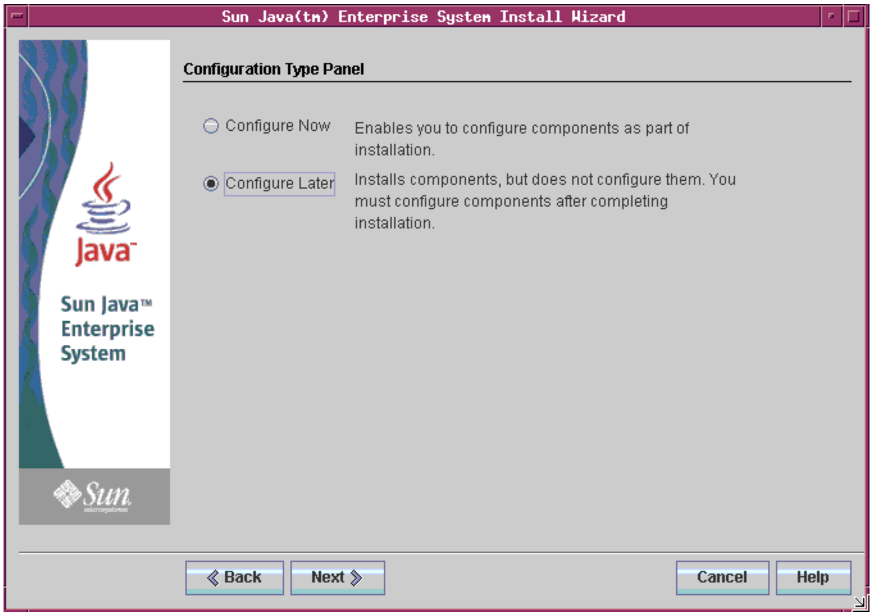
Component Name	Status	Details	Size
Availability Services			
☑ Sun Cluster 3.1 8/05			97 MB
□ Sun Cluster Agents for Sun Java(TM) System			14 MB

Sun(TM) Cluster software provides the ability to create highly available and scalable services in a cluster environment. The software helps provide a highly available environment that ensures that critical applications are available to end users.

Figure 4-2 Java ES Installer Component Selection

Java ES Configuration: Configure Later

For some of the *other* Java System applications, the Java ES installer has the ability to actually configure the applications. It does *not* have this ability for the Sun Cluster framework, and you must choose the `Configure Later` option, as shown in Figure 4-3:



Sun Java(tm) Enterprise System Install Wizard

Configuration Type Panel

○ Configure Now — Enables you to configure components as part of installation.

● Configure Later — Installs components, but does not configure them. You must configure components after completing installation.

Figure 4-3 Choose `Configure Later` for Sun Cluster

Software Installation

After confirmation, the installer will proceed to install the "shared components" (the auxiliary software discussed on page 4-8), and then the actual Sun Cluster framework packages.

You do *not* need to reboot your node before proceeding to the configuration of the Sun Cluster framework. You *will* be rebooting your node after the configuration.

Configuring the User `root` Environment

The `root` login environment should include the following search path and man page information:

```
PATH=$PATH:/usr/cluster/bin

MANPATH=$MANPATH:/usr/cluster/man:/usr/share/man
```

Sun Cluster Framework Configuration

The Sun Cluster configuration is done using one of the following three methods:

- Using the `scinstall` utility interactively – This is the most common method of configuring Sun Cluster, and the only one that will be discussed in detail in this module.
- Using jumpstart off a jumpstart server – The `scinstall` can be run on the jumpstart server in order to provision the jumpstart server so that the client (node) performs cluster configuration as part of jumpstart finish scripts.

  If you used this method, you would manually provision the jumpstart server so that the clients are provided a flash image from a cluster node on which the Java ES installer had been used to install the cluster software packages.
- Using the SunPlex Installer application – The SunPlex installer is a small, standalone web application. You automatically have access to the SunPlex installer if you reboot a node after having used the Java ES installer to install the cluster software.

Note – The SunPlex installer runs on `https://`*`nodename`*`:3000`. Its use is deprecated since it will not be supported in future versions of Sun Cluster software.

Understanding the `installmode` Flag

As you configure Sun Cluster software on cluster nodes and reboot the nodes into the cluster, a special flag called the `installmode` flag is set in the cluster CCR. When this flag is set, the following happens:

- The first node installed (node ID 1) has a quorum vote of 1
- All other nodes have a quorum vote of zero

This allows you to complete rebooting of the second node into the cluster while maintaining the quorum mathematics rules. If the second node had a vote (making a total of two in the cluster), the first node would kernel panic when the second node was rebooted after the cluster software was installed, as the first node would lose operational quorum!

One important side effect of the `installmode` flag is that you must be careful not to reboot the first node (node ID 1) until you can choose quorum devices and eliminate (reset) the `installmode` flag. If you accidentally reboot the first node, all the other nodes will kernel panic (as they will be left with zero votes out of a possible total of one).

If the installation is a single-node cluster, the `installmode` flag is not set. Post-installation steps to choose a quorum device and reset the `installmode` flag are unnecessary.

Automatic Quorum Configuration (Two-Node Cluster Only)

On a two node cluster only, you have the option (the defaults will always be to accept the option) of having the `scinstall` command insert a script that will automatically choose your quorum device as the second node boots into the cluster.

The quorum device chosen will be the first dual-ported disk or LUN (the one with the lowest Device ID (DID) number).

If you choose to allow automatic quorum configuration, the `installmode` flag will automatically be reset for you as well after the quorum is automatically configured.

You can disable two-node cluster automatic quorum configuration for one of the following reasons:

- You want to choose the quorum yourself.
- You have a dual-ported disk or LUN which is not capable of being a quorum device.
- You want to use a NAS device as a quorum device.

Automatic Reset of installmode Without Quorum (Clusters With More Than Two Nodes Only)

In clusters with more than two nodes, the scinstall will insert a script to automatically reset the installmode flag. It will *not* automatically configure a quorum device. You still have to do that manually after the install. By resetting installmode, each node will be assigned its proper single quorum vote.

Configuration Information Required by scinstall

The following information will be required by scinstall and should be prepared in advance.

Name of the Cluster and Names of All the Nodes

The cluster name is just a name agreed upon by the nodes, it is *not* a name that resolves to an IP address.

The nodes in the cluster *must* be able to resolve each other's host name. If for some reason this is true but the names are not in each node's /etc/hosts file (the names are resolvable through only NIS or DNS, for example), the /etc/hosts file will automatically be modified to include these names.

Location of Any Patches That You Want scinstall to Install for You

The scinstall utility can install patches. This is intended for patches to the cluster software framework itself, but it can be any patches. If you want scinstall to install patches, you must put them in one single directory, with an optional "patch order" file.

Cluster Transport IP Network Number and Netmask

It is recommended that you keep the default, allowing Sun Cluster to use the entire 172.16.x.x range of IP addresses for the cluster transport.

Note – When you specify a netmask of 255.255.0.0, it allows Sun Cluster to reserve the entire class B range of numbers for the cluster transport. It is *not* the actual netmask used by the transport adapters.

If you need to specify a different IP address range for the transport, you can do so. You will be asked for a netmask, but must still specify a class-B like netmask or larger.

Cluster Transport Adapters and Switches (Junctions)

You *must* be prepared to identify transport adapters on at least the first node on which you run `scinstall`. On other nodes you will normally let `scinstall` use its "auto-discovery" feature to automatically deduce the transport adapters.

You can define a two-node cluster topology as using switches or just using point-to-point cables. This does not even need to match the actual topology; the software really has no way of telling. It is just the definitions in the Cluster Configuration Repository (CCR), and whichever way you define it will be the way it is presented when you view the cluster configuration using command-line commands or the graphical web-based administration tool.

Names that you provide for switches are arbitrary, and are just used to match up transport connections between the various nodes.

Port names for specific switch connections are arbitrary except for SCI switches. For SCI, the port name must match the switch port number to which an actual cable is connected.

Partition or Placeholder File System for `/global/.devices/node@#`

You must have a partition on the root disk (recommended size 512 Mbytes) for this file system. While the default is that this is covered with an empty placeholder file system mounted on `/globaldevices`, you will be asked if you have a different placeholder file system or just an unused partition.

Using DES Authentication for Nodes to Authenticate With Each Other as They Join the Cluster

By default, nodes will be authenticated as they join the cluster using standard “Unix Authentication”. A reverse IP address lookup is done for a node trying to join the cluster, and if the resulting name is in the list of nodes that you typed in as nodes for this cluster, it will be allowed to add itself to the cluster.

The reasons for considering more stringent authentication are the following:

- Nodes that are adding themselves to the cluster communicate across the *public* network.
- A bogus node adding itself to the cluster can add bogus information to the CCR of existing nodes, or in fact, copy out *any* file from existing nodes

DES authentication, also known as secure remote procedure call (secure RPC) authentication, is a much stronger authentication that can not be spoofed by something simple like spoofing IP addresses.

Variations in Interactive `scinstall`

Following are the four different ways of using the interactive `scinstall` to do configuration of a brand new cluster:

- Configure entire cluster at once
  - Typical install
  - Custom install
- Configure cluster nodes one at a time
  - Typical install
  - Custom install

Configuring Entire Cluster at Once

If you choose the option to configure the entire cluster, you run `scinstall` on only one node. You should be aware of the following behavior:

- The node you are driving from will be the *last* node to join the cluster, as it needs to configure and reboot all the other nodes first.
- If you care about which node ID's are assigned to the nodes, you should drive from the node that you want to be the *highest* node ID, and list the other nodes in *reverse order*.
- The Sun Cluster software packages must already be installed on all nodes (by the Java ES installer). You therefore *do not* need remote shell access (neither `rsh` nor `ssh`) between the nodes. The remote configuration is done using Remote Procedure Calls (RPC) installed by the Sun Cluster packages. If you are concerned about authentication, you can use DES authentication as described on page 4-16.

Configuring Cluster Nodes One at a Time

If you choose this method, you run scinstall separately on each node.

You must complete scinstall and reboot into the cluster on the first node. This becomes the *sponsor node* to the remaining nodes.

If you have more than two nodes you can run scinstall simultaneously on all but the first node, but it might be hard to predict which node gets assigned which node ID. If you care, you should just run scinstall on the remaining nodes one at a time, and wait for each node to boot into the cluster before starting the next one.

Typical Installation Compared to Custom Installation

Both the “all at once” and “one-at-a-time” methods have Typical and Custom configuration options (to make a total of four variations).

The Typical configuration mode assumes the following responses:

- It will use network address 172.16.0.0 with netmask 255.255.0.0 for the cluster interconnect.
- It assumes you want to do autodiscovery of cluster transport adapters on the other nodes with the “all-at-once” method (on the “one-node-at-a-time” method it asks you if you want to use autodiscovery in both Typical and Custom modes).
- It uses the names switch1 and switch2 for the two transport junctions and assumes the use of junctions even for a two-node cluster.
- It assumes you have the standard empty /globaldevices mounted as an empty file system with the recommended size (512 Mbytes) to be remounted as the /global/.devices/node@# file system.
- It assumes you want to use standard system authentication (not DES authentication) for new nodes configuring themselves into the cluster.
- It checks if you have any patches first in the /var/cluster/patches directory and then in /var/patches. It will *not* add patches from both directories. If you have a file named patchlist in the same directory as the patches, it will be used as the patch order file.

Configuring Using All-at-Once and Typical Modes: Example

The following example shows the full dialog for the cluster installation that requires the least information, the "all-at-once," `Typical` mode installation. The example is from a two-node cluster, where the default will be to let `scinstall` set up a script that will automate configuration of the quorum device. In the example, `scinstall` is running on the node named `theo` which will become the *second* node (node ID 2).

```
# /usr/cluster/bin/scinstall

*** Main Menu ***

    Please select from one of the following (*) options:

      * 1) Install a cluster or cluster node
        2) Configure a cluster to be JumpStarted from this install server
        3) Add support for new data services to this cluster node
        4) Upgrade this cluster node
        5) Print release information for this cluster node

      * ?) Help with menu options
      * q) Quit

    Option:  1

*** Install Menu ***

    Please select from any one of the following options:

        1) Install all nodes of a new cluster
        2) Install just this machine as the first node of a new cluster
        3) Add this machine as a node in an existing cluster

        ?) Help with menu options
        q) Return to the Main Menu

    Option:  1
```

All-at-Once Introduction and Choosing Typical Compared to Custom

Selecting option 1 from the `scinstall` install menu presents a description of the requirements necessary for installation of the entire cluster from a single node as follows:

```
*** Installing all Nodes of a New Cluster ***


    This option is used to install and configure a new cluster.

    If either remote shell (see rsh(1)) or secure shell (see ssh(1)) root
    access is enabled to all of the new member nodes from this node, the
    Sun Cluster framework software will be installed on each node.
    Otherwise, the Sun Cluster software must already be pre-installed on
    each node with the "remote configuration" option enabled.

    The Java Enterprise System installer can be used to install the Sun
    Cluster framework software with the "remote configuration" option
    enabled. Since the installation wizard does not yet include support
    for cluster configuration, you must still use scinstall to complete
    the configuration process.

    Press Control-d at any time to return to the Main Menu.


    Do you want to continue (yes/no) [yes]?  yes
```

Type of Installation

```
>>> Type of Installation <<<

    There are two options for proceeding with cluster installation. For
    most clusters, a Typical installation is recommended. However, you
    might need to select the Custom option if not all of the Typical
    defaults can be applied to your cluster.

    For more information about the differences between the Typical and
    Custom installation methods, select the Help option from the menu.

    Please select from one of the following options:
        1) Typical
        2) Custom

        ?) Help
        q) Return to the Main Menu

    Option [1]: 1
```

Cluster Name, Cluster Nodes, and Remote Installation Confirmation

```
>>> Cluster Name <<<

    Each cluster has a name assigned to it. The name can be made up of
    any characters other than whitespace. Each cluster name should be
    unique within the namespace of your enterprise.

    What is the name of the cluster you want to establish? orangecat


>>> Cluster Nodes <<<

    This Sun Cluster release supports a total of up to 16 nodes.

    Please list the names of the other nodes planned for the initial
    cluster configuration. List one node name per line. When finished,
    type Control-D:

    Node name (Control-D to finish): vincent
    Node name (Control-D to finish):  ^D


    This is the complete list of nodes:

        vincent
        theo

    Is it correct (yes/no) [yes]? yes

Attempting to contact "vincent" ... done

    Searching for a remote install method ... done

    The Sun Cluster framework software is already installed on each of
    the new nodes of this cluster. And, it is able to complete the
    configuration process without remote shell access.


Press Enter to continue:
```

Cluster Transport Adapters

```
>>> Cluster Transport Adapters and Cables <<<

    You must identify the two cluster transport adapters which attach
    this node to the private cluster interconnect.

    Select the first cluster transport adapter for "theo":

        1) hme0
        2) qfe0
        3) qfe3
        4) Other

    Option: 1

    Searching for any unexpected network traffic on "hme0" ... done
    Verification completed. No traffic was detected over a 10 second
    sample period.

    Select the second cluster transport adapter for "theo":

        1) hme0
        2) qfe0
        3) qfe3
        4) Other

    Option: 2
```

Cluster Transport Adapters (Tagged VLAN-Capable Adapters)

The following shows a slight difference in the questionnaire when you have private network adapters capable of tagged VLANs. In this example, we do *not* have a shared physical adapter, and therefore do not need to use the tagged VLAN feature.

```
>>> Cluster Transport Adapters and Cables <<<

  You must identify the two cluster transport adapters which attach
  this node to the private cluster interconnect.

  Select the first cluster transport adapter for "dani":

      1) bge1
      2) ce1
      3) Other

  Option: 1

  Will this be a dedicated cluster transport adapter (yes/no) [yes]?yes

  Searching for any unexpected network traffic on "bge1" ... done
  Verification completed. No traffic was detected over a 10 second
  sample period.

  Select the second cluster transport adapter for "dani":

      1) bge1
      2) ce1
      3) Other

  Option:  2

  Will this be a dedicated cluster transport adapter (yes/no) [yes]?yes

  Searching for any unexpected network traffic on "ce1" ... done
  Verification completed. No traffic was detected over a 10 second
  sample period.
```

Cluster Transport Adapters (Actually Using Shared Adapters)

In the following variation we actually choose an adapter for the private network which is already configured (already using tagged VLAN) for the public network.

```
>>> Cluster Transport Adapters and Cables <<<

  You must identify the two cluster transport adapters which attach
  this node to the private cluster interconnect.

  Select the first cluster transport adapter for "dani":

      1) bge1
      2) ce0
      3) ce1
      4) Other

  Option:  2

  This adapter is used on the public network also, you will need to
  configure it as a tagged VLAN adapter for cluster transport.

  What is the cluster transport VLAN ID for this adapter?  5

  Searching for any unexpected network traffic on "ce5000" ... done
  Verification completed. No traffic was detected over a 10 second
  sample period.
```

Automatic Quorum Configuration

```
  >>> Quorum Configuration <<<

    Every two-node cluster requires at least one quorum device. By
    default, scinstall will select and configure a shared SCSI quorum
    disk device for you.

    This screen allows you to disable the automatic selection and
    configuration of a quorum device.

    The only time that you must disable this feature is when ANY of the
    shared storage in your cluster is not qualified for use as a Sun
    Cluster quorum device. If your storage was purchased with your
    cluster, it is qualified. Otherwise, check with your storage vendor
    to determine whether your storage device is supported as Sun Cluster
    quorum device.

    If you disable automatic quorum device selection now, or if you
    intend to use a quorum device that is not a shared SCSI disk, you
    must instead use scsetup(1M) to manually configure quorum once both
    nodes have joined the cluster for the first time.

    Do you want to disable automatic quorum device selection (yes/no)
[no]? no
```

Installation Verification and **sccheck**

```
Is it okay to begin the installation (yes/no) [yes]? yes

    During the installation process, sccheck(1M) is run on each of the
    new cluster nodes. If sccheck(1M) detects problems, you can either
    interrupt the installation process or check the log files after
    installation has completed.

    Interrupt the installation for sccheck errors (yes/no) [no]?  no
```

Installation and Configuration Messages

```
Installation and Configuration

    Log file - /var/cluster/logs/install/scinstall.log.20852

    Testing for "/globaldevices" on "theo" ... done
    Testing for "/globaldevices" on "vincent" ... done

    Starting discovery of the cluster transport configuration.

    The following connections were discovered:

        theo:hme0  switch1  vincent:hme0
        theo:qfe0  switch2  vincent:qfe0

    Completed discovery of the cluster transport configuration.

    Started sccheck on "theo".
    Started sccheck on "vincent".

    sccheck completed with no errors or warnings for "theo".
    sccheck completed with no errors or warnings for "vincent".


    Configuring "vincent" ... done
    Rebooting "vincent" ...

    Configuring "theo" ... done
    Rebooting "theo" ...

Log file - /var/cluster/logs/install/scinstall.log.20852


Rebooting ...
```

Configuring Using One-at-a-Time and Custom Modes: Example (First Node)

The following is an example of using the one-node-at-a-time configuration. The dialog is shown for `vincent`, the first node in the cluster. You can not install other cluster nodes until this node is rebooted into the cluster, and can then be the *sponsor node* for the other nodes.

```
vincent:/# /usr/cluster/bin/scinstall
*** Main Menu ***

    Please select from one of the following (*) options:

      * 1) Install a cluster or cluster node
        2) Configure a cluster to be JumpStarted from this install server
        3) Add support for new data services to this cluster node
        4) Upgrade this cluster node
        5) Print release information for this cluster node

      * ?) Help with menu options
      * q) Quit

    Option: 1

*** Install Menu ***

    Please select from any one of the following options:

        1) Install all nodes of a new cluster
        2) Install just this machine as the first node of a new cluster
        3) Add this machine as a node in an existing cluster

        ?) Help with menu options
        q) Return to the Main Menu

    Option: 2
```

First Node Introduction and Choosing Typical versus Custom

```
*** Installing just the First Node of a New Cluster ***


  This option is used to establish a new cluster using this machine as
  the first node in that cluster.

  Once the cluster framework software is installed, you will be asked
  for the name of the cluster. Then, you will have the opportunity to
  run sccheck(1M) to test this machine for basic Sun Cluster
  pre-configuration requirements.

  After sccheck(1M) passes, you will be asked for the names of the
  other nodes which will initially be joining that cluster. Unless this
  is a single-node cluster, you will be also be asked to provide
  certain cluster transport configuration information.

  Press Control-d at any time to return to the Main Menu.


  Do you want to continue (yes/no) [yes]? yes
```

Type of Installation

```
>>> Type of Installation <<<

  There are two options for proceeding with cluster installation. For
  most clusters, a Typical installation is recommended. However, you
  might need to select the Custom option if not all of the Typical
  defaults can be applied to your cluster.

  For more information about the differences between the Typical and
  Custom installation methods, select the Help option from the menu.

  Please select from one of the following options:

      1) Typical
      2) Custom

      ?) Help
      q) Return to the Main Menu

  Option [1]: 2
```

Patch Installation Option

```
>>> Software Patch Installation <<<

    If there are any Solaris or Sun Cluster patches that need to be added
    as part of this Sun Cluster installation, scinstall can add them for
    you. All patches that need to be added must first be downloaded into
    a common patch directory. Patches can be downloaded into the patch
    directory either as individual patches or as patches grouped together
    into one or more tar, jar, or zip files.

    If a patch list file is provided in the patch directory, only those
    patches listed in the patch list file are installed. Otherwise, all
    patches found in the directory will be installed. Refer to the
    patchadd(1M) man page for more information regarding patch list
files.

    Do you want scinstall to install patches for you (yes/no) [yes]?  no
```

Cluster Name

```
>>> Cluster Name <<<

    Each cluster has a name assigned to it. The name can be made up of
    any characters other than whitespace. Each cluster name should be
    unique within the namespace of your enterprise.

    What is the name of the cluster you want to establish?  orangecat
```

Option for sccheck

```
>>> Check <<<

    This step allows you to run sccheck(1M) to verify that certain basic
    hardware and software pre-configuration requirements have been met.
    If sccheck(1M) detects potential problems with configuring this
    machine as a cluster node, a report of failed checks is prepared and
    available for display on the screen. Data gathering and report
    generation can take several minutes, depending on system
    configuration.

    Do you want to run sccheck (yes/no) [yes]?  no
```

Cluster Nodes

```
>>> Cluster Nodes <<<

    This Sun Cluster release supports a total of up to 16 nodes.

    Please list the names of the other nodes planned for the initial
    cluster configuration. List one node name per line. When finished,
    type Control-D:

    Node name (Control-D to finish):  theo
    Node name (Control-D to finish):  ^D


    This is the complete list of nodes:

        vincent
        theo

    Is it correct (yes/no) [yes]?  yes
```

Authenticating Nodes with DES

```
>>> Authenticating Requests to Add Nodes <<<

    Once the first node establishes itself as a single node cluster,
    other nodes attempting to add themselves to the cluster configuration
    must be found on the list of nodes you just provided. You can modify
    this list using scconf(1M) or other tools once the cluster has been
    established.

    By default, nodes are not securely authenticated as they attempt to
    add themselves to the cluster configuration. This is generally
    considered adequate, since nodes which are not physically connected
    to the private cluster interconnect will never be able to actually
    join the cluster. However, DES authentication is available. If DES
    authentication is selected, you must configure all necessary
    encryption keys before any node will be allowed to join the cluster
    (see keyserv(1M), publickey(4)).

    Do you need to use DES authentication (yes/no) [no]?  no
```

Transport IP Address Range and Netmask

```
>>> Network Address for the Cluster Transport <<<

    The private cluster transport uses a default network address of
    172.16.0.0. But, if this network address is already in use elsewhere
    within your enterprise, you may need to select another address from
    the range of recommended private addresses (see RFC 1918 for
    details).

    If you do select another network address, bear in mind that the Sun
    Cluster software requires that the rightmost two octets always be
    zero.

    The default netmask is 255.255.0.0. You can select another netmask,
    as long as it minimally masks all bits given in the network address.

    Is it okay to accept the default network address (yes/no) [yes]? yes

    Is it okay to accept the default netmask (yes/no) [yes]?  yes
```

Choosing Whether to Define Switches and Switch Names

```
>>> Point-to-Point Cables <<<

    The two nodes of a two-node cluster may use a directly-connected
    interconnect. That is, no cluster transport junctions are configured.
    However, when there are greater than two nodes, this interactive form
    of scinstall assumes that there will be exactly two cluster transport
    junctions.

    Does this two-node cluster use transport junctions (yes/no) [yes]?yes

  >>> Cluster Transport Junctions <<<

    All cluster transport adapters in this cluster must be cabled to a
    transport junction, or "switch". And, each adapter on a given node
    must be cabled to a different junction. Interactive scinstall
    requires that you identify two switches for use in the cluster and
    the two transport adapters on each node to which they are cabled.

    What is the name of the first junction in the cluster [switch1]?<CR>

    What is the name of the second junction in the cluster [switch2]?<CR>
```

Transport Adapters and Connections to Switches

```
>>> Cluster Transport Adapters and Cables <<<

    You must configure at least two cluster transport adapters for each
    node in the cluster. These are the adapters which attach to the
    private cluster interconnect.

    Select the first cluster transport adapter:

        1) hme0
        2) qfe0
        3) qfe3
        4) Other

    Option:  1


    Adapter "hme0" is an Ethernet adapter.

    Searching for any unexpected network traffic on "hme0" ... done
    Verification completed. No traffic was detected over a 10 second
    sample period.

    The "dlpi" transport type will be set for this cluster.

    Name of the junction to which "hme0" is connected [switch1]?  <CR>

    Each adapter is cabled to a particular port on a transport junction.
    And, each port is assigned a name. You can explicitly assign a name
    to each port. Or, for Ethernet and Infiniband switches, you can
    choose to allow scinstall to assign a default name for you. The
    default port name assignment sets the name to the node number of the
    node hosting the transport adapter at the other end of the cable.

    For more information regarding port naming requirements, refer to the
    scconf_transp_jct family of man pages (e.g.,
    scconf_transp_jct_dolphinswitch(1M)).

    Use the default port name for the "hme0" connection (yes/no) [yes]?
yes
```

Second Transport Adapter

```
    Select the second cluster transport adapter:

        1) hme0
        2) qfe0
        3) qfe3
        4) Other

    Option:  2

    Adapter "qfe0" is an Ethernet adapter.

    Searching for any unexpected network traffic on "qfe0" ... done
    Verification completed. No traffic was detected over a 10 second
    sample period.

    Name of the junction to which "qfe0" is connected [switch2]?  switch2

    Use the default port name for the "qfe0" connection (yes/no) [yes]?
yes
```

Global Devices File System

```
>>> Global Devices File System <<<

    Each node in the cluster must have a local file system mounted on
    /global/.devices/node@<nodeID> before it can successfully participate
    as a cluster member. Since the "nodeID" is not assigned until
    scinstall is run, scinstall will set this up for you.

    You must supply the name of either an already-mounted file system or
    raw disk partition which scinstall can use to create the global
    devices file system. This file system or partition should be at least
    512 MB in size.

    If an already-mounted file system is used, the file system must be
    empty. If a raw disk partition is used, a new file system will be
    created for you.

    The default is to use /globaldevices.

    Is it okay to use this default (yes/no) [yes]? yes
```

Automatic Quorum Configuration (Two-Node Cluster)

```
>>> Quorum Configuration <<<

    Every two-node cluster requires at least one quorum device. By
    default, scinstall will select and configure a shared SCSI quorum
    disk device for you.

    This screen allows you to disable the automatic selection and
    configuration of a quorum device.

    The only time that you must disable this feature is when ANY of the
    shared storage in your cluster is not qualified for use as a Sun
    Cluster quorum device. If your storage was purchased with your
    cluster, it is qualified. Otherwise, check with your storage vendor
    to determine whether your storage device is supported as Sun Cluster
    quorum device.

    If you disable automatic quorum device selection now, or if you
    intend to use a quorum device that is not a shared SCSI disk, you
    must instead use scsetup(1M) to manually configure quorum once both
    nodes have joined the cluster for the first time.

    Do you want to disable automatic quorum device selection (yes/no)
[no]? no
```

Automatic Reboot

```
  >>> Automatic Reboot <<<

    Once scinstall has successfully installed and initialized the Sun
    Cluster software for this machine, it will be necessary to reboot.
    After the reboot, this machine will be established as the first node
    in the new cluster.

    Do you want scinstall to reboot for you (yes/no) [yes]? yes
```

Option Confirmation

```
>>> Confirmation <<<

    Your responses indicate the following options to scinstall:

      scinstall -ik \
           -C orangecat \
           -F \
           -T node=vincent,node=theo,authtype=sys \
           -A trtype=dlpi,name=hme0 -A trtype=dlpi,name=qfe0 \
           -B type=switch,name=switch1 -B type=switch,name=switch2 \
           -m endpoint=:hme0,endpoint=switch1 \
           -m endpoint=:qfe0,endpoint=switch2 \
           -P task=quorum,state=INIT

    Are these the options you want to use (yes/no) [yes]? yes

    Do you want to continue with the install (yes/no) [yes]? yes
```

Configuration Messages

Some of the post-installation steps for which scinstall is printing out messages here (NTP, hosts file, nsswitch.conf file, and IPMP) are discussed later in this module.

```
Checking device to use for global devices file system ... done

Initializing cluster name to "orangecat" ... done
Initializing authentication options ... done
Initializing configuration for adapter "hme0" ... done
Initializing configuration for adapter "qfe0" ... done
Initializing configuration for junction "switch1" ... done
Initializing configuration for junction "switch2" ... done
Initializing configuration for cable ... done
Initializing configuration for cable ... done

Setting the node ID for "vincent" ... done (id=1)

Checking for global devices global file system ... done
Updating vfstab ... done

Verifying that NTP is configured ... done
Initializing NTP configuration ... done

Updating nsswitch.conf ...
done

Adding clusternode entries to /etc/inet/hosts ... done

Configuring IP Multipathing groups in "/etc/hostname.<adapter>" files

IP Multipathing already configured in "/etc/hostname.qfe1".
IP Multipathing already configured in "/etc/hostname.qfe2".

Verifying that power management is NOT configured ... done
Unconfiguring power management ... done
/etc/power.conf has been renamed to /etc/power.conf.062605225734
Power management is incompatible with the HA goals of the cluster.
Please do not attempt to re-configure power management.

Ensure that the EEPROM parameter "local-mac-address?" is set to "true"
... done
Ensure network routing is disabled ... done
Log file - /var/cluster/logs/install/scinstall.log.294
Rebooting ...
```

Configuring Additional Nodes for One-at-a-Time Method: Example

In the one-at-a-time method, once the first node has rebooted into the cluster, you can configure the remaining node or nodes. Here, there is not as much difference between the `Typical` and `Custom` modes. You will see in this example, the `Typical` mode, it does not ask about patches or about the global devices file system. In a two node cluster, you have no choice about the automatic quorum selection or the authentication mechanism, since it was already chosen on the first node.

```
theo:/# scinstall

  *** Main Menu ***

    Please select from one of the following (*) options:

      * 1) Install a cluster or cluster node
        2) Configure a cluster to be JumpStarted from this install server
        3) Add support for new data services to this cluster node
        4) Upgrade this cluster node
        5) Print release information for this cluster node

      * ?) Help with menu options
      * q) Quit

    Option:  1

*** Install Menu ***

    Please select from any one of the following options:

        1) Install all nodes of a new cluster
        2) Install just this machine as the first node of a new cluster
        3) Add this machine as a node in an existing cluster

        ?) Help with menu options
        q) Return to the Main Menu

    Option:  3
```

Additional Node Configuration and Choosing Typical Versus Custom

```
  *** Adding a Node to an Existing Cluster ***


    This option is used to add this machine as a node in an already
    established cluster. If this is an initial cluster install, there may
    only be a single node which has established itself in the new
cluster.

    Once the cluster framework software is installed, you will be asked
    to provide both the name of the cluster and the name of one of the
    nodes already in the cluster. Then, sccheck(1M) is run to test this
    machine for basic Sun Cluster pre-configuration requirements.

    After sccheck(1M) passes, you may be asked to provide certain cluster
    transport configuration information.

    Press Control-d at any time to return to the Main Menu.


    Do you want to continue (yes/no) [yes]? yes
```

Type of Installation

```
>>> Type of Installation <<<

    There are two options for proceeding with cluster installation. For
    most clusters, a Typical installation is recommended. However, you
    might need to select the Custom option if not all of the Typical
    defaults can be applied to your cluster.

    For more information about the differences between the Typical and
    Custom installation methods, select the Help option from the menu.

    Please select from one of the following options:

        1) Typical
        2) Custom

        ?) Help
        q) Return to the Main Menu

    Option [1]: 1
```

Sponsoring Node

After entering the name of the sponsor node (first node already booted into the cluster and the cluster name), the authentication and the cluster name are checked with the sponsor node.

```
>>> Sponsoring Node <<<

    For any machine to join a cluster, it must identify a node in that
    cluster willing to "sponsor" its membership in the cluster. When
    configuring a new cluster, this "sponsor" node is typically the first
    node used to build the new cluster. However, if the cluster is
    already established, the "sponsoring" node can be any node in that
    cluster.

    Already established clusters can keep a list of hosts which are able
    to configure themselves as new cluster members. This machine should
    be in the join list of any cluster which it tries to join. If the
    list does not include this machine, you may need to add it using
    scconf(1M) or other tools.


    And, if the target cluster uses DES to authenticate new machines
    attempting to configure themselves as new cluster members, the
    necessary encryption keys must be configured before any attempt to
    join.

    What is the name of the sponsoring node?  vincent
```

Cluster Name

```
>>> Cluster Name <<<

    Each cluster has a name assigned to it. When adding a node to the
    cluster, you must identify the name of the cluster you are attempting
    to join. A sanity check is performed to verify that the "sponsoring"
    node is a member of that cluster.

    What is the name of the cluster you want to join?  orangecat

    Attempting to contact "vincent" ... done

    Cluster name "orangecat" is correct.

Press Enter to continue:
```

Option for `sccheck`

```
>>> Check <<<

    This step allows you to run sccheck(1M) to verify that certain basic
    hardware and software pre-configuration requirements have been met.
    If sccheck(1M) detects potential problems with configuring this
    machine as a cluster node, a report of failed checks is prepared and
    available for display on the screen. Data gathering and report
    generation can take several minutes, depending on system
    configuration.

    Do you want to run sccheck (yes/no) [yes]?  no
```

Cluster Transport Autodiscovery Option

```
  >>> Autodiscovery of Cluster Transport <<<

    If you are using Ethernet or Infiniband adapters as the cluster
    transport adapters, autodiscovery is the best method for configuring
    the cluster transport.

    Do you want to use autodiscovery (yes/no) [yes]?  yes


    Probing ........

    The following connections were discovered:

        vincent:hme0  switch1  theo:hme0
        vincent:qfe0  switch2  theo:qfe0

    Is it okay to add these connections to the configuration (yes/no)
[yes]? yes
```

Automatic Reboot and Option Confirmation

```
>>> Automatic Reboot <<<

    Once scinstall has successfully installed and initialized the Sun
    Cluster software for this machine, it will be necessary to reboot.
    The reboot will cause this machine to join the cluster for the first
    time.

    Do you want scinstall to reboot for you (yes/no) [yes]?
>>> Confirmation <<<

    Your responses indicate the following options to scinstall:

      scinstall -ik \
           -C orangecat \
           -N vincent \
           -A trtype=dlpi,name=hme0 -A trtype=dlpi,name=qfe0 \
           -m endpoint=:hme0,endpoint=switch1 \
           -m endpoint=:qfe0,endpoint=switch2

    Are these the options you want to use (yes/no) [yes]?  yes

    Do you want to continue with the install (yes/no) [yes]? yes
```

Configuration Messages

```
Checking device to use for global devices file system ... done

Adding node "theo" to the cluster configuration ... done
Adding adapter "hme0" to the cluster configuration ... done
Adding adapter "qfe0" to the cluster configuration ... done
Adding cable to the cluster configuration ... done
Adding cable to the cluster configuration ... done

Copying the config from "vincent" ... done

Copying the postconfig file from "vincent" if it exists ... done
Copying the Common Agent Container keys from "vincent" ... done

Setting the node ID for "theo" ... done (id=2)

Verifying the major number for the "did" driver with "vincent" ... done

Checking for global devices global file system ... done
Updating vfstab ... done

Verifying that NTP is configured ... done
Initializing NTP configuration ... done

Updating nsswitch.conf ... done

Adding clusternode entries to /etc/inet/hosts ... done

Configuring IP Multipathing groups in "/etc/hostname.<adapter>" files

IP Multipathing already configured in "/etc/hostname.qfe1".
IP Multipathing already configured in "/etc/hostname.qfe2".

Verifying that power management is NOT configured ... done
Unconfiguring power management ... done
/etc/power.conf has been renamed to /etc/power.conf.062605232015
Power management is incompatible with the HA goals of the cluster.
Please do not attempt to re-configure power management.

Ensure that the EEPROM parameter "local-mac-address?" is set to "true"
... done

Ensure network routing is disabled ... done

Updating file ("ntp.conf.cluster") on node vincent ... done
Updating file ("hosts") on node vincent ... done

Log file - /var/cluster/logs/install/scinstall.log.2949


Rebooting ...
```

Solaris OS Files and Settings Automatically Configured by scinstall

No matter which of the four scinstall variations that you use, scinstall automatically configures the following files and settings on each cluster node:

- /etc/hosts
- /etc/nsswitch.conf
- /etc/inet/ntp.conf.cluster
- /etc/hostname.*xxx*
- /etc/vfstab
- /etc/notrouter
- local-mac-address? setting in electrically erasable programmable read-only memory (EEPROM) (SPARC only)

Changes to /etc/hosts

The scinstall utility automatically adds all the cluster names and IP addresses to each node's hosts file if it was not there already (all the names already had to be *resolvable*, through some name service, for scinstall to work at all).

Changes to /etc/nsswitch.conf

The scinstall utility makes the following changes:

- It makes sure the files keyword precedes every other name service for every entry in the file.
- It adds the cluster keyword for the hosts and netmasks keywords. This keyword modifies the standard Solaris OS resolution libraries so that they can resolve the cluster transport host names and netmasks directly from the CCR. The default transport host names (associated with IP addresses on the clprivnet0 adapter) are clusternode1-priv, clusternode2-priv, and so on. These names can be used by any utility or application as normal resolvable names without having to be entered in any other name service.

Creation of File /etc/inet/ntp.conf.cluster

This file contains a Network Time Protocol Configuration which, if used, will have all nodes synchronize their time clocks against each other (and *only* against each other). Starting in Sun Cluster 3.1 8/05 (Update 4) this file automatically contains only lines for cluster nodes defined during scinstall, and therefore should not need to be modified. For a two-node cluster, for example, it will include the lines:

```
peer clusternode1-priv prefer
peer clusternode2-priv
```

The entire file will be ignored if there is a standard /etc/inet/ntp.conf file at boot time.

Modification of /etc/hostname.*xxx* Files to Include IPMP

Any *existing* /etc/hostname.*xxx* files that do not yet indicate IPMP group membership are rewritten so that the adapter in question is placed in a singleton (one-member) IPMP group. The following shows an example file, as modified by scinstall:

```
vincent:/# cat /etc/hostname.qfe1
vincent netmask + broadcast + group sc_ipmp0 up
```

Any real, multiadapter IPMP groups providing real failover must still be configured by hand, as you will learn in Module 8. If they are already configured *before* scinstall is run then they are untouched.

Modification of /etc/vfstab

The vfstab file is modified so that the /global/.devices/node@# mount point replaces any previous placeholder, such as /globaldevices. In addition, the DID device is used for this file system rather than the traditional /dev/rdsk/c#t#d#. This ensures that each of these devices has a unique name cluster-wide. The following shows the modification to the vfstab file:

```
vincent:/etc/inet# grep global /etc/vfstab
#/dev/dsk/c0t0d0s4      /dev/rdsk/c0t0d0s4 /globaldevices  ufs 2 yes -
/dev/did/dsk/d1s4 /dev/did/rdsk/d1s4 /global/.devices/node@1 ufs 2 no
global
```

Insertion of /etc/notrouter

This empty file assures that cluster nodes do not accidentally turn themselves into routers. It is not supported to have a cluster node function as a router.

Modification of local-mac-address?

On SPARC systems, this EEPROM variable is set to `true` so that each network adapter is given a unique Media Access Control (MAC) address (that is, Ethernet address for Ethernet adapters) in order to support IPMP. This is discussed further in Module 8.

Automatic Quorum Configuration and `installmode` Resetting

On a two-node cluster on which you chose to allow automatic quorum configuration, the quorum device is chosen (the lowest possible DID device number) as the second node boots into the cluster for the first time.

If your cluster has more than two nodes, no quorum device is selected automatically, but the `installmode` flag is automatically reset as the last node boots into the cluster.

On Solaris 8 and 9 OS, the auto-configuration of the quorum happens as part of a boot script on the last node booting into the cluster. It will be complete by the time you get the login prompt on this last node.

In Solaris 10 OS, as the last node boots into the cluster, you get the login prompt on the last node booting into the cluster *before* the quorum auto-configuration runs. This is because the boot environment is controlled by the SMF of Solaris 10 OS, which runs boot services in parallel and gives you the login prompt before many of the services are complete. The auto-configuration of the quorum device does not complete until a minute or so later. Do *not* attempt to configure the quorum device by hand, as the auto-configuration will eventually run to completion.

The following console messages indicate the automatic quorum selection:

```
Jun 26 23:23:57 vincent cl_runtime: NOTICE: CMM: Votecount changed from 0
to 1 for node theo.
Jun 26 23:23:57 vincent cl_runtime: NOTICE: CMM: Cluster members: vincent
theo.
Jun 26 23:23:57 vincent cl_runtime: NOTICE: CMM: node reconfiguration #5
completed.
Jun 26 23:23:58 vincent cl_runtime: NOTICE: CMM: Quorum device 1
(/dev/did/rdsk/d4s2) added; votecount = 1, bitmask of nodes with
configured paths = 0x3.
Jun 26 23:23:58 vincent cl_runtime: NOTICE: CMM: Registered key on and
acquiredquorum device 1 (gdevname /dev/did/rdsk/d4s2).
Jun 26 23:23:58 vincent cl_runtime: NOTICE: CMM: Quorum device
/dev/did/rdsk/d4s2: owner set to node 1.
Jun 26 23:23:59 vincent cl_runtime: NOTICE: CMM: Cluster members: vincent
theo.
Jun 26 23:23:59 vincent cl_runtime: NOTICE: CMM: node reconfiguration #6
completed
```

Manual Quorum Selection

You will need to choose a quorum device or quorum devices manually in the following circumstances:

- Two-node cluster where you disabled automatic quorum selection
- Any cluster of more than two nodes

Verifying DID Devices

If you are going to be manually choosing quorum devices that are physically attached disks or LUN’s (as opposed to a NAS device quorum), you need to know the DID device number for the quorum device or devices that you want to choose.

The `scdidadm` command shows the DID numbers assigned to the disks in the cluster.

You need to know the DID device number for the quorum device you choose in the next step. You can choose any multiported disk.

Note – The local disks (single-ported) appear at the beginning and end of the output and cannot be chosen as quorum devices.

```
# scdidadm -L
1        vincent:/dev/rdsk/c0t0d0        /dev/did/rdsk/d1
2        vincent:/dev/rdsk/c0t1d0        /dev/did/rdsk/d2
3        vincent:/dev/rdsk/c0t6d0        /dev/did/rdsk/d3
4        vincent:/dev/rdsk/c1t0d0        /dev/did/rdsk/d4
4        theo:/dev/rdsk/c1t0d0           /dev/did/rdsk/d4
5        vincent:/dev/rdsk/c1t1d0        /dev/did/rdsk/d5
5        theo:/dev/rdsk/c1t1d0           /dev/did/rdsk/d5
6        vincent:/dev/rdsk/c1t2d0        /dev/did/rdsk/d6
6        theo:/dev/rdsk/c1t2d0           /dev/did/rdsk/d6
7        vincent:/dev/rdsk/c1t3d0        /dev/did/rdsk/d7
7        theo:/dev/rdsk/c1t3d0           /dev/did/rdsk/d7
8        vincent:/dev/rdsk/c1t8d0        /dev/did/rdsk/d8
```

```
8         theo:/dev/rdsk/c1t8d0          /dev/did/rdsk/d8
9         vincent:/dev/rdsk/c1t9d0       /dev/did/rdsk/d9
9         theo:/dev/rdsk/c1t9d0          /dev/did/rdsk/d9
10        vincent:/dev/rdsk/c1t10d0      /dev/did/rdsk/d10
10        theo:/dev/rdsk/c1t10d0         /dev/did/rdsk/d10
11        vincent:/dev/rdsk/c1t11d0      /dev/did/rdsk/d11
11        theo:/dev/rdsk/c1t11d0         /dev/did/rdsk/d11
12        vincent:/dev/rdsk/c2t0d0       /dev/did/rdsk/d12
12        theo:/dev/rdsk/c2t0d0          /dev/did/rdsk/d12
13        vincent:/dev/rdsk/c2t1d0       /dev/did/rdsk/d13
13        theo:/dev/rdsk/c2t1d0          /dev/did/rdsk/d13
14        vincent:/dev/rdsk/c2t2d0       /dev/did/rdsk/d14
14        theo:/dev/rdsk/c2t2d0          /dev/did/rdsk/d14
15        vincent:/dev/rdsk/c2t3d0       /dev/did/rdsk/d15
15        theo:/dev/rdsk/c2t3d0          /dev/did/rdsk/d15
16        vincent:/dev/rdsk/c2t8d0       /dev/did/rdsk/d16
16        theo:/dev/rdsk/c2t8d0          /dev/did/rdsk/d16
17        vincent:/dev/rdsk/c2t9d0       /dev/did/rdsk/d17
17        theo:/dev/rdsk/c2t9d0          /dev/did/rdsk/d17
18        vincent:/dev/rdsk/c2t10d0      /dev/did/rdsk/d18
18        theo:/dev/rdsk/c2t10d0         /dev/did/rdsk/d18
19        vincent:/dev/rdsk/c2t11d0      /dev/did/rdsk/d19
19        theo:/dev/rdsk/c2t11d0         /dev/did/rdsk/d19
20        theo:/dev/rdsk/c0t0d0          /dev/did/rdsk/d20
21        theo:/dev/rdsk/c0t1d0          /dev/did/rdsk/d21
22        theo:/dev/rdsk/c0t6d0          /dev/did/rdsk/d22
```

Choosing Quorum and Resetting the `installmode` Attribute (Two-Node Cluster)

Before a new cluster can operate normally, you must reset the `installmode` attribute on all nodes. On a two node cluster where automatic quorum selection was disabled, the `installmode` will still be set on the cluster. You must choose a quorum device as a prerequisite to resetting `installmode`.

Choosing quorum Using the `scsetup` Utility

The `scsetup` utility is a menu-driven interface which, once the `installmode` flag is reset, turns into a general menu-driven alternative to low-level cluster commands. This is discussed more in Module 5.

The scsetup utility recognizes if the installmode flag is still set, and will not present any of its normal menus until you reset it. For a two-node cluster, this involves choosing a single quorum device first.

```
# /usr/cluster/bin/scsetup
>>> Initial Cluster Setup <<<

    This program has detected that the cluster "installmode" attribute is
    still enabled. As such, certain initial cluster setup steps will be
    performed at this time. This includes adding any necessary quorum
    devices, then resetting both the quorum vote counts and the
    "installmode" property.

    Please do not proceed if any additional nodes have yet to join the
    cluster.

    Is it okay to continue (yes/no) [yes]?  yes

    Do you want to add any quorum disks (yes/no) [yes]? yes

    Following are supported Quorum Devices types in Sun Cluster. Please
    refer to Sun Cluster documentation for detailed information on these
    supported quorum device topologies.

    What is the type of device you want to use?

        1) Directly attached shared disk
        2) Network Attached Storage (NAS) from Network Appliance

        q)

    Option: 1
>>> Add a SCSI Quorum Disk <<<

    A SCSI quorum device is considered to be any Sun Cluster supported
    attached storage which connected to two or more nodes of the cluster.
    Dual-ported SCSI-2 disks may be used as quorum devices in two-node
    clusters. However, clusters with more than two nodes require that
    SCSI-3 PGR disks be used for all disks with more than two
    node-to-disk paths.

    You can use a disk containing user data or one that is a member of a
    device group as a quorum device.

    For more information on supported quorum device topologies, see the
    Sun Cluster documentation.
    Is it okay to continue (yes/no) [yes]? yes

    Which global device do you want to use (d<N>)?  d4

    Is it okay to proceed with the update (yes/no) [yes]?  yes

scconf -a -q globaldev=d4

    Command completed successfully.


Press Enter to continue:

    Do you want to add another quorum device (yes/no) [yes]?  no

    Once the "installmode" property has been reset, this program will
    skip "Initial Cluster Setup" each time it is run again in the future.
    However, quorum devices can always be added to the cluster using the
    regular menu options. Resetting this property fully activates quorum
    settings and is necessary for the normal and safe operation of the
    cluster.

    Is it okay to reset "installmode" (yes/no) [yes]?  yes


scconf -c -q reset
scconf -a -T node=.

    Cluster initialization is complete.


    Type ENTER to proceed to the main menu:


  *** Main Menu ***

    Please select from one of the following options:

        1) Quorum
        2) Resource groups
        3) Data Services
        4) Cluster interconnect
        5) Device groups and volumes
        6) Private hostnames
        7) New nodes
        8) Other cluster properties

        ?) Help with menu options
        q) Quit
```

Choosing a Quorum Device (Clusters With More Than Two Nodes)

In a cluster of more than two nodes the `installmode` flag is *always* automatically reset, but the quorum device or devices are *never* automatically selected.

You should use `scsetup` to choose quorum devices, but the initial screens will look a little different since the `installmode` flag is already reset.

```
# /usr/cluster/bin/scsetup

  *** Main Menu ***

    Please select from one of the following options:

        1) Quorum
        2) Resource groups
        3) Data Services
        4) Cluster interconnect
        5) Device groups and volumes
        6) Private hostnames
        7) New nodes
        8) Other cluster properties

        ?) Help with menu options
        q) Quit

    Option: 1

  *** Quorum Menu ***

    Please select from one of the following options:

        1) Add a quorum device
        2) Remove a quorum device

        ?) Help
        q) Return to the Main Menu

    Option: 1
```

From here the dialog looks much like the previous example, except that the `installmode` is already reset so after adding your quorum devices you just return to the main menu.

Registering NAS Devices and Choosing a NAS Device as a Quorum Device

You saw options in both of the previous examples of the scsetup utility for choosing a NAS device as a quorum device.

Before you can use a NAS device as a quorum you must *register* the NAS device. Recall that in Sun Cluster 3.1 8/05 (Update 4) the only supported NAS device is a NetApp filer, and that you must also have the NTAPclnas package installed on each cluster node.

The following example shows using the scnas command to register a NAS device:

Registering a NAS device with the scnas command looks like the following:

```
# scnas -a -h netapps25 -t netapp -o userid=root
Please enter password:
```

Once the NAS device is registered, you can use the scsetup utility to establish one of the iSCSI LUNs on the device as a quorum. Recall that the iSCSI protocol is used on the NetApp filer NAS device *only* for the quorum mechanism. From the scsetup dialog the interaction would like like the following:

```
What is the type of device you want to use?

       1) Directly attached shared disk
       2) Network Attached Storage (NAS) from Network Appliance

       q)

       Option: 2
>>> Add a Netapp NAS Quorum Device <<<

   A Netapp NAS device can be configured as a quorum device for Sun
   Cluster. The NAS configuration data includes a device name, which is
   given by the user and must be unique across all quorum devices, the
   filer name, and a LUN id, which defaults to 0 if not specified.
   Please refer to the scconf(1M) man page and other Sun Cluster
   documentation for details.

   The NAS quorum device must be setup before configuring it with Sun
   Cluster. For more information on setting up Netapp NAS filer,
creating the device, and installing the license and the Netapp
   binaries, see the Sun Cluster documentation.

   Is it okay to continue (yes/no) [yes]? yes

   What name do you want to use for this quorum device?  netapps

   What is the name of the filer [netapps]?  netapps25

   What is the LUN id on the filer [0]? 0

   Is it okay to proceed with the update (yes/no) [yes]? yes

scconf -a -q name=netapps,type=netapp_nas,filer=netapps25,lun_id=0
```

Registering NAS Mounted Directories (for Data Fencing)

You use the scnasdir command to register on the NAS device the specific directories that are being used to serve cluster data. The Sun Cluster client implementation is then able to perform data fencing on these specific directories. In the NetApp Filer implementation, data fencing is accomplished by removing the name of a node from the NetApp Filer exports list as it is being fenced out of the cluster.

Registering the specific NAS directories for failure fencing looks like the following:

```
# scnasdir -r -h netapps25 -d /vol/vol_01_03
# scnasdir -r -h netapps25 -d /vol/vol_01_04
```

You can verify the configuration of the NAS device into the CCR using the -p option to the scnas or scnasdir commands as in the following example:

```
# scnas -p

Filers of type "netapp":

    Filer name:                netapps25
        type:                  netapp
        password:              *******
        userid:                root
        directories:           /vol/vol_01_03
        directories:           /vol/vol_01_04
```

Performing Post-Installation Verification

When you have completed the Sun Cluster software installation on all nodes, verify the following information:

- General cluster status
- Cluster configuration information

Verifying General Cluster Status

The `scstat` utility shows the current status of various cluster components. You can use it to display the following information:

- The cluster name and node names
- Names and status of cluster members
- Status of resource groups and related resources
- Cluster interconnect status

The following `scstat-q` command option shows the cluster membership and quorum vote information:

```
# scstat -q

-- Quorum Summary --

  Quorum votes possible:      3
  Quorum votes needed:        2
  Quorum votes present:       3


-- Quorum Votes by Node --

                    Node Name           Present Possible Status
                    ---------           ------- -------- ------
  Node votes:       vincent             1        1       Online
  Node votes:       theo                1        1       Online


-- Quorum Votes by Device --

                    Device Name         Present Possible Status
                    -----------         ------- -------- ------
  Device votes:     /dev/did/rdsk/d4s2  1        1       Online
```

Verifying Cluster Configuration Information

Cluster configuration information is stored in the CCR on each node. You should verify that the basic CCR values are correct. The `scconf -p` command shows general cluster information along with detailed information about each node in the cluster.

The following example shows the nature of the `scconf` output for a two node cluster following installation and post-installation. Some output is deleted.

```
# scconf -p
Cluster name:                                      orangecat
Cluster ID:                                        0x42B9DA63
Cluster install mode:                              disabled
Cluster private net:                               172.16.0.0
Cluster private netmask:                           255.255.0.0
Cluster new node authentication:                   unix
Cluster new node list:                             <. - Exclude all
nodes>
Cluster transport heart beat timeout:              10000
Cluster transport heart beat quantum:              1000
Cluster nodes:                                     vincent theo

Cluster node name:                                 vincent
  Node ID:                                         1
  Node enabled:                                    yes
  Node private hostname:                           clusternode1-priv
  Node quorum vote count:                          1
  Node reservation key:                            0x42B9DA6300000001
  Node transport adapters:                         hme0 qfe0

  Node transport adapter:                          hme0
    Adapter enabled:                               yes
    Adapter transport type:                        dlpi
    Adapter property:                              device_name=hme
    Adapter property:                              device_instance=0
    Adapter property:                              lazy_free=0
    Adapter property:                  dlpi_heartbeat_timeout=10000
    Adapter property:                  dlpi_heartbeat_quantum=1000
    Adapter property:                              nw_bandwidth=80
    Adapter property:                              bandwidth=10
    Adapter property:
netmask=255.255.255.128
    Adapter property:
ip_address=172.16.0.129
```

```
    Adapter port names:                           0

    Adapter port:                                 0
      Port enabled:                               yes

  Node transport adapter:                         qfe0
    Adapter enabled:                              yes
    Adapter transport type:                       dlpi
    Adapter property:                             device_name=qfe
    Adapter property:                             device_instance=0
    Adapter property:                             lazy_free=1
    Adapter property:                    dlpi_heartbeat_timeout=10000
    Adapter property:                    dlpi_heartbeat_quantum=1000
    Adapter property:                             nw_bandwidth=80
    Adapter property:                             bandwidth=10
    Adapter property:                            netmask=255.255.255.128
    Adapter property:                             ip_address=172.16.1.1
    Adapter port names:                           0

    Adapter port:                                 0
      Port enabled:                               yes

.
.
.
Information about the other node removed here for brevity
.
.

Cluster transport junctions:                      switch1 switch2

Cluster transport junction:                       switch1
  Junction enabled:                               yes
  Junction type:                                  switch
  Junction port names:                            1 2

  Junction port:                                  1
    Port enabled:                                 yes

  Junction port:                                  2
    Port enabled:                                 yes

Cluster transport junction:                       switch2
  Junction enabled:                               yes
  Junction type:                                  switch
  Junction port names:                            1 2
  Junction port:                                   1
    Port enabled:                                  yes

  Junction port:                                   2
    Port enabled:                                  yes


Cluster transport cables

                    Endpoint            Endpoint            State
                    --------            --------            -----
        Transport cable:   vincent:hme0@0      switch1@1 Enabled
        Transport cable:   vincent:qfe0@0      switch2@1 Enabled
        Transport cable:   theo:hme0@0         switch1@2 Enabled
        Transport cable:   theo:qfe0@0         switch2@2 Enabled


Quorum devices:                                    d4

Quorum device name:                                d4
  Quorum device votes:                             1
  Quorum device enabled:                           yes
  Quorum device name:                              /dev/did/rdsk/d4s2
  Quorum device hosts (enabled):                   vincent theo
  Quorum device hosts (disabled):
  Quorum device access mode:                       scsi2
```

Exercise: Installing the Sun Cluster Server Software

In this exercise, you complete the following tasks:

- Task 1 – Verifying the Boot Disk
- Task 2 – Verifying the Environment
- Task 3 – Updating the Name Service
- Task 4 – Installing the Sun Cluster Packages
- Task 5 – Configuring a New Cluster – All Nodes at Once Method
- Task 6 – Configuring a New Cluster – One Node at a Time Method
- Task 7 – Verifying an Automatically Selected Quorum Device (Two-Node Cluster)
- Task 8 – Configuring a Quorum Device (Three-Node Cluster, or Two-Node Cluster With No Automatic Selection)
- Task 9 – Verifying the Cluster Configuration and Status
- Task 10 – Testing Basic Cluster Operation

Preparation

Obtain the following information from your instructor if you have not already done so in a previous exercise:

Ask your instructor about the location of the Sun Cluster 3.1 software. Record the location.

Software location: ____________________________

Task 1 – Verifying the Boot Disk

Perform the following steps on all nodes to verify that the boot disks have a minimum 100 Mbyte `/globaldevices` partition, and a small partition for use by Solstice DiskSuite software replicas.

Perform the following steps:

1. Type the **`mount`** command, and record the logical path to the boot disk on each node (typically `/dev/dsk/c0t0d0`).

   Node 1 boot device: _______________

   Node 2 boot device: _______________

2. Type the **prtvtoc** command to verify each boot disk meets the Sun Cluster software partitioning requirements.

   ```
   # /usr/sbin/prtvtoc /dev/dsk/c0t0d0s2
   ```

Note – Append slice 2 (s2) to the device path. The sector count for the /globaldevices partition should be at least 200,000.

Task 2 – Verifying the Environment

Perform the following steps on all nodes:

1. Verify that the /.profile file on each cluster node contains the following environment variables:

   ```
   PATH=$PATH:/usr/cluster/bin

   MANPATH=$MANPATH:/usr/cluster/man:/usr/share/man

   TERM=dtterm
   EDITOR=vi
   export PATH MANPATH TERM EDITOR
   ```

Note – If necessary, create the .profile file as follows:
cp /etc/skel/local.profile /.profile.

2. If you edit the file, verify the changes are correct by logging out and in again as user root.
3. On all cluster nodes, edit the /etc/default/login file and comment out the CONSOLE=/dev/console line.

Task 3 – Updating the Name Service

Perform the following steps to update the name service:

1. Edit the /etc/hosts file on the administrative workstation and all cluster nodes, and add the IP addresses and host names of the administrative workstation and cluster nodes.
2. If you are using a naming service, add the IP addresses and host names to the name service.

Note – Your lab environment might already have all of the IP addresses and host names entered in the `/etc/hosts` file.

Task 4 – Installing the Sun Cluster Packages

Perform the following steps on all nodes. You will save time if you perform the installation simultaneously in all nodes (you will have to manage more than one GUI install, but that is why you are in an advanced class!)

Note – These instructions have you do a graphical install. Feel free to try the non-graphical install (`./installer -nodisplay`), or do graphical installs on some nodes and non-graphical installs on other nodes to compare the two.

1. On your administrative workstation or display, figure out the number of your display. In the RLDC environment, each student or group may have a different display number. The output may look like `unix:6.0` or maybe just like `:9.0`. In either case the display number is the number between the colon and the `.0`:

```
(# or $) echo $DISPLAY
```

2. On the administrative workstation or display, enable remote X-windows display from the nodes:

```
(# or $) /usr/openwin/bin/xhost +nodename1
(# or $) /usr/openwin/bin/xhost +nodename2
(# or $) /usr/openwin/bin/xhost +nodename3
```

3. On your cluster node, set and export the `DISPLAY`, if it is not already set:

```
# env|grep DISPLAY
# DISPLAY=display-name-or-IP:display-#
# export DISPLAY
```

4. Run the installer:

```
# cd sc_framework_location/Solaris_sparc
# ./installer
```

   a. Accept the license agreement.

b. Choose any additional languages (each additional language can add significant time to your install).

c. Choose *only* the Sun Cluster 3.1 8/05 (*not* the Agents).

d. Verify that system requirements are met.

e. Choose `Configure Later.`

f. Continue with the installation. The “shared components” will be installed.

g. Deselect Product Registration (on the GUI only) and continue with the package installation (this is automatic in the non-GUI install).

Task 5 – Configuring a New Cluster – All Nodes at Once Method

If you prefer to do the “one-node-at-a-time” method skip this task and do Task 6 instead.

Perform the following steps to configure the entire cluster using the “all nodes at once” method.

1. Log in as user `root` on the node that you want to have assigned the *highest* node ID in the cluster. This is the node you will drive from.
2. Start the `scinstall` utility(`/usr/cluster/bin/scinstall).`
3. As the installation proceeds, make the following choices:

   a. From the Main Menu choose Option 1, Establish a new cluster.

   b. From the Install Menu choose Option 1, Install all nodes of a new cluster.

   c. From the Type of Installation Menu choose Option 1, Typical.

   d. Furnish your assigned cluster name.

   e. Enter the names of the other nodes in your cluster. They will be rebooted and assigned Node ID’s in *reverse order* to what you type.

   f. Select the adapters that will be used for the cluster transport. If you are asked (with a tagged VLAN-capable adapter) if it is a dedicated cluster transport adapter, answer “yes”.

   g. For a two-node cluster, do not disable the automatic quorum selection (answer “No”).

4. You may observe the following error messages as each node reboots. They are all normal.

```
devfsadm: minor_init failed for module
/usr/lib/devfsadm/linkmod/SUNW_scmd_link.so
/usr/cluster/bin/scdidadm: Could not load DID instance list.
Cannot open /etc/cluster/ccr/did_instances.
Booting as part of a cluster
ip: joining multicasts failed (18) on clprivnet0 - will use link layer
broadcasts for multicast
t_optmgmt: System error: Cannot assign requested address
```

Task 6 – Configuring a New Cluster – One Node at a Time Method

If you chose to install your cluster in "Task 5 – Configuring a New Cluster – All Nodes at Once Method" do not do this task.

Perform the following steps to configure the first node in your cluster (the one that will be assigned node ID 1). You must wait for this node to complete and reboot into the cluster before configuring other nodes

1. Start the `scinstall` utility (`/usr/cluster/bin/scinstall`).
2. As the installation proceeds, make the following choices:
   a. Choose Option 1 from the Main Menu, Establish a new cluster.
   b. Choose Option 2 from the Install Menu, Install just this machine as the first node of a new cluster.
   c. From the Type of Installation Menu choose Option 1, Typical.
   d. Furnish your assigned cluster name.
   e. Allow `sccheck` to run.
   f. Furnish the name of the other nodes to be added later.
   g. Verify the list of node names.
   h. Select the transport adapters. If you are asked (with a tagged VLAN-capable adapter) if it is a dedicated cluster transport adapter, answer "yes."
   i. For a two node cluster, do not disable the automatic quorum selection (answer "No").
   j. Reply **yes** to the automatic reboot question.

k. Examine the `scinstall` command options for correctness. Accept them if they seem appropriate.

   You must wait for this node to complete rebooting to proceed to the second node.

3. You may observe the following error messages during the node reboot. They are all normal.

```
devfsadm: minor_init failed for module
/usr/lib/devfsadm/linkmod/SUNW_scmd_link.so
/usr/cluster/bin/scdidadm: Could not load DID instance list.
Cannot open /etc/cluster/ccr/did_instances.
Booting as part of a cluster
ip: joining multicasts failed (18) on clprivnet0 - will use link layer
broadcasts for multicast
t_optmgmt: System error: Cannot assign requested address
```

Perform the following steps on each additional node in the cluster. If you have more than two nodes, you could do all the other nodes simultaneously, although then it will be hard to predict which node gets which node ID. If you want this to be predictable, do one node at a time.

1. Start the `scinstall` utility on the new node.
2. As the installation proceeds, make the following choices:

   a. Choose Option 1 from the Main Menu.

   b. Choose Option 3 from the Install Menu, Add this machine as a node in an established cluster.

   c. From the Type of Installation Menu choose Option 1, Typical.

   d. Provide the name of a sponsoring node.

   e. Provide the name of the cluster you want to join.

      Type **`scconf -p | more`** on the first node (the sponsoring node) if you have forgotten the name of the cluster.

   f. Use auto-discovery for the transport adapters.

   g. Reply **`yes`** to the automatic reboot question.

   h. Examine and approve the `scinstall` command line options.

   i. You will see the same normal boot error messages on the additional nodes as on the first node.

Note – You see interconnect-related errors on the existing nodes until the new node completes the first portion of its reboot operation.

Task 7 – Verifying an Automatically Selected Quorum Device (Two-Node Cluster)

Perform the following steps to verify automatic quorum selection on a two-node cluster:

1. In a two node cluster, assuming you have allowed automatic quorum selection, wait after the second node boots until you see console messages (on all nodes) indicating that the quorum device has been selected.
2. On either node, type the **`scstat -q`** command. Verify that your quorum device was chosen and that you have the expected number of quorum votes.

Task 8 – Configuring a Quorum Device (Three-Node Cluster, or Two-Node Cluster With No Automatic Selection)

Perform the following steps to do quorum selection on a three-node cluster:

1. In a three node-cluster, wait after the third node boots until you see console messages (on all nodes) indicating that the quorum votes are being reset.
2. On Node 1, type the **`scdidadm -L`** command, and record the DID devices you intend to configure as quorum disks. For example, if you had three nodes in a "Pair + 1" configuration, you would want two quorum devices.

   Quorum disks: _______ (`d4`, `d6`, and so on)

Note – Pay careful attention. The first few DID devices might be local disks, such as the boot disk and a CD-ROM (target 6). Examine the standard logical path to make sure the DID device you select is a disk in a storage array and is connected to more than one node.

3. On Node 1, type the **`scsetup`** command. Navigate to the section for adding a new quorum device and supply the name of the first DID device (global device) you selected in the previous step. You should see output similar to the following.

   ```
   scconf -a -q globaldev=d12
   May  3 22:29:13 vincent cl_runtime: NOTICE: CMM:
   Cluster members: vincent theo apricot.
   May  3 22:29:13 proto192 cl_runtime: NOTICE: CMM: node
   reconfiguration #4 completed.
   ```

4. If you have three nodes in a "Pair +N" configuration, for example, you should add a second quorum device.
5. Reply **`yes`** to the reset installmode question (two-node cluster only).

   You should see a "Cluster initialization is complete" message.
6. Type **`q`** to quit the `scsetup` utility.

Task 9 – Verifying the Cluster Configuration and Status

Perform the following steps on all nodes to verify the completed cluster installation:

1. On all nodes, type the **`scstat -q`** command.

   You should see the correct node and disk quorum votes

2. On all nodes, type the **`scstat -W`** command.

   You should see active redundant interconnect paths between all pairs of nodes.

3. On all nodes, type the **`scdidadm -L`** command.

   Each shared (dual-ported) DID device should show a logical path from each cluster node.

4. On either node, type the **`scconf -p`** command.

   The cluster status, node names, transport configuration, and quorum device information should be complete.

Task 10 – Testing Basic Cluster Operation

Perform the following steps to verify the basic cluster software operation:

Note – You are using a command that has not yet been presented in the course. If you have any questions, consult with your instructor.

1. Ensure that you are still connected to the console of all your cluster nodes.
   - If you are connected, go to the next step.
   - If you are not connected, connect to all nodes with the `cconsole` *`clustername`* command.
2. Log in to one of your cluster host systems as user `root`.
3. On one node, issue the command to shut down all cluster nodes.

```
# scshutdown -y -g 15
```

Note – The `scshutdown` command completely shuts down all cluster nodes, including the Solaris OS.

4. Ensure that all nodes of your cluster are at the OpenBoot programmable read-only memory (PROM) ok prompt. Consult your instructor if this is not the case.

5. Boot each node in turn.

   Each node should come up and join the cluster. Consult your instructor if this is not the case.

6. Verify the cluster status.

   ```
   # scstat
   ```

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 5

Performing Basic Cluster Administration

Objectives

Upon completion of this module, you should be able to:

- Identify the cluster daemons
- Verify cluster status from the command line
- Perform basic cluster startup and shutdown operations, including booting nodes in non-cluster mode and placing nodes in a maintenance state
- Describe the Sun Cluster administration utilities

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- What must be monitored in the Sun Cluster software environment?
- How current does the information need to be?
- How detailed does the information need to be?
- What types of information are available?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)

Identifying Cluster Daemons

When a cluster node is fully booted into a cluster, there are several cluster daemons that will be added to the traditional Solaris OS.

None of these daemons require any manual maintenance, regardless of which version of Solaris OS you are running on. However, behind the scenes, the daemons are managed in a slightly different way in Solaris 10 OS.

- Solaris 8 OS and Solaris 9 OS – Daemons are launched by traditional Solaris OS boot scripts (and are thus guaranteed to be running by the time you get the console login prompt after a boot). A special cluster specific daemon monitor, `rcp.pmfd`, is required for restarting some daemons.
- Solaris 10 OS – Daemons are launched by the Solaris 10 OS Service Management Facility (SMF). At boot time, therefore, you may see a console login prompt before many of these daemons are launched. SMF itself can restart some daemons.

The following is taken from a cluster node running Solaris 10 OS:

```
root      4     0   0 14:40:14 ? 1:08 cluster
root    218     1   0 14:41:04 ? 0:00 /usr/cluster/lib/sc/failfastd
root    220     1   0 14:41:04 ? 0:00 /usr/cluster/lib/sc/clexecd
root   2287     1   0 14:43:03 ? 0:00 /usr/cluster/lib/sc/cl_eventd
root    208     1   0 14:41:01 ? 0:00 /usr/cluster/lib/sc/qd_userd
root   2563     1   0 14:43:09 ? 0:04 /usr/cluster/lib/sc/rgmd
root   2347     1   0 14:43:04 ? 0:00 /usr/cluster/lib/sc/sparcv9/rpc.pmfd
root   2295     1   0 14:43:03 ? 0:01 /usr/cluster/bin/pnmd
root   2376     1   0 14:43:05 ? 0:00 /usr/cluster/lib/sc/cl_eventlogd
root   2291     1   0 14:43:03 ? 0:15 /usr/cluster/lib/sc/scdpmd
root   2269     1   0 14:43:02 ? 0:01 /usr/cluster/lib/sc/rpc.fed

root   2293     1   0 14:43:03 ? 0:03 /usr/cluster/lib/sc/cl_ccrad
```

- `cluster` – This is a system process (created by the kernel) to encapsulate the kernel threads that make up the core kernel range of operations.

  There is no way to kill this process (even with a `KILL` signal), because it is always in the kernel.

- `failfastd` – This daemon is the failfast proxy server. Other daemons that require services of the failfast device driver register with `failfastd`. The failfast daemon allows the kernel to panic if certain essential daemons have failed.

- `clexecd` – This is used by cluster kernel threads to execute `userland` commands (such as the `run_reserve` and `dofsck` commands). It is also used to run cluster commands remotely (like the `scshutdown` command).

  This daemon registers with `failfastd` so that a failfast device driver will panic the kernel if this daemon is killed and not restarted in 30 seconds.

- `cl_eventd` – This daemon registers and forwards cluster events (such as nodes entering and leaving the cluster). With Sun Cluster 3.1 10/03 or higher software, user applications can register themselves to receive cluster events.

  The daemon automatically gets respawned if it is killed.

- `qd_userd` – This daemon serves as a proxy whenever any quorum device activity requires execution of some command in userland (for example, a NAS quorum device).

  If you kill this daemon, you must restart it manually.

- `rgmd` – This is the resource group manager, which manages the state of all cluster-unaware applications.

  A `failfast` driver panics the kernel if this daemon is killed and not restarted in 30 seconds.

- `rpc.fed` – This is the "fork-and-exec" daemon, which handles requests from `rgmd` to spawn methods for specific data services.

  A `failfast` driver panics the kernel if this daemon is killed and not restarted in 30 seconds.

- `rpc.pmfd` – This is the process monitoring facility. It is used as a general mechanism to initiate restarts and failure action scripts for some cluster framework daemons (in Solaris 8 OS and Solaris 9 OS), and for most application daemons and application fault monitors (in Solaris 8, 9, and 10 OS)

A failfast driver panics the kernel if this daemon is stopped and not restarted in 30 seconds.

- pnmd – This is the public network management daemon, which manages network status information received from the local IPMP running on each node and facilitates application failovers caused by complete public network failures on nodes.

  It is automatically restarted if it is stopped.

- cl_eventlogd – This daemon logs cluster events into a binary log file. At the time of writing for this course, there is no published interface to this log.

  It is automatically restarted if it is stopped.

- cl_ccrad – This daemon provides access from userland management applications to the CCR.

  It is automatically restarted if it is stopped.

- scdpmd – This daemon monitors the status of disk paths, so that they can be reported in the output of the scdpm command).

  It is automatically restarted if it is stopped.

Using Cluster Status Commands

There are several cluster status commands. Some of the commands have uses other than status reporting. Many of the commands, or specific options to the commands, must be run as the user `root` or with cluster RBAC authorizations. See Appendix C, "Role-Based Access Control Authorizations" for more information about RBAC and the Sun Cluster 3.1 RBAC authorizations.

Checking Status Using the `scstat` Command

The `scstat` utility displays the current state of the Sun Cluster 3.1 software and its components. Run `scstat` on any one node. When run without any options, `scstat` displays the status for all components of the cluster. This display includes the following information:

- A list of cluster members
- The status of each cluster member
- The status of resource groups and resources
- The status of every path on the cluster interconnect
- The status of every disk device group
- The status of every quorum device
- The status of every Internet Protocol Network Multipathing group and public network adapter

Viewing the Cluster Configuration

Display information about the overall cluster configuration with the `scconf` command. You do not need to be the user `root` to run this form of the command. For additional information, include the verbose options.

```
# scconf -p
Cluster name:                                     orangecat
Cluster ID:                                       0x42B9DA63
Cluster install mode:                             disabled
Cluster private net:                              172.16.0.0
Cluster private netmask:                          255.255.0.0
Cluster new node authentication:                  unix
Cluster new node list:                            <. - Exclude all
nodes>
```

```
Cluster transport heart beat timeout:              10000
Cluster transport heart beat quantum:              1000
Cluster nodes:                                     vincent theo

Cluster node name:                                 vincent
  Node ID:                                         1
  Node enabled:                                    yes
  Node private hostname:                           clusternode1-priv
  Node quorum vote count:                          1
  Node reservation key:                            0x42B9DA6300000001
  Node transport adapters:                         hme0 qfe0

  Node transport adapter:                          hme0
    Adapter enabled:                               yes
    Adapter transport type:                        dlpi
    Adapter property:                              device_name=hme
    Adapter property:                              device_instance=0
    Adapter property:                              lazy_free=0
    Adapter property:                   dlpi_heartbeat_timeout=10000
    Adapter property:                   dlpi_heartbeat_quantum=1000
    Adapter property:                              nw_bandwidth=80
    Adapter property:                              bandwidth=10
    Adapter property:
netmask=255.255.255.128
    Adapter property:
ip_address=172.16.0.129
    Adapter port names:                            0

    Adapter port:                                  0
      Port enabled:                                yes

  Node transport adapter:                          qfe0
    Adapter enabled:                               yes
    Adapter transport type:                        dlpi
    Adapter property:                              device_name=qfe
    Adapter property:                              device_instance=0
    Adapter property:                              lazy_free=1
    Adapter property:                     dlpi_heartbeat_timeout=10000
    Adapter property:                     dlpi_heartbeat_quantum=1000
    Adapter property:                              nw_bandwidth=80
    Adapter property:                              bandwidth=10
    Adapter property:                             netmask=255.255.255.128
    Adapter property:                              ip_address=172.16.1.1
    Adapter port names:                            0

    Adapter port:                                  0
```

```
      Port enabled:                                yes

.
.
.
Information about the other node removed here for brevity
.
.

Cluster transport junctions:                       switch1 switch2

Cluster transport junction:                        switch1
  Junction enabled:                                yes
  Junction type:                                   switch
  Junction port names:                             1 2

  Junction port:                                   1
    Port enabled:                                  yes

  Junction port:                                   2
    Port enabled:                                  yes

Cluster transport junction:                        switch2
  Junction enabled:                                yes
  Junction type:                                   switch
  Junction port names:                             1 2

  Junction port:                                   1
    Port enabled:                                  yes

  Junction port:                                   2
    Port enabled:                                  yes


Cluster transport cables

                    Endpoint            Endpoint            State
                    --------            --------            -----
        Transport cable:   vincent:hme0@0      switch1@1 Enabled
        Transport cable:   vincent:qfe0@0      switch2@1 Enabled
        Transport cable:   theo:hme0@0         switch1@2 Enabled
        Transport cable:   theo:qfe0@0         switch2@2 Enabled


Quorum devices:                                    d4
Quorum device name:                                   d4
  Quorum device votes:                                1
  Quorum device enabled:                              yes
  Quorum device name:                                 /dev/did/rdsk/d4s2
  Quorum device hosts (enabled):                      vincent theo
  Quorum device hosts (disabled):
  Quorum device access mode:                          scsi2
```

Validating a Basic Cluster Configuration

The `sccheck` utility examines Sun Cluster 3.1 nodes for known vulnerabilities and configuration problems. It also delivers reports that describe all failed checks, if any. The reports are placed into the `/var/cluster/sccheck` directory by default. The `sccheck` utility should be run any time configuration changes are performed.

The utility runs one of the following two sets of checks, depending on the state of the node that issues the command:

- Preinstallation checks – When issued from a node that is not running as an active cluster member, the `sccheck` utility runs preinstallation checks on that node. These checks ensure that the node meets the minimum requirements to be successfully configured with Sun Cluster 3.1 software.
- Cluster configuration checks – When issued from an active member of a running cluster, the `sccheck` utility runs configuration checks on the specified or default set of nodes. These checks ensure that the cluster meets the basic configuration required for a cluster to be functional. The `sccheck` utility produces the same results for this set of checks, regardless of which cluster node issues the command.

The `sccheck` utility runs configuration checks and uses the `explorer`(1M) utility to gather system data for check processing. The `sccheck` utility first runs single-node checks on each *nodename* specified, then runs multiple-node checks on the specified or default set of nodes.

The `sccheck` command runs in two steps, data collection and analysis. Data collection can be time consuming, depending on the system configuration. You can invoke `sccheck` in verbose mode with the `-v1` flag to print progress messages, or you can use the `-v2` flag to run `sccheck` in highly verbose mode, which prints more detailed progress messages, especially during data collection.

Each configuration check produces a set of reports that are saved. For each specified nodename, the sccheck utility produces a report of any single-node checks that failed on that node. Then, the node from which sccheck was run produces an additional report for the multiple-node checks. Each report contains a summary that shows the total number of checks executed and the number of failures, grouped by check severity level.

Each report is produced in both ordinary text and in Extensible Markup Language (XML). The reports are produced in English only.

The sccheck command can be run with or without options. Running the command from an active cluster node without options causes it to generate reports on all the active cluster nodes and to place the reports into the default reports directory.

Disk Path Monitoring

Disk path monitoring (DPM) administration commands enable you to receive notification of disk-path failure. The scdpm command manages the disk-path monitoring daemon, /usr/cluster/lib/sc/scdpmd, in the Sun Cluster 3.1 environment. This command is used to monitor and unmonitor disk paths. You can also use the scdpm command to display the status of disk paths. All of the accessible disk paths in the cluster or on a specific node are printed to the standard output. The scdpm command must be run from a cluster node that is online in cluster mode. You can specify either a global name or a UNIX name when you monitor a new disk path. Additionally, you can force the daemon to reread the entire disk configuration.

The disk path is represented by a node name and a disk name. The node name must be the hostname or the word all to address all of the nodes in the cluster. The disk name must be the global disk name, a UNIX path name, or the word all to address all the disks in the node. The disk name can be either the full global path name or just the disk name, for example, /dev/did/dsk/d3 or d3. The disk name can also be the full UNIX path name, for example, /dev/rdsk/c0t0d0s0.

Disk path status changes are logged into /var/adm/messages with the syslogd LOG_INFO facility level. All failures are logged by using the LOG_ERR facility level.

The following command monitors a new disk path. In the following example, all nodes monitor /dev/did/dsk/d3 where this path is valid.

```
    # scdpm -m /dev/did/dsk/d3
```

The following command monitors new paths on a single node. The daemon on the node named vincent monitors paths to the /dev/did/dsk/d4 and /dev/did/dsk/d5 disks.

```
    # scdpm -m vincent:d4 -m vincent:d5
```

The following command prints all disk paths in the cluster and their status.

```
    # scdpm -p all:all
```

Checking Status Using the scinstall Utility

During the Sun Cluster software installation, the scinstall utility is copied into the /usr/cluster/bin directory. You can run the scinstall utility with options that display the Sun Cluster software revision, the names and revision of installed packages, or both. The displayed information is for the local node only. This is the only form of scinstall that you can run as a non-root user. A typical scinstall status output follows.

```
vincent:/# scinstall -pv
Sun Cluster 3.1u4 for Solaris 10 sparc
SUNWscr:       3.1.0,REV=2005.05.20.19.16
SUNWscu:       3.1.0,REV=2005.05.20.19.16
SUNWscsckr:    3.1.0,REV=2005.05.20.19.16
SUNWscscku:    3.1.0,REV=2005.05.20.19.16
SUNWscnmr:     3.1.0,REV=2005.05.20.19.16
SUNWscnmu:     3.1.0,REV=2005.05.20.19.16
SUNWscdev:     3.1.0,REV=2005.05.20.19.16
SUNWscgds:     3.1.0,REV=2005.05.20.19.16
SUNWscman:     3.1.0,REV=2005.05.20.19.16
SUNWscsal:     3.1.0,REV=2005.05.20.19.16
SUNWscsam:     3.1.0,REV=2005.05.20.19.16
SUNWscvm:      3.1.0,REV=2005.05.20.19.16
SUNWmdmr:      3.1.0,REV=2005.05.20.19.16
SUNWmdmu:      3.1.0,REV=2005.05.20.19.16
SUNWscmasar:   3.1.0,REV=2005.05.20.19.16
SUNWscmasau:   3.1.0,REV=2005.05.20.19.16
SUNWscva:      3.1.0,REV=2005.05.20.19.16
SUNWscspm:     3.1.0,REV=2005.05.20.19.16
```

```
SUNWscspmu:     3.1.0,REV=2005.05.20.19.16
SUNWscspmr:     3.1.0,REV=2005.05.20.19.16
```

Caution – Use the `scinstall` utility carefully. It is possible to create serious cluster configuration errors using the `scinstall` utility.

Examining the `/etc/cluster/release` File

The `/etc/cluster/release` file gives you specific information about the release of the Sun Cluster software framework that is installed on your node:

```
# cat /etc/cluster/release
        Sun Cluster 3.1u4 for Solaris 10 sparc
Copyright 2005 Sun Microsystems, Inc. All Rights Reserved.
```

Controlling Clusters

Basic cluster control includes starting and stopping clustered operation on one or more nodes and booting nodes in non-cluster mode.

Starting and Stopping Cluster Nodes

The Sun Cluster software starts automatically during a system boot operation. Use the standard `init` or `shutdown` command to shut down a single node. You use the `scshutdown` command to shut down all nodes in the cluster.

Before shutting down a node, you should switch resource groups to the next preferred node and then run `shutdown` or `init` on the node.

Note – After an initial Sun Cluster software installation, there are no configured resource groups with which to be concerned.

Shutting Down a Cluster

You can shut down the entire cluster with the `scshutdown` command from any active cluster node. Once your cluster is in production and running clustered applications, you will have a goal of *never* having to do this. The whole purpose of the Sun Cluster environment is that you should always be able to keep at least one node running.

```
# scshutdown -y -g 30
/etc/rc0.d/K05stoprgm: Calling scswitch -S (evacuate)
svc.startd: The system is coming down.  Please wait.
svc.startd: 88 system services are now being stopped.
Jun 23 13:29:49 vincent rpcbind: rpcbind terminating on
signal.
Jun 23 13:29:51 vincent syslogd: going down on signal 15
Jun 23 13:29:53 cl_eventlogd[2110]: Going down on signal
15.
Jun 23 13:29:54 Cluster.PNM: PNM daemon exiting.
Jun 23 13:29:54 Cluster.scdpmd: going down on signal 15
svc.startd: The system is down.
syncing file systems... done
WARNING: CMM: Node being shut down.
Program terminated
{0} ok
```

Booting Nodes in Non-Cluster Mode

Occasionally, you might want to boot a node without it joining the cluster. This might be to debug some sort of problem preventing a node from joining a cluster, or to perform maintenance. For example, you upgrade the cluster software itself when a node is booted into maintenance mode. Other nodes may still be up running your clustered applications.

To other nodes that are still booted into the cluster, a node that is booted into non-cluster node looks like it has failed completely. It can not be reached across the cluster transport.

```
ok boot -x
Rebooting with command: boot -x
Boot device: /pci@1f,4000/scsi@3/disk@0,0:a  File and args: -x
SunOS Release 5.10 Version Generic 64-bit
Copyright 1983-2005 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Hostname: vincent
Not booting as part of a cluster

vincent console login:
Jun 23 13:32:54 vincent xntpd[410]: couldn't resolve `clusternode1-priv',
giving up on it
Jun 23 13:32:54 vincent xntpd[410]: couldn't resolve `clusternode2-priv',
giving up on it
```

Placing Nodes in Maintenance Mode

If you anticipate a node will be down for an extended period, you can place the node in a maintenance state from an active cluster node. The maintenance state disables the node's quorum vote. You cannot place an active cluster member in a maintenance state. A typical command is as follows:

```
# scconf -c -q node=theo,maintstate
```

The scstat command shows that the possible vote for theo is now set to 0.

In addition, the vote count for any SCSI-2 (dual ported) quorum device physically attached to the node is also set to 0.

You can reset the maintenance state for a node by rebooting the node into the cluster. The node and any directly-attached SCSI-2 quorum devices regain their votes.

Maintenance Mode Example

To see the value of placing a node in maintenance state, consider the following topology (shown in Figure 5-1), which was described in Module 3, "Preparing for Installation and Understanding Quorum Devices."

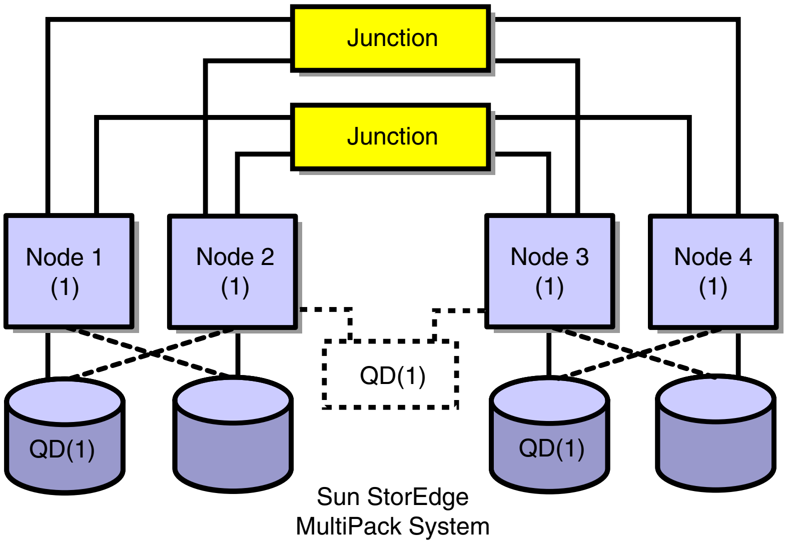


Figure 5-1 Clustered-Pair Quorum Devices

Now, imagine Nodes 3 and 4 are down. The cluster can still continue with Nodes 1 and 2, because you still have four out of the possible seven quorum votes.

Now imagine you wanted to halt Node 1 and keep Node 2 running as the only node in the cluster. If there were no such thing as maintenance state, this would be impossible. Node 2 would panic with only three out of seven quorum votes. But if you put Nodes 3 and 4 into maintenance state, you reduce the total possible votes to three and Node 2 can continue with two out of three votes.

Cluster Administration Utilities

There are three general approaches to performing cluster administration tasks. These are the following:

- Use the low-level administration commands.
- Use the `scsetup` utility.
- Use the SunPlex Manager web-based GUI.

Low-Level Administration Commands

There are a surprisingly small number of commands used to administer the cluster, but some of the commands have a large number of options. The commands are the following:

- `scconf` – For configuration of hardware and device groups
- `scrgadm` – For configuration of resource groups
- `scswitch` – For switching device groups and resource groups
- `scstat` – For printing the status of anything
- `scdpm` – For monitoring disk paths

As this course presents device groups and resource groups in the following four modules, it will focus on the using the low-level commands for administration. Seeing and understanding all the proper options to these commands is the best way to understand the capability of the cluster.

Using the `scsetup` Command

`The scsetup` command is a menu-driven utility meant to guide you through a dialogue, rather than you having to remember the options to the commands listed in the previous section. The `scsetup` utility does all of its actual work by calling the previously-listed commands. It also always shows the lower-level commands as it runs them, so it has educational value as well.

```
# scsetup
    *** Main Menu ***

        Please select from one of the following options:

            1) Quorum
            2) Resource groups
            3) Data services
            4) Cluster interconnect
            5) Device groups and volumes
            6) Private hostnames
            7) New nodes
            8) Other cluster properties

            ?) Help with menu options
            q) Quit

        Option:
```

Comparing Low-Level Command and `scsetup` Usage

There are many examples of the usage of the commands in the sections about device groups and resource groups in the following four modules.

As a comparison, assume your task was to add a new cluster transport. You have a third private transport physically connected in a two-node cluster, without switches, and you need to configure it into the cluster.

Note that the `scsetup` utility does not save you from needing the knowledge about "how things are done" in the Sun Cluster software environment. To configure a transport into Sun Cluster software without a switch, you must do the following:

1. Add a transport adapter definition on each node.
2. Add a transport cable definition between them.

Now, given *that* knowledge alone and without knowing any options to the scconf command, you could easily follow the menus of the scsetup command to accomplish the task. You choose Menu Option 4, Cluster interconnect, to see the following:

```
*** Cluster Interconnect Menu ***

    Please select from one of the following options:

        1) Add a transport cable
        2) Add a transport adapter to a node
        3) Add a transport junction
        4) Remove a transport cable
        5) Remove a transport adapter from a node
        6) Remove a transport junction
        7) Enable a transport cable
        8) Disable a transport cable

        ?) Help
        q) Return to the Main Menu

    Option:
```

Menu Option 2 guides you through adding the adapters, and then Menu Option 1 guides you through adding the cables.

If you were to use the scconf command "by hand" to accomplish the same tasks, the commands you would need to run would look like the following:

```
# scconf -a -A trtype=dlpi,name=qfe0,node=vincent
# scconf -a -A trtype=dlpi,name=qfe0,node=theo
# scconf -a -m endpoint=vincent:qfe0,endpoint=theo:qfe0
```

These are the same commands eventually called by scsetup.

SunPlex Manager

The SunPlex Manager is a web-based administrative tool for the Sun Cluster environment.

Starting in Sun Cluster 3.1 9/04 (Update 3), SunPlex Manager runs underneath the Sun Java Web Console. Sun Java Web Console is a single sign-on interface to many Solaris OS applications that are implemented as web back-ends. When you log in to the Sun Java Web Console, you may see other web applications available besides the SunPlex Manager.

The Java ES installer that is used to install the Sun Cluster framework packages automatically installs the Sun Java Web Console, if needed (in Solaris 10 OS, for example, it is already part of the base OS if you did a full install), and the Sun Plex Manager.

You can use any modern Java enabled web-browser to access the Sun Java Web Console and the SunPlex Manager.

Usage of the SunPlex Manager is much like the usage of `scsetup` in that:

- You still really need to understand the “nature” of the cluster tasks before accomplishing anything.
- It saves you from having to remember options to the lower level commands.
- It accomplishes all its actions through the lower level commands, and shows the lower level commands as it runs them.

SunPlex Manager has the additional benefit of offering some graphical topology views of the Sun Cluster software nodes with respect to device groups, resource groups, and the like.

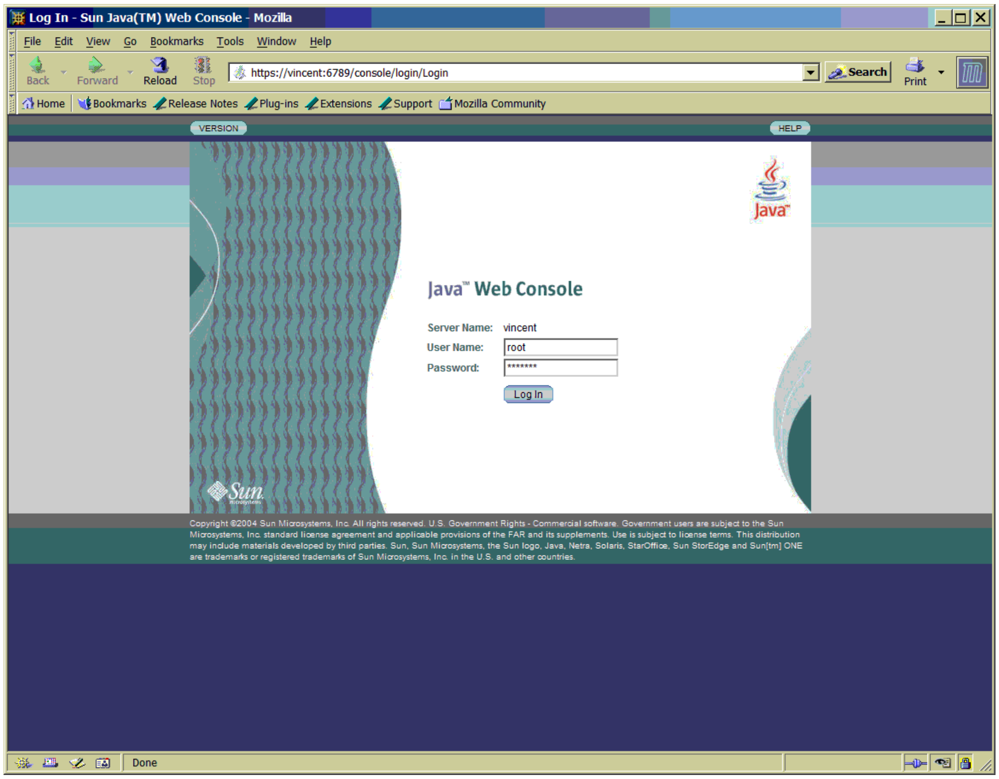
Many of the exercises in the modules following this one offer you the opportunity to view and manage device and resource groups through SunPlex Manager. However, this course will not focus on the details of using SunPlex Manager as the method of accomplishing the task. Like `scsetup`, after you understand the nature of a task, SunPlex Manager can guide you through it. This is frequently easier than researching and typing all the options correctly to the lower level commands.

Logging In to the Web Console

Use the following URL on a web browser enabled with Java technology to access the Sun Java Web Console.

```
https://nodename:6789
```

Figure 5-2 shows the Web Console login window. Cookies must be enabled on your web browser to allow Sun Java Web Console login and usage.


Log In - Sun Java(TM) Web Console - Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
Search
Print
Home Bookmarks Release Notes Plug-ins Extensions Support Mozilla Community
VERSION
HELP
Java
Java™ Web Console
Log In
Sun Microsystems
Done

https://vincent:6789/console/login/Login

Server Name: vincent
User Name: root
Password: *******

Copyright ©2004 Sun Microsystems, Inc. All rights reserved. U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms. This distribution may include materials developed by third parties. Sun, Sun Microsystems, the Sun logo, Java, Netra, Solaris, StarOffice, Sun StorEdge and Sun[tm] ONE are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

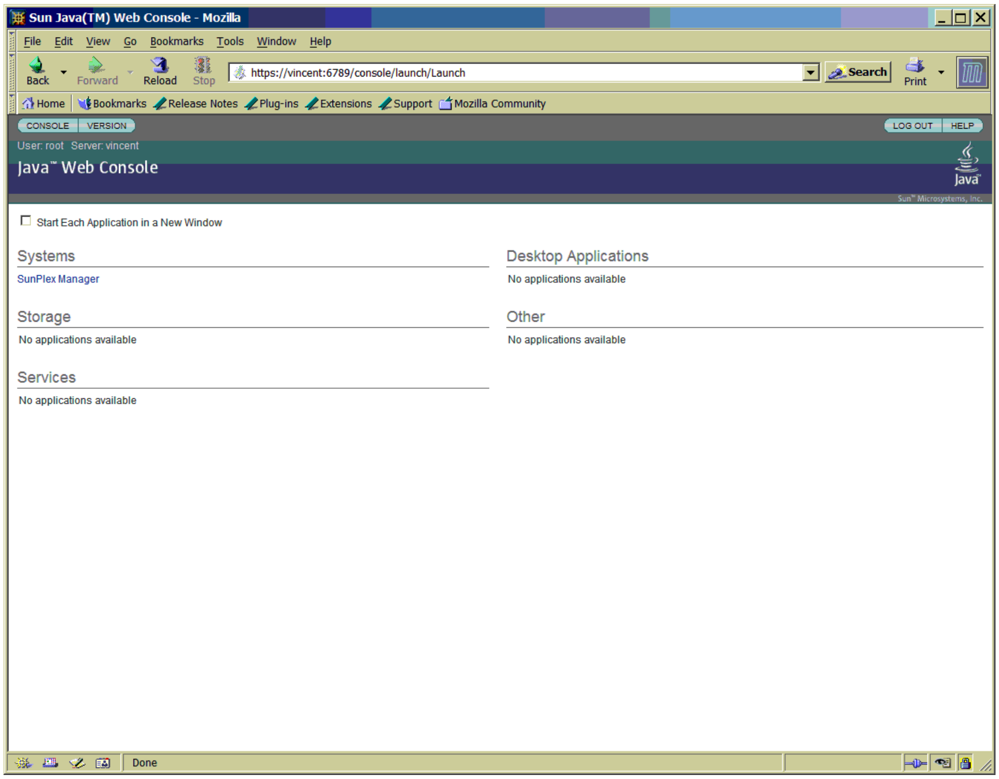
Figure 5-2 Sun Java Web Console Login Window

Log in with the user name of `root` and the password of root.

Alternatively, you can create a non-root user or role authorized to have full or limited access through RBAC. By default, any user will be able to log in to SunPlex Manager through the Sun Java Web Console and have view-only access. RBAC authorizations for Sun Cluster are discussed further in Appendix C, "Role-Based Access Control Authorizations."

Accessing SunPlex Manager

The SunPlex Manager application is always available from within the Sun Java Web Console. You may have other administrative applications available as well. You can choose to open the SunPlex Manager in its own browser window or in the same window. Figure 5-3 shows the screen you access after logging in to the Sun Java Web Console.


Sun Java(TM) Web Console - Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
Search Print
Home Bookmarks Release Notes Plug-ins Extensions Support Mozilla Community
CONSOLE VERSION
LOG OUT HELP
Java™ Web Console
Java
Sun™ Microsystems, Inc.
Done

https://vincent:6789/console/launch/Launch

User: root Server: vincent

□ Start Each Application in a New Window

Systems

SunPlex Manager

Storage

No applications available

Services

No applications available

Desktop Applications

No applications available

Other

No applications available

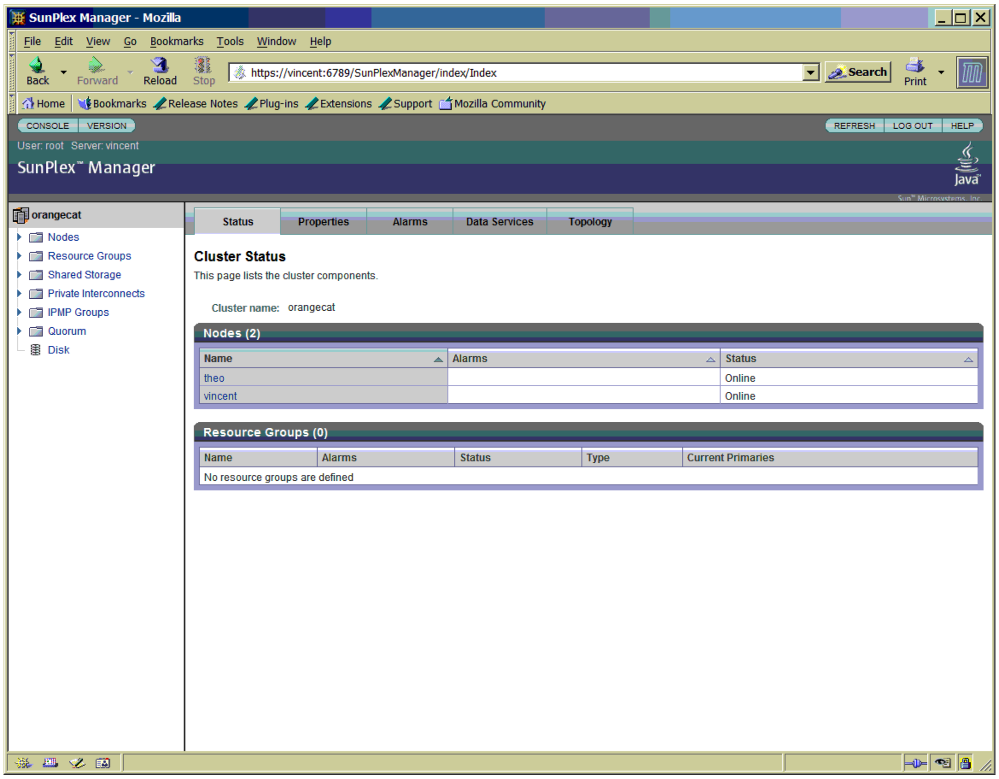
Figure 5-3 Applications Available Underneath Sun Java Web Console

Navigating the SunPlex Manager Main Window

SunPlex Manager navigation is simple, with the tree-like navigation bar on the left, and the display corresponding to your current selection in the main frame.

Topological views are available through a tab in the main window.

Figure 5-4 shows the initial screen.


SunPlex Manager - Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
https://vincent:6789/SunPlexManager/index/Index
Search Print
Home Bookmarks Release Notes Plug-ins Extensions Support Mozilla Community
CONSOLE VERSION
REFRESH LOG OUT HELP
User: root Server: vincent
SunPlex™ Manager
Java
Sun™ Microsystems, Inc.
orangecat
Nodes
Resource Groups
Shared Storage
Private Interconnects
IPMP Groups
Quorum
Disk
Status Properties Alarms Data Services Topology

Cluster Status

This page lists the cluster components.

Cluster name: orangecat

Nodes (2)

Name	Alarms	Status
theo		Online
vincent		Online

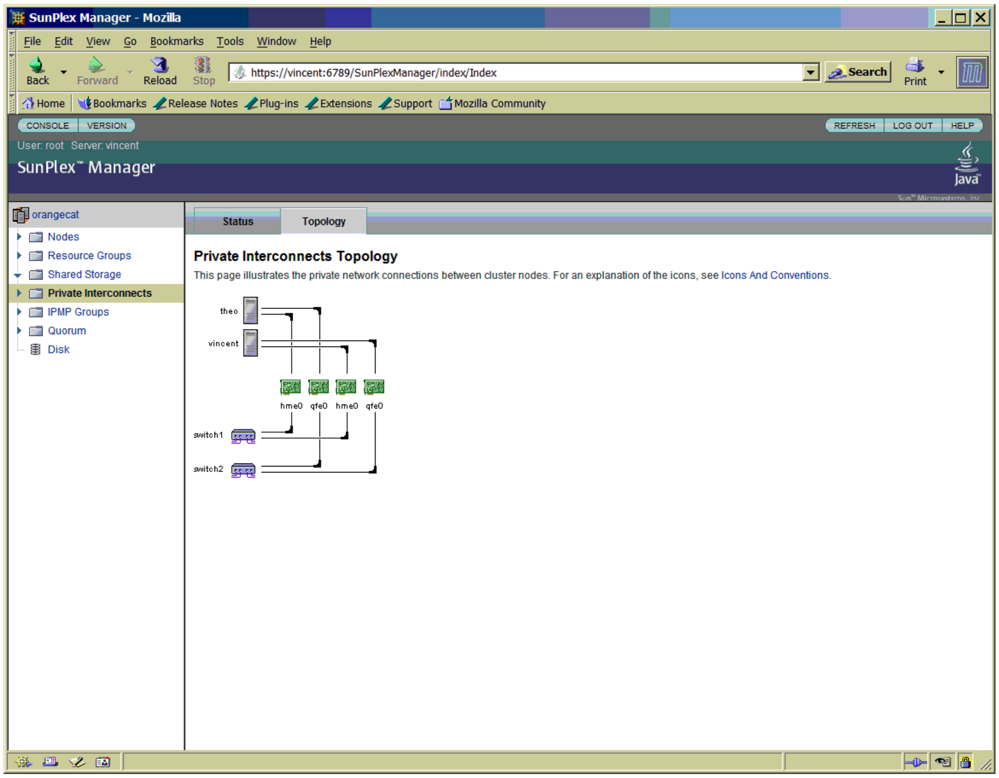
Resource Groups (0)

Name	Alarms	Status	Type	Current Primaries
No resource groups are defined				

Figure 5-4 SunPlex Manager Main Window

Viewing a SunPlex Manager Graphical Topology Example

Figure 5-5 demonstrates one of the graphical topologies available in SunPlex Manager. This example shows the nodes and transport topology.



Private Interconnects Topology

This page illustrates the private network connections between cluster nodes. For an explanation of the icons, see Icons And Conventions.

Figure 5-5 Graphic Topology Window

Exercise: Performing Basic Cluster Administration

In this exercise, you complete the following tasks:

- Task 1 – Verifying Basic Cluster Status
- Task 2 – Starting and Stopping Cluster Nodes
- Task 3 – Placing a Node in Maintenance State
- Task 4 – Preventing Cluster Amnesia
- Task 5 – Booting Nodes in Non-Cluster Mode
- Task 6 – Navigating SunPlex Manager

Preparation

Join all nodes in the cluster and run the `cconsole` tool on the administration workstation.

Note – During this exercise, when you see italicized names, such as ***IPaddress***, ***enclosure_name***, ***node1***, or ***clustername*** embedded in a command string, substitute the names appropriate for your cluster.

Task 1 – Verifying Basic Cluster Status

Perform the following steps to verify the basic status of your cluster:

1. Use the `scstat` command to verify the current cluster membership.

   ```
   # scstat -q
   ```

2. Record the quorum configuration from the previous step.

   Quorum votes possible: _____

   Quorum votes needed: _____

   Quorum votes present: _____

3. Verify all cluster disk paths.

   ```
   # scdpm -p all:all
   ```

4. Verify the revision of the currently installed Sun Cluster software on each cluster node.

   ```
   # scinstall -pv
   ```

Task 2 – Starting and Stopping Cluster Nodes

Perform the following steps to start and stop configured cluster nodes:

1. Verify that all nodes are active cluster members.
2. Shut down Node 2.

   ```
   # init 0
   ```
3. Join Node 2 into the cluster again by performing a boot operation.
4. When all nodes are members of the cluster, run `scshutdown` on one node to shut down the entire cluster.

   ```
   # scshutdown -y -g 60 Log off now!!
   ```
5. Join Node 1 into the cluster by performing a boot operation.
6. When Node 1 is in clustered operation again, verify the cluster quorum configuration again.

   ```
   # scstat -q | grep “Quorum votes”
     Quorum votes possible:      3
     Quorum votes needed:        2
     Quorum votes present:       2
   ```
7. Leave other nodes down for now.

Task 3 – Placing a Node in Maintenance State

Perform the following steps to place a node in the maintenance state:

1. On Node 1, use the `scconf` command to place Node 2 into a maintenance state.

   ```
   # scconf -c -q node=node2,maintstate
   ```

Note – Substitute the name of your node for ***node2***.

2. Verify the cluster quorum configuration again.

   ```
   # scstat -q | grep “Quorum votes”
   ```

Note – The number of *possible* quorum votes should be reduced by two. The quorum disk drive vote is also removed.

3. Boot Node 2 again. This should reset its maintenance state. You should see the following message on both nodes:

   ```
   NOTICE: CMM: Votecount changed from 0 to 1 for node
   theo
   ```

4. Boot any additional nodes into the cluster.
5. Verify the cluster quorum configuration again. The number of possible quorum votes should be back to normal.

Task 4 – Preventing Cluster Amnesia

Perform the following steps to demonstrate how the cluster prevents cluster amnesia, using persistent reservations on the quorum device. You can review the information about what you will see by checking Module 3, "Preparing for Installation and Understanding Quorum Devices."

1. Verify that all nodes are active cluster members. If you have a three-node cluster, shut down the node *not* attached to storage with `init 0`.
2. In another window, shut down Node 1 with `init 0`.
3. Shut down Node 2.

   ```
   # init 0
   ```

4. Boot Node 1.

   Node 1 should hang waiting for operational quorum. This is because the reservation key for Node 1 has been removed from the quorum disk.

5. Now boot Node 2. Both nodes should complete the cluster software startup sequence and join in clustered operation.
6. Boot any other nodes into the cluster.

Task 5 – Booting Nodes in Non-Cluster Mode

Perform the following steps to boot a cluster node so that it does not participate in clustered operation:

1. Shut down Node 2.
2. Boot Node 2 in non-cluster mode.

```
ok boot -x
```

Note – You should see a message similar to: `Not booting as part of a cluster`. You can also add the single-user mode option: `boot -sx`.

3. Verify the quorum status again.
4. Return Node 2 to clustered operation.

```
# init 6
```

Task 6 – Navigating SunPlex Manager

Perform the following steps:

1. In a web browser window on your administrative workstation, type the following URL:

   `https://`*`nodename`*`:6789`

   where *`nodename`* is the name of any of your nodes currently booted into the cluster.

   If you do not succeed in reaching the Sun Java Web Console, you might need to disable or set exceptions for the proxy settings in your web browser. Ask your instructor if you need help.

   If you are running the class with remote equipment, you may be able to use a web browser to access SunPlex Manager in one of two ways:

   a. Running the browser on your remote display.

   b. Running the browser locally and tunnelling the traffic through `ssh`.

Note – Consult your instructor about how this will be done.

2. Log in as the user `root` with the root password.
3. Navigate to the SunPlex Manager application.
4. Familiarize yourself with SunPlex Manager navigation and the topological views.

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 6

Using VERITAS Volume Manager for Volume Management

Objectives

Upon completion of this module, you should be able to:

- Describe the most important concepts of VERITAS Volume Manager (VxVM) and issues involved in using VxVM in the Sun Cluster 3.1 software environment
- Differentiate between `bootdg/rootdg` and shared storage disk groups
- Initialize a VERITAS Volume Manager disk
- Describe the basic objects in a disk group
- Describe the types of volumes that will be created for Sun Cluster 3.1 software environments
- Describe the general procedures for installing and administering VxVM in the cluster
- Describe the DMP restrictions
- Install VxVM 4.1 software using `installvm` and initialize VxVM using `scvxinstall`
- Use basic commands to put disks in disk groups and build volumes
- Describe the two flags used to mark disks under the normal hot relocation
- Register and resynchronize VxVM disk groups with the cluster
- Manage device groups
- Create global and failover file systems
- Describe the procedure used for mirroring the boot drive

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- Which VxVM features are the most important to clustered systems?
- Are there any VxVM feature restrictions when VxVM is used in the Sun Cluster software environment?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Microsystems, Inc. *VERITAS Volume Manager™ 4.1 Administrator's Guide* (Solaris)
- Sun Microsystems, Inc. VERITAS Volume Manager™ 4.1 Hardware Notes (Solaris)
- Sun Microsystems, Inc. *VERITAS Volume Manager™ 4.1 Troubleshooting Guide* (Solaris)

Introducing VxVM in the Sun Cluster Software Environment

This module does not intend to replace a full four-day class in VxVM administration. Rather, it briefly introduces the most important concepts of VxVM and focuses on issues involved in using VxVM in the Sun Cluster 3.1 environment.

Only the following versions of VxVM are supported in Sun Cluster 3.1 software:

- VxVM 3.5, for Solaris 8 OS and Solaris 9 OS
- VxVM 4.0, for Solaris 8 OS and Solaris 9 OS
- VxVM 4.1, for Solaris 8 OS, Solaris 9 OS, and Solaris 10 OS

VxVM is available for SPARC servers only. There is no version compatible with the Solaris OS on x86 platforms. Clusters running on the Solaris x86 platform use Solaris Volume Manager.

Exploring VxVM Disk Groups

The assignment of disk drives or logical unit numbers (LUNs) into *disk groups* is the most important organizational concept to understand to manage VxVM in the cluster environment.

Every disk or LUN used by VxVM must be a member of exactly one disk group. Disks in the same disk group act as an organizational unit.

Shared Storage Disk Groups

All of the data associated with applications that run in the Sun Cluster 3.1 software environment must be in storage that is physically ported to at least two nodes. Disk groups are created using *only* disks in multiple arrays that are connected to the same collection of nodes.

VERITAS Management on Local Disks (Optional in VxVM 4.x)

Prior to VxVM 4.0 VxVM *required* that each node have an independent disk group named `rootdg`. In these versions you can not configure anything else in VxVM until `rootdg` is configured. You needed to have this disk group even if you did not intend VxVM to manage your boot disk at all, in that case you would have to have at least one (preferably two, for high-availability) “sacrificial” local disks so that you could have a `rootdg`.

Starting in VxVM 4.0, it is *optional* to have any disk group at all configured for the local disks. You are perfectly free to not configure any VxVM configurations on local disks, and you still are able to go ahead and create your shared storage disk groups.

If you want to have VxVM 4.x manage your boot disk (that is, to *encapsulate* your boot disk), then you will have this disk (and presumably another -- there is no reason to encapsulate root if you are not going to mirror it) in a dedicated local VxVM group. This group can have any group name. VxVM will automatically set up a symbolic link so that whatever the actual group name is there will be a link to it called `bootdg`.

Typical Sun Cluster VxVM Configuration

Figure 6-1 shows a typical organization of disks into disk groups in a simple Sun Cluster software environment. While the name of the actual disk group underlying the bootdg link can be anything, the tool that is used in the Sun Cluster environment to encapsulate the root disk (scvxinstall, which will be discussed later in the module still names it rootdg.)

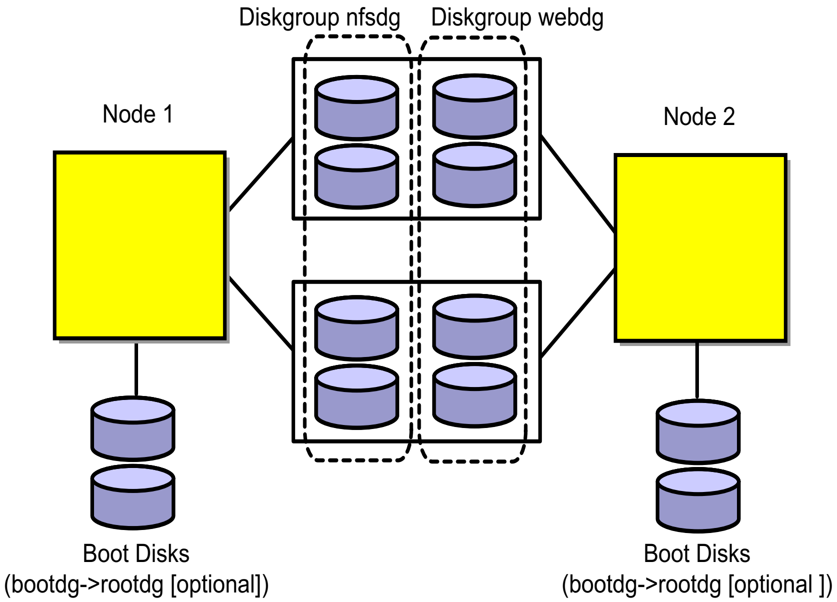


Figure 6-1 Typical Organization of Disks in a Sun Cluster Software Environment

Shared Storage Disk Group Ownership

While the disks in disk groups used for Sun Cluster software application data must be physically connected to at least two nodes, disk groups are owned, or *imported*, by only one node at a time. The node that is currently importing a disk group will be the one that does the following:

- It physically reads and writes data to the drives within the disk group.
- It manages the disk group. VxVM commands pertaining to that diskgroup can be issued only from the node importing the disk group.
- It can voluntarily give up ownership of the disk group by *deporting* the disk group.

Sun Cluster Management of Disk Groups

After a disk group is managed by the cluster, that cluster is responsible for issuing all VxVM commands to import and deport disk groups. The node currently importing the disk group becomes the *primary device group server.*

Registering VxVM disk groups so that the Sun Cluster software has knowledge of them is described in"Registering VxVM Disk Groups" on page 6-36.

Sun Cluster Global Devices Within a Disk Group

While the fact that only one node imports a disk group remains true in the Sun Cluster software environment, Sun Cluster software's global device infrastructure will make it *appear* that devices in the disk group are accessible from all nodes in the cluster, including nodes that are not even physically connected to the disks in the disk group.

All nodes that are not the current primary for a particular disk group, even other nodes that *are* physically connected to the disks, actually access the device data through the cluster transport.

VxVM Cluster Feature Used Only for ORACLE® Parallel Server and ORACLE Real Application Cluster

You might see references throughout the VERITAS documentation to a Cluster Volume Manager (CVM) feature, and creating disk groups with a specific *shared* flag.

This is a separately licensed feature of VxVM that allows simultaneous ownership of a disk group by multiple nodes.

In the Sun Cluster 3.1 software, this VxVM feature can be used *only* by the ORACLE Parallel Server (OPS) and ORACLE Real Application Clusters (RAC) application.

Do not confuse the usage of *shared storage disk group* throughout this module with the CVM feature of VxVM.

Initializing a VERITAS Volume Manager Disk

Before a disk can be put into a VxVM disk group, it must be initialized by VxVM. The disk is divided into two sections called the *private* region and the *public* region.

The private and public regions are used for the following different purposes:

- The private region is used for configuration information.
- The public region is used for data storage.

Traditional Solaris Disks and Cross-Platform Data Sharing (CDS) Disks

VxVM has two different ways of initializing disks. One is the traditional way which uses separate Solaris disk slices for the private region (slice 3) and the public region (slice 4). This is a Solaris specific disk layout.

As shown in Figure 6-2, the private region is small. It is almost always one cylinder in size on any modern disk.

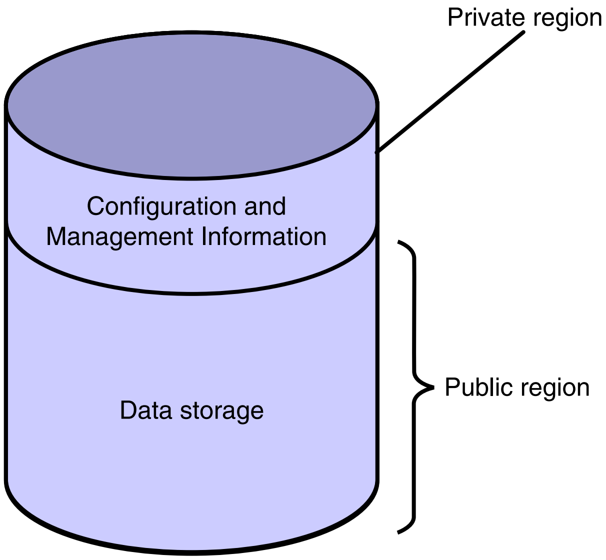


Figure 6-2 VxVM Disk Initialization (Traditional Solaris Only Disks)

Starting in VxVM 4.0, the new default way to partition disks is called Cross-Platform Data Sharing (CDS) disks. The VXVM configuration on these disks can be read by VxVM running on all its supported platforms, not just Solaris. As shown in Figure 6-3 this layout combines the configuration and data storage into one large partition (slice 7) covering the entire disk.

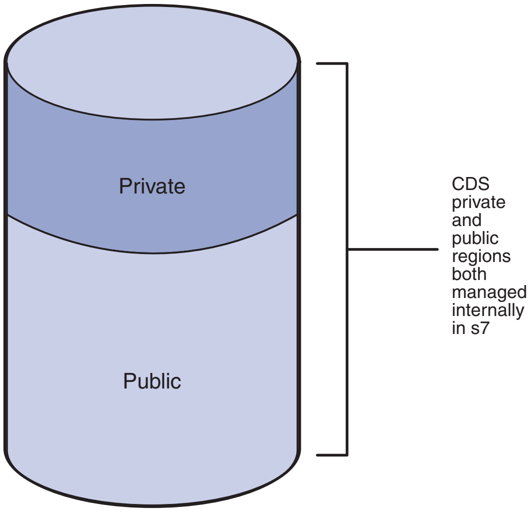

Private
Public
CDS private and public regions both managed internally in s7

Figure 6-3 CDS Disk Layout

Since, in the Sun Cluster environment you will not be able to access VxVM disks from any servers other than those in the same cluster, it does not matter which disk layout you choose for your data disks.

You cannot use CDS disks for the `bootdg`.

While initializing a disk and putting it in a disk group are two separate operations, the `vxdiskadd` utility, demonstrated later in this module, can perform both steps for you. The defaults are always to use CDS disks, but the `vxdiskadd` command can guide you through setting up your disks either way.

Reviewing the Basic Objects in a Disk Group

This section reviews some of the basic naming terminology of objects in disk groups. Many of the objects (such as subdisks and plexes) are often automatically created for you when you use the recommended commands (such as `vxassist`) to create volumes.

Although the GUI for VxVM furnishes useful visual status information, the most reliable and the quickest method of checking status is from the command line. Command-line status tools are easy to use in script files, `cron` jobs, and remote logins.

Disk Names or Media Names

Each disk that you put in a diskgroup has a logical name that you can assign. This logical name can be anything, and is independent of its Solaris OS logical device name (`c#t#d#`).

VxVM does *not* use the Sun Cluster Disk IDs (DIDs). Its own logical naming scheme serves a similar purpose.

Subdisk

A subdisk is a contiguous piece of a physical disk that is used as a building block for a volume. The smallest possible subdisk is 1 block and the largest is the whole public region of the disk.

Plex

A plex, or data plex, has the following characteristics:

- A plex is a copy of the data in the volume.
- A volume with *one* data plex is *not* mirrored.
- A volume with *two* data plexes *is* mirrored.
- A volume with *five* data plexes is mirrored *five* ways.

Volume

The volume is the actual logical storage device created and used by all disk consumers (for example, file system, swap space, and ORACLE raw space) above the volume management layer.

Only volumes are given block and character device files.

The basic volume creation method (`vxassist`) lets you specify parameters for your volume. This method automatically creates the subdisks and plexes for you.

Layered Volume

A layered volume is a technique used by VxVM that lets you create RAID 1+0 configurations. For example, you can create two mirrored volumes, and then use those as components in a "larger" striped volume.

While this sounds complicated, the automation involved in the recommended `vxassist` command makes such configurations easy to create.

Exploring Volume Requirements in the Sun Cluster Software Environment

The only requirement for creating a volume in shared storage disk groups for the Sun Cluster software environment, is that you *must* mirror across controllers unless there is full redundancy (hardware RAID and multipathing) provided by the controller itself.

With that in mind, any of the following are acceptable volumes to hold your Sun Cluster software data.

Simple Mirrors

Figure 6-4 demonstrates subdisks from disks in different storage arrays forming the plexes of a mirrored volume.

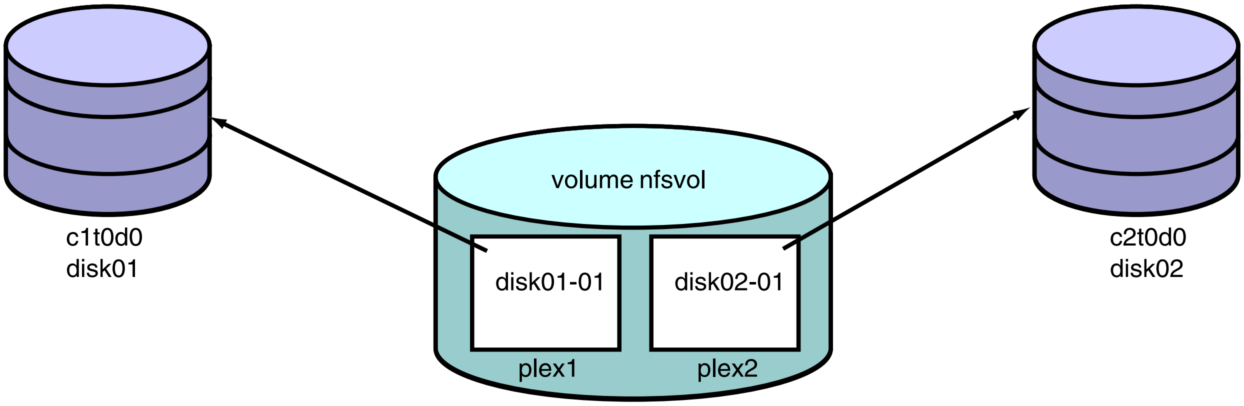


Figure 6-4 Forming the Plexes of a Mirrored Volume

Mirrored Stripe ("Mirror-Stripe")

Figure 6-5 demonstrates subdisks from disks in the same array striped together. This is mirrored with a similar configuration to the other array.

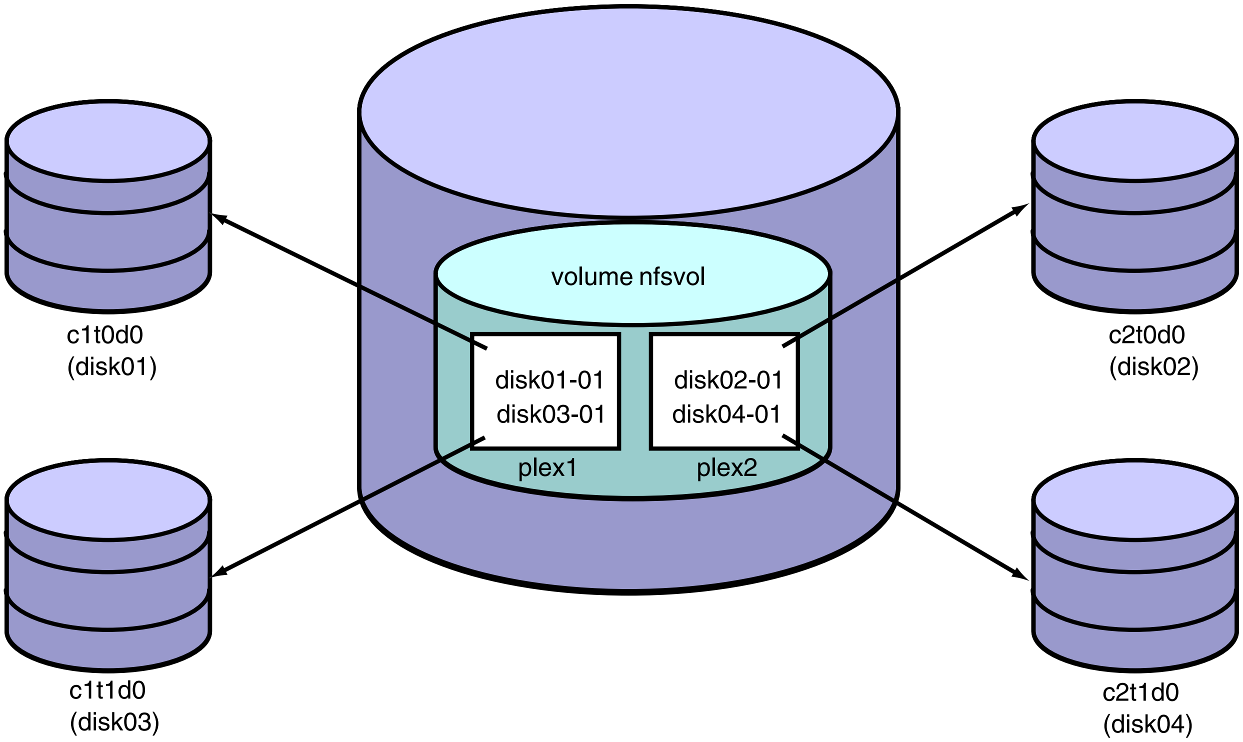


Figure 6-5 Mirror-Stripe Volume

Striped Mirrors ("Stripe-Mirror")

Figure 6-6 shows two mirrored sub-volumes, both mirrored across storage arrays, striped together to form the final volume.

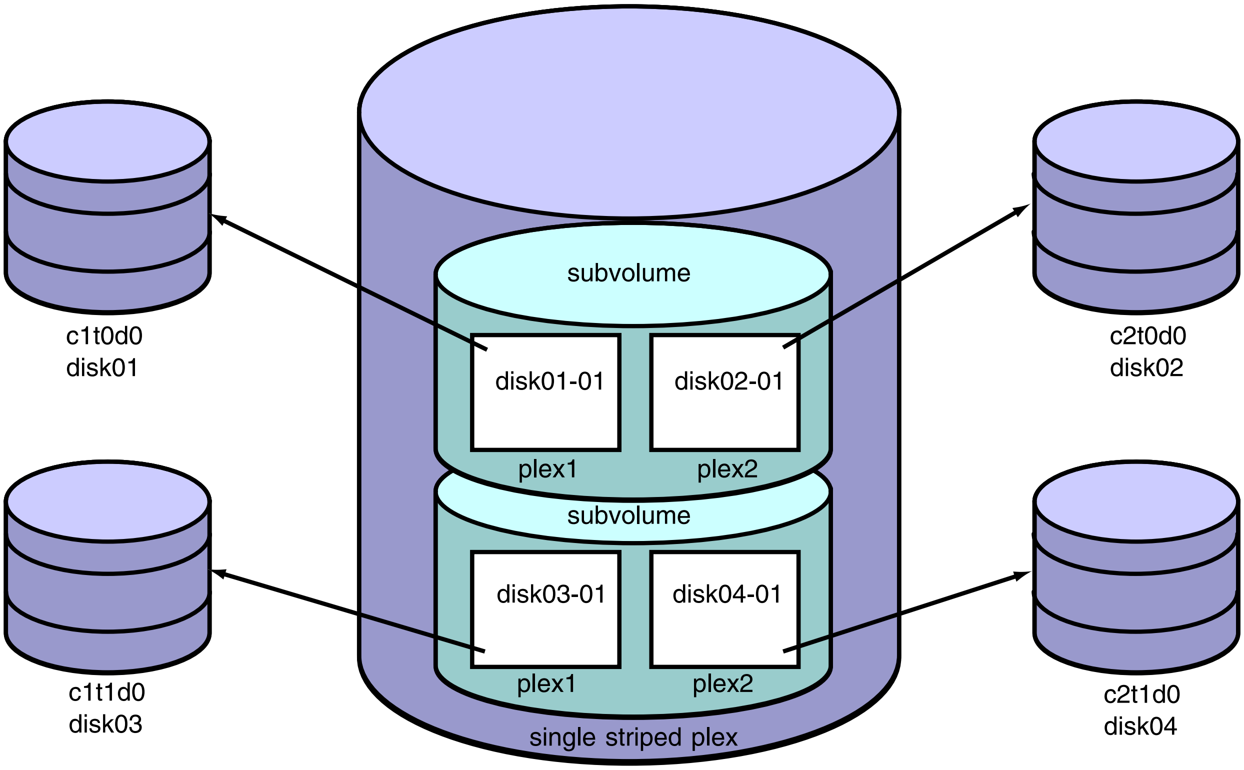


Figure 6-6 Stripe-Mirror Volume

Fortunately, as demonstrated later in the module, this is easy to create.

Often this configuration (RAID 1+0) is preferred because there may be less data to resynchronize after disk failure, and you could suffer simultaneous failures in each storage array. For example, you could lose both `disk01` and `disk04`, and your volume will still be available.

Dirty Region Logs for Volumes in the Cluster

Dirty region log (DRL) is an optional feature you can add on a per-volume basis. The DRL is an extra plex for the volume that does not hold data but rather holds bitmaps of which regions of the data may be “dirty.” That is, they may have to be resynchronized in the volume after a Solaris OS crash.

DRLs have no effect whatsoever on resynchronization behavior following disk failure.

Without a DRL, a mirrored volume must be completely resynchronized after a Solaris OS crash. That is because on recovery (on another node, in the cluster environment), VxVM has no way of knowing which bits of data it may have been “in the middle of writing” when the crash occurred.

With the DRL, VxVM knows exactly which regions of the data marked in the DRL bitmap might need to be resynchronized. Regions not marked in the bitmap are known to be clean, and do not need to be resynchronized.

You use a DRL for any large volume in the cluster. The whole purpose of the cluster is making everything behave as well as possible after crashes.

Size of DRL

DRLs are small. Use the `vxassist` command to size them for you. The size of the logs that the command chooses depends on the version of VxVM, but might be, for example, 26 kilobytes (Kbytes) of DRL per 1 gigabyte (Gbyte) of storage.

If you grow a volume, just delete the DRL, and then use the `vxassist` command to add it back again so that it gets sized correctly.

Location of DRL

You can put DRL subdisks on the same disk as the data if the volume will not be heavily written. An example of this is web data.

For a heavily written volume, performance will be better with a DRL on a different disk than the data. For example, you could dedicate one disk to holding all the DRLs for all the volumes in the group. They are so small that performance on this one disk is still fine.

Viewing the Installation and bootdg/rootdg Requirements in the Sun Cluster Software Environment

The rest of this module is dedicated to general procedures for installing and administering VxVM in the cluster. It describes both easy, automatic ways, and more difficult, manual ways for meeting the requirements described in the following list.

- The vxio major number must be identical on all nodes – This is the entry in the /etc/name_to_major file for the vxio device driver. If the major number is not the same on all nodes, the global device infrastructure cannot work on top of VxVM.
- VxVM *must be* installed on all nodes physically connected to shared storage – On non-storage nodes, you can install VxVM if you will use it to encapsulate and mirror the boot disk.

  If not (for example, you may be using Solaris Volume Manager software to mirror root storage), then a non-storage node requires only that the vxio major number be added to the /etc/name_to_major file.
- A license is required on all storage nodes not attached to Sun StorEdge A5x00 arrays – But, if you choose not to install VxVM on a non-storage node you don’t need a license for that node.

Requirements for bootdg/rootdg

- A standard rootdg must be created on all nodes where VxVM 3.5 is installed – The options to initialize rootdg on each node are:
  - Encapsulate the boot disk so it can be mirrored – Encapsulating the boot disk preserves its data, creating volumes in rootdg for your root disk partitions, including /global/.devices/node@#.

    You cannot encapsulate the boot disk if it has more than five slices on it already. VxVM requires two free slice numbers to create the private and public regions of the boot disk.
  - Initialize other local disks into rootdg – These could be used to hold other local data, such as application binaries.
- You can have an optional bootdg (pointing to actual arbitrary group name) on VxVM 4.x. You will likely have this only if you want to encapsulate and mirror the OS disk.

Issues When Encapsulating the Boot Disk

There are several issues involved with encapsulating the boot disk. The first issue has nothing specifically to do with the cluster environment, but it may make you think harder about whether you really want to encapsulate root using VxVM, or if you want to use another solution to manage your boot disk.

At the time of writing of this course, you cannot have a logging root file system on Solaris 9 OS or Solaris 10 OS if you encapsualte the boot disk with VxVM 4.1

Note – The problem involves boot timing. The problem has been solved for VxVM 3.5 (with patch 112392-08 or later) and VxVM 4.0 (with patch 115217-05 or later), but only for Solaris 9 OS. Solaris 10 OS requires VxVM 4.1, where the issue has not been solved at this time. You can search for Sun Alert ID #101562 (if you have an account on SunSolve) for more information. The VxVM 4.1 boot disk encapsulation process automatically puts the `nologging` option in the `/etc/vfstab` file for your root file system for Solaris 9 and Solaris 10. Solaris 8 is not effected by this problem

It is critical to note that without a logging root file system you can suffer much longer boot times after a Solaris crash, as your root file system must undergo the `fsck` procedure.

Cluster-Specific Issues When Encapsulating the Boot Disk

The cluster-specific caveats all involve the fact that among the file systems being encapsulated on your boot disk is the `/global/.devices/node@#` file system. The requirements are the following:

- Unique volume name across all nodes for the `/global/.devices/node@#` volume
- Unique minor number across nodes for the `/global/.devices/node@#` volume

The reason for these restrictions is that this (unique among the file systems on your boot disk) is mounted as a global file system. The normal Solaris OS `/etc/mnttab` logic predates global file systems and still demands that each device and major/minor combination be unique.

If you want to encapsulate the boot disk, you should use the `scvxinstall` utility which will automate the correct creation of this volume (with different volume names and minor numbers on each node). This is discussed later in this module.

Describing DMP Restrictions

The DMP driver is a VxVM product feature. As shown in Figure 6-7, the DMP driver can access the same storage array through more than one path. The DMP driver automatically configures multiple paths to the storage array if they exist. Depending on the storage array model, the paths are used either simultaneously for load-balancing or one at a time in a primary/backup configuration.

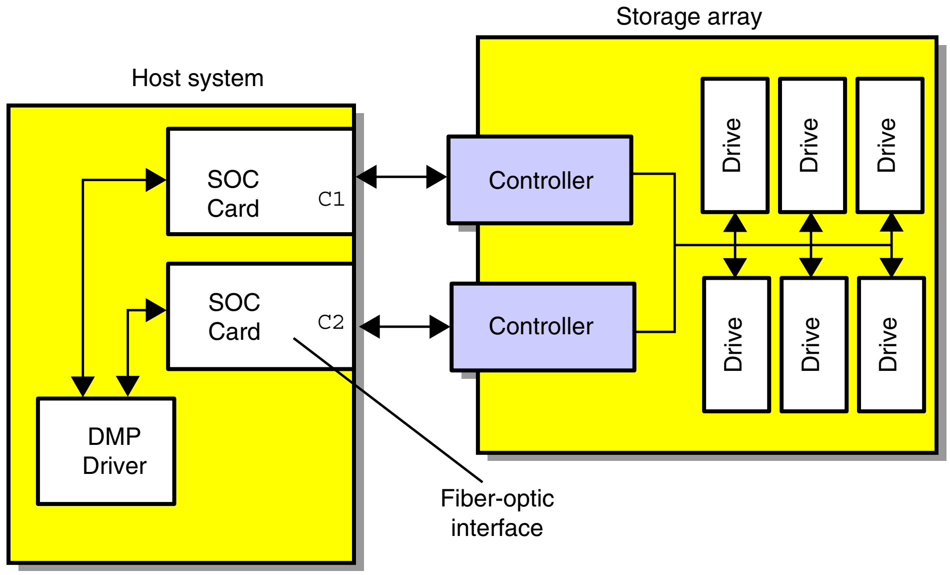


Figure 6-7 Dynamic Multipath Driver

The Sun Cluster software environment is incompatible with multipathed devices under control of the DMP device driver. In VxVM versions 3.1 and earlier (which are not supported by Sun Cluster 3.1 software), DMP could be permanently disabled by laying down some dummy symbolic links before installing the VxVM software. You will notice in examples later in this module that the `scvxinstall` utility actually still tries to do this, although it does not have any effect on VxVM versions actually supported in the Sun Cluster 3.1 software.

DMP Restrictions in Sun Cluster 3.1 (VxVM 4.x and 3.5)

In versions of VxVM supported by Sun Cluster 3.1 (VxVM 4.1, 4.0. and 3.5), you cannot permanently disable the DMP feature. Even if you take steps to disable DMP, it automatically re-enables itself each time the system is booted.

Having multiple paths from the same node to the same storage under control of VxVM DMP is still *not* supported. You can do one of the following:

- Do not have any multiple paths at all from a node to the same storage
- Have multiple paths from a node under the control of Sun StorEdge Traffic Manager software or EMC PowerPath software

Supported Multipathing Under Sun StorEdge Traffic Manager Software

Multipathing *is* supported using Sun StorEdge Traffic Manager software, formerly known as `mpxio`. Volume Manager detects Sun StorEdge Traffic Manager software devices and does not use DMP on those paths.

Figure 6-8 shows an example of multipathing using Sun StorEdge T3 array partner pairs.

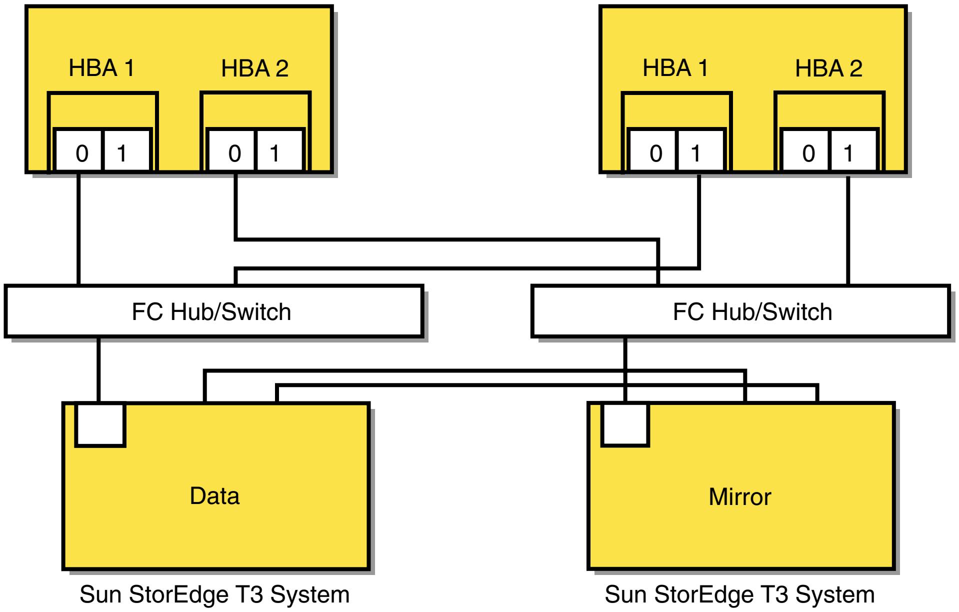


Figure 6-8 Example of Multipathing Using Sun StorEdge T3 Array Partner-Pair

Installing VxVM in the Sun Cluster 3.1 Software Environment

The following sections describe the following:

- Using the `installvm` utility to install and initialize VxVM 4.1
- Using the `scvxinstall` utility to install VxVM (versions below VxVM 4.1), and to synchronize `vxio` major numbers and encapsulate root (any version)
- Installing VxVM manually after the cluster is installed
- Addressing VxVM cluster-specific issues if you installed VxVM before the cluster was installed

Using the VxVM `installvm` Utility

Starting in VxVM 4.1 it is *required* that you install the VxVM software first before configuring for use with the cluster with `scvxinstall`. The recommended way is to use VXVM's own `installvm` utility. This is a text-based installer that has the following characteristics:

- It can install the software on multiple servers (multiple nodes) if `rsh/rcp` or `ssh/scp` is enabled between the nodes (or you can choose to run it separately on each node).
- The minimum set of software that it installs includes the VxVM GUI back-end (but not the front-end).
- It guides you through entering licenses and initializing the software at the end of the install.

Note – The VxVM4.1 packages are distributed on the media as `.tar.gz` files. The `installvm` utility, extracts these correctly. If you wanted to install VxVM 4.1 without using `installvm`, you could copy and extract these archives manually and then install, at a minimum, the `VRTSvlic` and `VRTSvxvm` packages, in that order, on each node. You would then have to manually use `vxinstall` to license and initialize the software.

Example of installvm Screens

The installvm guides you logically through the tasks. The initial screen looks like the following:

```
VERITAS VOLUME MANAGER 4.1 INSTALLATION PROGRAM

Copyright (c) 2005 VERITAS Software Corporation. All rights reserved.
.
.

Enter the system names separated by spaces on which to install VxVM:
vincent theo


Checking system communication:

Checking OS version on vincent ............................ SunOS 5.10
Verifying global zone on vincent .............................. global
Checking VRTSvxvm package ............................... not installed
Verifying communication with theo .................... ping successful
Attempting rsh with theo .............................. rsh successful
Attempting rcp with theo .............................. rcp successful
Checking OS version on theo ............................... SunOS 5.10
Verifying global zone on theo ................................. global
Checking VRTSvxvm package ............................... not installed
Creating log directory on theo .................................. Done

Logs for installvm are being created in /var/tmp/installvm628164918.

Using /usr/bin/rsh and /usr/bin/rcp to communicate with remote systems.

Initial system check completed successfully.

Press [Return] to continue:
```

Describing the scvxinstall Utility

The scvxinstall utility is a shell script included with the cluster that automates VxVM installation (in versions below 4.1) and configuration.

You should use scvxinstall to actually install the VxVM packages for versions below VxVM 4.1; starting in VxVM 4.1 you cannot use scvxinstall until after the VxVM software is already installed, licensed, and initialized.

Regardless of whether you can use scvxinstall to install the software, it has the following functionality:

- If you choose *not* to encapsulate the boot disk:
    - It tries to disable DMP. As previously mentioned, this is ineffective on the VxVM versions supported in Sun Cluster 3.1 software.
    - It automatically negotiates a vxio major number and properly edits the /etc/name_to_major file.
- If you choose to encapsulate the boot disk, it has the above functionality and the following additions:
    - It encapsulates your boot disk in a disk group named rootdg, and creates the bootdg link in VxVM 4.x.
    - It gives different volume names for the volumes containing the /global/.devices/node@# file systems on each node.
    - It edits the vfstab file properly for this same volume. The problem is that this particular line currently has a DID device on it, and the normal VxVM does not understand DID devices.
    - It installs a script to “reminor” the rootdg on reboot.
    - It reboots you into a state where VxVM on your node is fully operational.

Examining an `scvxinstall` Example

The following is sample output from running the `scvxinstall` utility. Since the example is using VxVM 4.1, this would fail if the VxVM installation and initialization had not already been performed by `installvm`.

```
# scvxinstall

Do you want Volume Manager to encapsulate root [no]?  yes

Disabling DMP.
scvxinstall:  /dev/vx/dmp is not set up correctly.
scvxinstall:  Warning: Unable to disable DMP, but installation will
continue...
scvxinstall:  /dev/vx/rdmp is not set up correctly.
scvxinstall:  Warning: Unable to disable DMP, but installation will
continue...
The Volume Manager package installation is already complete.
Discovered and using the predefined vxio number 270...
Volume Manager installation is complete.
Verifying encapsulation requirements.

The Volume Manager root disk encapsulation step will begin in 20 seconds.
Type Ctrl-C to abort ....................
Arranging for Volume Manager encapsulation of the root disk.
Reinitialized the volboot file...
The setup to encapsulate rootdisk is complete...
Updating /global/.devices entry in /etc/vfstab.

This node will be re-booted in 20 seconds.
Type Ctrl-C to abort ....................
```

The node reboots two times. The first reboot completes the VERITAS boot disk encapsulation process, and the second reboot brings the node back into clustered operations again.

Installing and Configuring VxVM Manually

You should use the `scvxinstall` utility to install (versions less than VxVM 4.1) and initialize VxVM (all versions) in an already-clustered environment, especially if you are encapsulating your boot disk.

Manually Installing VxVM Software (Versions less than 4.1)

The following are the only VxVM packages required on all the nodes:

- VRTSvxvm
- VRTSvlic

You are also free to install the packages for the VERITAS Enterprise Administrator GUI.

Manually Checking and Fixing vxio Major Numbers

Immediately after software installation, you must check all the nodes and make sure the major number for vxio is identical.

```
# grep vxio /etc/name_to_major
vxio 270
```

If it is not the same on all nodes, pick a number to which to change it that does not appear anywhere else in any node's file already. You might have to edit the file on one or more nodes.

Manually Using vxdiskadm to Encapsulate Boot

Use the following steps to make boot-disk encapsulation work correctly on the cluster:

1. Give a different name for the root disk on each node. Do *not* use the default disk name on each node.

   The result will be that the volume name for the /global/.devices/node@# file system will be different on each node.

2. Make sure you do not enter any shared disk storage drives in the rootdg (or whatever you call the group in VxVM4.x).

3. Before you reboot manually, put back the normal /dev/dsk/c#t#d#s# and /dev/rdsk/c#t#d#s# on the line in the /etc/vfstab file for the /global/.devices/node@# file system.

   Without making this change, VxVM cannot figure out that it must edit that line after the reboot to put in the VxVM volume name.

4. Reboot with boot -x. VxVM arranges a second reboot after that which will bring you back into the cluster.

5. That second reboot will fail on all but one of the nodes because of the conflict in minor numbers for `rootdg` (which is really just for the `/global/.devices/node@#` volume).

   On those nodes, give the root password to go into single user mode. Use the following command to fix the problem:

   ```
   # vxdg reminor rootdg 50*nodeid
   ```

   For example, use 50 for Node 1, 100 for Node 2, and so forth. This will provide a unique set of minor numbers for each node.

6. Reboot the nodes for which you had to repair the minor numbers.

Configuring a Pre-existing VxVM for Sun Cluster 3.1 Software

The last possibility is that you have a potential Sun Cluster 3.1 software node where the proper version of VxVM was already installed and initialized *before* the Sun Cluster environment was installed. The following subsections detail how to provision your VxVM to deal with the cluster-specific issues mentioned on the previous pages. They assume you will deal with these issues *before* you install the Sun Cluster software.

Fixing the `vxio` Major Number

If you need to change the `vxio` major number to make it agree with the other nodes, do the following:

1. Manually edit the `/etc/name_to_major` file.
2. Unmirror and unencapsulate the root disk, if it is encapsulated.
3. Reboot.
4. Reencapsulate and remirror the boot disk, if desired.

Fixing the `/globaldevices` Volume Name and Minor Number

If you are still at a point before cluster installation, you need to have a `/globaldevices` placeholder file system or at least a correctly-sized device ready for cluster installation.

If this is on a volume manager device:

1. Rename the volume if it has the same name as the `/globaldevices` or `/global/.devices/node@#` volume of any other node. You may need to manually edit the `/etc/vfstab` file.
2. Perform a reminor operation on the `rootdg` if it is using the same set of minor numbers as another node. Follow the same procedure as Step 5 on page 6-28.

Creating Shared Disk Groups and Volumes

The examples in this section are not trying to exhaustively cover all the possible ways of initializing disks and creating volumes. Rather, these are demonstrations of some simple ways of initializing disks, populating disk groups, and creating the mirrored configurations described earlier in the module.

Listing Available Disks

The following command lists all of the disks visible to VxVM:

```
vincent:/# vxdisk -o alldgs list
DEVICE       TYPE            DISK         GROUP        STATUS
c0t0d0s2     auto:sliced     rootdg_1     rootdg       online
c0t1d0s2     auto:none       -            -            online invalid
c1t0d0s2     auto:none       -            -            online invalid
c1t1d0s2     auto:cdsdisk    -           (testdg)      online
c1t2d0s2     auto:none       -            -            online invalid
c1t3d0s2     auto:none       -            -            online invalid
c1t8d0s2     auto:none       -            -            online invalid
c1t11d0s2    auto:none       -            -            online invalid
c2t0d0s2     auto:none       -            -            online invalid
c2t1d0s2     auto:cdsdisk    -           (testdg)      online
c2t2d0s2     auto:none       -            -            online invalid
c2t3d0s2     auto:none       -            -            online invalid
c2t8d0s2     auto:none       -            -            online invalid
c2t9d0s2     auto:none       -            -            online invalid
c2t10d0s2    auto:none       -            -            online invalid
c2t11d0s2    auto:none       -            -            online invalid
```

Without the `-o alldgs` list the command would have information only about disk groups currently imported on this particular node. In this output the `testdg` has parentheses around it since it is not imported on this node, but the command scans the configuration information on every single disk anyway.

Disks with the *invalid* state are not yet initialized by VxVM.

Initializing Disks and Putting Them in a New Disk Group

One of the simplest ways to do this is using the dialogue presented by the vxdiskadd command. It will guide you through initializing disks and putting them in existing or new groups.

```
# vxdiskadd c1t0d0 c1t1d0 c2t0d0 c2t1d0
.
.
```

Verifying Disk Groups Imported on a Node

The following command shows which groups are imported:

```
# vxdg list
NAME         STATE           ID
rootdg       enabled              1120070264.6.vincent
nfsdg        enabled,cds          1120071527.33.vincent
```

Note – The ID of a disk group contains the name of the node *on which it was created.* Later, you might see the exact same disk group with the same ID imported on a different node.

```
              # vxdisk list
c0t0d0s2     auto:sliced     rootdg_1     rootdg       online
c0t1d0s2     auto:none       -            -            online invalid
c1t0d0s2     auto:cdsdisk    nfs1         nfsdg        online nohotuse
c1t1d0s2     auto:cdsdisk    nfs3         nfsdg        online nohotuse
c1t2d0s2     auto:none       -            -            online invalid
c1t3d0s2     auto:none       -            -            online invalid
c1t8d0s2     auto:none       -            -            online invalid
c1t11d0s2    auto:none       -            -            online invalid
c2t0d0s2     auto:cdsdisk    nfs2         nfsdg        online nohotuse
c2t1d0s2     auto:cdsdisk    nfs4         nfsdg        online nohotuse
c2t2d0s2     auto:none       -            -            online invalid
c2t3d0s2     auto:none       -            -            online invalid
c2t8d0s2     auto:none       -            -            online invalid
c2t9d0s2     auto:none       -            -            online invalid
c2t10d0s2    auto:none       -            -            online invalid
c2t11d0s2    auto:none       -            -            online invalid
```

The following shows the status of an entire group where no volumes have been created yet:

```
# vxprint -g nfsdg
TY NAME         ASSOC    KSTATE   LENGTH   PLOFFS   STATE    TUTIL0
PUTIL0
dg nfsdg        nfsdg    -        -        -        -        -        -

dm nfs1         c1t0d0s2 -        71124864 -        -        -        -
dm nfs2         c2t0d0s2 -        71124864 -        -        -        -
dm nfs3         c1t1d0s2 -        71124864 -        -        -        -
dm nfs4         c2t1d0s2 -        71124864 -        -        -        -
```

Building a Simple Mirrored Volume

The following example shows building a simple mirror, with a subdisk from a disk in one controller mirrored with one from another controller.

```
# vxassist -g nfsdg make nfsvol 100m layout=mirror nfs1 nfs2
# vxprint -g nfsdg
TY NAME         ASSOC      KSTATE   LENGTH   PLOFFS STATE TUTIL0 PUTIL0
dg nfsdg        nfsdg      -        -        -        -        -      -

dm nfs1         c1t0d0s2   -        71124864 -        -        -      -
dm nfs2         c2t0d0s2   -        71124864 -        -        -      -
dm nfs3         c1t1d0s2   -        71124864 -        -        -      -
dm nfs4         c2t1d0s2   -        71124864 -        -        -      -

v  nfsvol       fsgen      ENABLED  204800   -        ACTIVE   -      -
pl nfsvol-01    nfsvol     ENABLED  204800   -        ACTIVE   -      -
sd nfs1-01      nfsvol-01  ENABLED  204800   0        -        -      -
pl nfsvol-02    nfsvol     ENABLED  204800   -        ACTIVE   -      -
sd nfs2-01      nfsvol-02  ENABLED  204800   0        -        -      -
```

Building a Mirrored Striped Volume (RAID 0+1)

The following example rebuilds the mirror. Each plex is striped between two disks in the same array so mirroring is still across controllers.

```
# vxassist -g nfsdg remove volume nfsvol
# vxassist -g nfsdg make nfsvol 100m layout=mirror-stripe mirror=ctlr
# vxprint -g nfsdg
TY NAME         ASSOC     KSTATE   LENGTH   PLOFFS STATE TUTIL0 PUTIL0
dg nfsdg        nfsdg     -        -        -        -        -      -

dm nfs1         c1t0d0s2  -        71124864 -        -        -      -
dm nfs2         c2t0d0s2  -        71124864 -        -        -      -
dm nfs3         c1t1d0s2  -        71124864 -        -        -      -
dm nfs4         c2t1d0s2  -        71124864 -        -        -      -

v  nfsvol       fsgen     ENABLED  204800   -        ACTIVE   -      -
pl nfsvol-01    nfsvol    ENABLED  204800   -        ACTIVE   -      -
sd nfs1-01      nfsvol-01 ENABLED  102400   0        -        -      -
sd nfs3-01      nfsvol-01 ENABLED  102400   0        -        -      -
pl nfsvol-02    nfsvol    ENABLED  204800   -        ACTIVE   -      -
sd nfs2-01      nfsvol-02 ENABLED  102400   0        -        -      -
sd nfs4-01      nfsvol-02 ENABLED  102400   0        -        -      -
```

Building a Striped Mirrored Volume (RAID 1+0)

The final example rebuilds the volume as a stripe of two mirrors. As mentioned earlier in this module, this layout is often preferred for faster resynchronization on disk failure and greater resiliency.

```
# vxassist -g nfsdg remove volume nfsvol
# vxassist -g nfsdg make nfsvol 100m layout=stripe-mirror mirror=ctlr
# vxprint -g nfsdg
TY NAME         ASSOC       KSTATE   LENGTH   PLOFFS STATE TUTIL0 PUTIL0
dg nfsdg        nfsdg       -        -        -        -        -       -

dm nfs1         c1t0d0s2    -        71124864 -        -        -       -
dm nfs2         c2t0d0s2    -        71124864 -        -        -       -
dm nfs3         c1t1d0s2    -        71124864 -        -        -       -
dm nfs4         c2t1d0s2    -        71124864 -        -        -       -

v  nfsvol       fsgen       ENABLED  204800   -        ACTIVE   -       -
pl nfsvol-03    nfsvol      ENABLED  204800   -        ACTIVE   -       -
sv nfsvol-S01   nfsvol-03   ENABLED  102400   0        -        -       -
sv nfsvol-S02   nfsvol-03   ENABLED  102400   0        -        -       -

v  nfsvol-L01   fsgen       ENABLED  102400   -        ACTIVE   -       -
pl nfsvol-P01   nfsvol-L01  ENABLED  102400   -        ACTIVE   -       -
sd nfs1-02      nfsvol-P01  ENABLED  102400   0        -        -       -
pl nfsvol-P02   nfsvol-L01  ENABLED  102400   -        ACTIVE   -       -
sd nfs2-02      nfsvol-P02  ENABLED  102400   0        -        -       -

v  nfsvol-L02   fsgen       ENABLED  102400   -        ACTIVE   -       -
pl nfsvol-P03   nfsvol-L02  ENABLED  102400   -        ACTIVE   -       -
sd nfs3-02      nfsvol-P03  ENABLED  102400   0        -        -       -
pl nfsvol-P04   nfsvol-L02  ENABLED  102400   -        ACTIVE   -       -
sd nfs4-02      nfsvol-P04  ENABLED  102400   0        -        -       -
```

Examining Hot Relocation

Hot relocation is controlled by the vxrelocd daemon, which is started by default after you initialize the volume manager.

When a mirrored volume breaks under hot relocation, VxVM looks for space to substitute for the broken half of the mirror. If one large disk breaks, VxVM can concatenate together many new subdisks from other disks in the diskgroup to recreate the broken plex. It will *never* use any disk space from disks in the surviving plex.

The SPARE and NOHOTUSE Flags

Under normal hot relocation, disks can be marked with one of the two following flags, but not both:

- SPARE

  Disks marked with this flag are the *preferred* disks to use for hot relocation. These disks can still be used to build normal volumes.
- NOHOTUSE

  Disks marked with this flag are *excluded* from consideration to use for hot relocation.

In the following example, disk nfs2 is set as a preferred disk for hot relocation, and disk nfs1 is excluded from hot relocation usage:

```
# vxedit -g nfsdg set spare=on nfs2
# vxedit -g nfsdg set nohotuse=on nfs1
```

The flag settings will be visible in the output of the vxdisk list and vxprint commands.

Registering VxVM Disk Groups

After you create a new VxVM disk group and volumes, you must manually register the disk group using either the scsetup utility or the scconf command for the disk group to be managed by the cluster. The scsetup utility is recommended.

When a VxVM disk group is registered in the Sun Cluster software environment, it is referred to as a *device group*.

Until a VxVM disk group is registered, the cluster does not detect it. While you can build volumes and perform all normal VxVM administration tasks, you will not be able to use the volume manager devices in the cluster.

If you create a new volume or delete a volume in a VxVM disk group that is already registered with the cluster, you must synchronize the disk device group by using scsetup. Such configuration changes include adding or removing volumes, as well as changing the group, owner, or permissions of existing volumes. Synchronization after volumes are created or deleted ensures that the global namespace is in the correct state.

The following example shows a disk group that is known to VxVM, but not yet known to the cluster:

```
# vxdg list
NAME         STATE           ID
rootdg       enabled             1120070264.6.vincent
nfsdg        enabled,cds         1120071527.33.vincent
```

```
# scstat -D

-- Device Group Servers --

                         Device Group        Primary
Secondary
                         ------------        -------             -------


-- Device Group Status --

                              Device Group        Status
                              ------------        ------



-- Multi-owner Device Groups --

                              Device Group        Online Status
                              ------------        -------------
```

Note – Multiowner device groups are a feature of Solaris Volume Manager that lets multiple nodes running ORACLE RAC simultaneously access the drives. It is Solaris Volume Manager's equivalent to VXVM's Cluster Volume Manager (CVM) feature.

Using the `scconf` Command to Register Disk Groups

The following command registers a VxVM disk group as a cluster device group:

```
# scconf -a -D type=vxvm,name=nfsdg,\
nodelist=vincent,theo,\
preferenced=true,failback=disabled
```

This command uses the following parameters:

- The **`nodelist`** should contain all the nodes and *only* the nodes physically connected to the disk group’s disks.
- The **`preferenced=true/false`** affects whether the nodelist indicates an order of failover preference. On a two node cluster this option is only meaningful if failback is enabled.
- The **`failback=disabled/enabled`** affects whether a preferred node “takes back” its device group when it joins the cluster. The default value is disabled. If it is enabled, then you must also have `preferenced` set to `true`.

Using the `scsetup` Command to Register Disk Groups

The `scsetup` utility can guide you through the previous options in a menu-driven environment. From the main menus, selecting Menu Option 5 (Device Groups and Volumes) gives you the following submenu:

```
*** Device Groups Menu ***

    Please select from one of the following options:

        1) Register a VxVM disk group as a device group
        2) Synchronize volume information for a VxVM device group
        3) Unregister a VxVM device group
        4) Add a node to a VxVM device group
        5) Remove a node from a VxVM device group
        6) Change key properties of a device group

        ?) Help
        q) Return to the Main Menu
```

Verifying and Controlling Registered Device Groups

Use the `scstat -D` command to verify registered device groups:

```
# scstat -D
-- Device Group Servers --

                         Device Group        Primary Secondary
                         ------------        ------- --------
  Device group servers:  nfsdg               vincent theo


-- Device Group Status --

                              Device Group        Status
                              ------------        ------
  Device group status:        nfsdg               Online


-- Multi-owner Device Groups --

                              Device Group        Online Status
                              ------------        -------------
```

Note – By default, even if there are more than two nodes in the `nodelist` for a device group, only one node shows up as secondary. If the primary fails, the secondary becomes primary and another (spare) node will become secondary.

There is another parameter, `numsecondaries=#`, that when used on the `scconf` command line allows you to have more than one secondary.

When VxVM disk groups are registered as Sun Cluster software device groups and have the status Online, *never* use the `vxdg import` and `vxdg deport` commands to control ownership of the disk group. This will cause the cluster to treat the device group as failed.

Instead, use the following command syntax to control disk group ownership:

```
# scswitch -z -D nfsdg -h node_to_switch_to
```

Managing VxVM Device Groups

The `scconf -c` command can be used to perform cluster-specific changes to VxVM device groups.

Resynchronizing Device Groups

After a VxVM device group is registered with the cluster, it must be *resynchronized* any time a new volume is created or deleted in the disk group. This instructs the Sun Cluster software to scan the disk group and build and remove the appropriate global device files:

```
# scconf -c -D name=nfsdg,sync
```

Making Other Changes to Device Groups

The properties of existing VxVM device groups can also be changed. For example, the `failback` and `preferenced` properties of a group can be modified after it is registered.

```
# scconf -c -D \
name=nfsdg,preferenced=false,failback=disabled
```

Using the Maintenance Mode

You can take a VxVM device group "out of service," as far as the cluster is concerned, for emergency repairs.

To put the device group in maintenance mode, all of the VxVM volumes must be *unused* (unmounted or otherwise not open). Then you can issue the following:

```
# scswitch -m -D nfsdg
```

It is rare that you will ever want to do this because almost all repairs can still be done while the device group is in service.

To come back out of maintenance mode, just switch the device group onto a node as in the following:

```
# scswitch -z -D nfsdg -h new_primary_node
```

Using Global and Failover File Systems on VxVM Volumes

Sun Cluster 3.1 supports running data services on the following categories of file systems:

- *Global file systems* – Accessible to all cluster nodes simultaneously, even those not physically connected to the storage
- *Failover file systems* – Mounted only on the node running the failover data service, which must be physically connected to the storage

The file system type can be UFS or VxFS, regardless of whether you are using global or failover file systems. These examples and the exercises assume you are using UFS.

Creating File Systems

The distinction between global and failover file system is *not* made at the time of file system creation. Use `newfs` as normal to create a UFS file system for the cluster on a *registered* disk group:

```
# newfs /dev/vx/rdsk/nfsdg/nfsvol
```

Mounting File Systems

The distinction between global and failover file system is made in the `/etc/vfstab` "mount-at-boot" and "options" columns.

A global file system entry should look like the following, and it should be identical on all nodes who may run services which access the file system (including nodes not physically connected to the storage):

```
/dev/vx/dsk/nfsdg/nfsvol /dev/vx/rdsk/nfsdg/nfsvol /global/nfs ufs 2 yes global,logging
```

A local failover file system entry looks like the following, and it should be identical on all nodes who may run services which access the file system (can only be nodes physically connected to the storage):

```
/dev/vx/dsk/nfsdg/nfsvol /dev/vx/rdsk/nfsdg/nfsvol /localnfs ufs 2 no logging
```

Note – The `logging` option is the default starting in Solaris 9 OS 9/04 (Update 7) and continuing to Solaris 10 OS. It is *always* the default if you have the `global` option.

Mirroring the Boot Disk With VxVM

Assuming you have encapsulated your boot drive with VxVM, mirroring the boot drive is a simple procedure which can be undertaken any time while the cluster is online, without any reboot.

1. Initialize and add another local drive to `bootdg`, if this has not been done yet. This drive must be at least as big as the boot drive. Give the drive the VxVM logical name `rootmir`, for example. This is done as part of the `vxdiskadd` dialogue.

   Make sure you *do not* make the disk a CDS disk (choose `sliced`)

   ```
   # vxdiskadd c0t8d0
   ```

2. Use the `vxmirror` command, or the `vxdiskadm` "Mirror volumes on a disk" option to mirror all the volumes on the boot drive.

Caution – Do *not* mirror each volume separately with `vxassist`. This will leave you in a state that the mirrored drive is "unencapsulatable." That is, if you ever need to unencapsulate the second disk you will not be able to.

If you mirror the root drive correctly as shown, the second drive becomes just as "unencapsulatable" as the original. That is because correct "restricted subdisks" are laid down onto the new drive. These can later be turned into regular Solaris OS partitions because they are on cylinder boundaries.

```
# vxmirror -g bootdg rootdg_1 rootmir
.
.
```

3. The previous command also creates aliases for both the original root partition and the new mirror. You will need to manually set your `boot-device` variable so you can boot off both the original and the mirror.

   ```
   # eeprom|grep vx
   devalias vx-rootdg_1 /pci@1f,4000/scsi@3/disk@0,0:a
   devalias vx-rootmir /pci@1f,4000/scsi@3/disk@8,0:a

   # eeprom boot-device="vx-rootdg_1 vx-rootmir"
   ```

4. Verify the system has been instructed to use these device aliases on boot:

   ```
   # eeprom|grep use-nvramrc
   use-nvramrc?=true
   ```

Exercise: Configuring Volume Management

In this exercise, you complete the following tasks:

- Task 1 – Selecting Disk Drives
- Task 2 – Using the `installvm` Utility to Install and Initialize VxVM Software
- Task 3 – Installing the VxVM Patch
- Task 4 – Using `scvxinstall` to Verify the `vxio` Major Number
- Task 5 – Adding `vxio` on Any Non-Storage Node on Which You Have Not Installed VxVM
- Task 6 – Rebooting All Nodes
- Task 7 – Configuring Demonstration Volumes
- Task 8 – Registering Demonstration Disk Groups
- Task 9 – Creating a Global `nfs` File System
- Task 10 – Creating a Global `web` File System
- Task 11 – Testing Global File Systems
- Task 12 – Managing Disk Device Groups
- Task 13 – Viewing and Managing VxVM Device Groups Through SunPlex Manager
- Task 14 (Optional) – Encapsulating the Boot Disk on a Cluster Node

Preparation

Record the location of the VxVM software you will install during this exercise.

Location: ______________________________

During this exercise, you create two data service disk groups. Each data service disk group contains a single mirrored volume. Encapsulating the boot disk is an optional exercise at the end. The setup is shown in Figure 6-9.

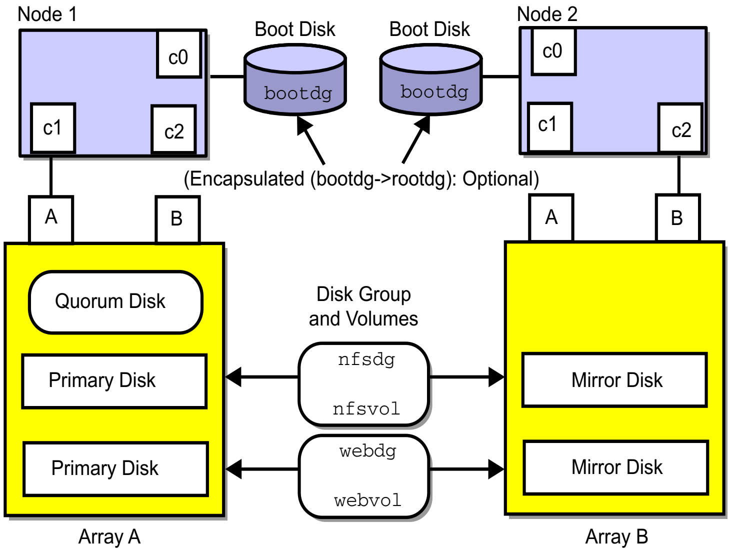


Figure 6-9 Configuring Volume Management

Note – During this exercise, when you see italicized names, such as ***IPaddress***, ***enclosure_name***, ***node1***, or ***clustername*** embedded in a command string, substitute the names appropriate for your cluster.

Task 1 – Selecting Disk Drives

Before proceeding with this exercise, you must have a clear picture of the disk drives that are used throughout this course. You have already configured one disk drive as a quorum disk. It is perfectly acceptable to use this as one of your disk group drives as well. In this exercise, you must identify the boot disk and two disks in each storage array for use in the two demonstration disk groups, `nfsdg` and `webdg`.

Perform the following step to select a disk drive:

Use the `scdidadm -L` and `format` commands to identify and record the logical addresses of disks for use during this exercise. Use the address format: `c0t2d0`.

Quorum disk: _________

Node 1 Node 2

Boot disks: _________ _________

Array A Array B

`nfsdg` **disks:** _________ _________

`webdg` **disks:** _________ _________

Task 2 – Using the `installvm` Utility to Install and Initialize VxVM Software

Perform the following steps to install and initialize the VxVM software on your cluster host systems. You need to install VxVM on each node that is physically connected to the shared storage. (You can optionally install it on a non-storage node).

1. Choose a cluster node from which to drive the installation.
2. On *other nodes*, edit the `/.rhosts` file to enable incoming `rsh/rcp.`
3. On the node you want to drive from, run the installer:

```
# cd vxvm_4.1_software_location/volume_manager
# ./installvm
```

4. Respond to the installvm as follows:
   a. Enter the names of all nodes on which you want to install VxVM, including the one you are driving from.
   b. When it asks about optional packages, choose option (3) to view the descriptions one by one. Choose to install the man pages (VRTSvmman). You do not need any other optional packages.
   c. Choose to install all nodes simultaneously.
   d. Enter the license information as directed by your instructor (if you are using A5x00 storage you need no further license).
   e. Choose to have the installer complete the setup for you.
   f. Answer No when asked about enclosure-based naming.
   g. Answer No when asked about a default disk group.
   h. Do not reboot yet (you will do this later).

Task 3 – Installing the VxVM Patch

Do the following on all nodes on which you have installed VxVM, one at a time:

```
# cd vxvm_patch_location
# patchadd 117080-xy
```

Task 4 – Using scvxinstall to Verify the vxio Major Number

Perform the following steps on all nodes on which you have installed VxVM, one at a time.

1. Run scvxinstall and choose *not* to encapsulate root.
2. Let scvxinstall verify the vxio major number.

Task 5 – Adding `vxio` on Any Non-Storage Node on Which You Have Not Installed VxVM

If you have a non-storage node on which you have not installed VxVM, do the following:

Edit `/etc/name_to_major` and add a line at the bottom containing the same `vxio` major number as used on the storage nodes:

`vxio` *`same_major_number_as_other_nodes`*

Task 6 – Rebooting All Nodes

Reboot all your nodes. In the real production environment, you would always do this one node at a time to maintain the high availability of any cluster services that might already be configured.

Task 7 – Configuring Demonstration Volumes

Perform the following steps to configure two demonstration disk groups, with each containing a single mirrored volume:

1. On Node 1, create the `nfsdg` disk group with your previously selected logical disk addresses.

```
# vxdiskadd c#t#d# c#t#d#
Which disk group [<group>,none,list,q,?] (default: none) nfsdg
Create a new group named nfsdg? [y,n,q,?] (default: y) y
Create the disk group as a CDS disk group? [y,n,q,?] (default: y) y
Use default disk names for these disks? [y,n,q,?] (default: y) y
Add disks as spare disks for nfsdg? [y,n,q,?] (default: n) n
Exclude disks from hot-relocation use? [y,n,q,?] (default: n) y
.
.
Enter desired private region length
[<privlen>,q,?] (default: 2048) 2048
```

Caution – If you are prompted about encapsulating the disk, you should reply **`no`**. If you are prompted about clearing old disk usage status from a previous training class, you should reply **`yes`**. In your work environment, be careful when answering these questions because you can destroy critical data or cluster node access information.

2. Verify the status of the disk group and the names and ownership of the disks in the `nfsdg` disk group.

   ```
   # vxdg list
   # vxdisk list
   ```

3. On Node 1, verify that the new `nfsdg` disk group is globally linked.

   ```
   # ls -l /dev/vx/dsk/nfsdg
   lrwxrwxrwx   1 root     root          12 Dec  6  2002
   /dev/vx/dsk/nfsdg ->
   /global/.devices/node@1/dev/vx/dsk/nfsdg
   ```

4. On Node 1, create a 500-Mbyte mirrored volume in the `nfsdg` disk group.

   ```
   # vxassist -g nfsdg make nfsvol 500m layout=mirror
   ```

5. On Node 1, create the `webdg` disk group with your previously selected logical disk addresses. Answers in the dialog can be similar to those in step 1.

   ```
   # vxdiskadd c#t#d# c#t#d#
   Which disk group [<group>,none,list,q,?]
   (default: none) webdg
   ```

6. On Node 1, create a 500-Mbyte mirrored volume in the `webdg` disk group.

   ```
   # vxassist -g webdg make webvol 500m layout=mirror
   ```

7. Type the `vxprint` command on both nodes. Notice that Node 2 does not see the disk groups created and still imported on Node 1.

8. Issue the following command on the node that currently owns the disk group:

   ```
   # newfs /dev/vx/rdsk/webdg/webvol
   ```

   It should fail with a `no such device or address` error.

Task 8 – Registering Demonstration Disk Groups

Perform the following steps to register the two new disk groups with the Sun Cluster framework software:

1. On Node 1, use the `scconf` utility to manually register the `nfsdg` disk group.

```
# scconf -a -D type=vxvm,name=nfsdg,\
nodelist=node1:node2,preferenced=true,failback=enabled
```

Note – Put the local node (Node 1) first in the node list.

2. On Node 1, use the `scsetup` utility to register the `webdg` disk group.

```
# scsetup
```

3. From the main menu, complete the following steps:
   a. Select option 5, “Device groups and volumes.”
   b. From the device groups menu, select option 1, “Register a VxVM disk group as a device group.”

4. Answer the `scsetup` questions as follows:

```
Name of the VxVM disk group you want to register? webdg
Do you want to configure a preferred ordering (yes/no)
[yes]? yes

Are both nodes attached to all disks in this group
(yes/no) [yes]?  yes
```

Note – Read the previous question carefully. On a cluster with more than two nodes it will ask if *all* nodes are connected to the disks. In a “Pair +1” configuration, for example, you need to answer **`no`** and let it ask you which nodes are connected to the disks.

```
Which node is the preferred primary for this device
group? node2

Enable "failback" for this disk device group (yes/no)
[no]?  yes
```

Note – Make sure you specify *`node2`* as the preferred primary node. You might see warnings about disks configured by a previous class that still contain records about a disk group named `webdg`. These warnings can be safely ignored.

5. From either node, verify the status of the disk groups.

```
# scstat -D
```

Task 9 – Creating a Global nfs File System

Perform the following steps on Node 1 to create and mount a demonstration file system on the nfsdg disk group volume:

1. On Node 1, create a file system on nfsvol in the nfsdg disk group.

```
# newfs /dev/vx/rdsk/nfsdg/nfsvol
```

2. On *all nodes*, including any non-storage nodes, create a global mount point for the new file system.

```
# mkdir /global/nfs
```

On all nodes, add a mount entry in the /etc/vfstab file for the new file system with the global and logging mount options.

```
/dev/vx/dsk/nfsdg/nfsvol /dev/vx/rdsk/nfsdg/nfsvol \
/global/nfs  ufs 2 yes global,logging
```

Note – Do not use the line continuation character (\) in the vfstab file.

3. On Node 1, mount the /global/nfs file system.

```
# mount /global/nfs
```

4. Verify that the file system is mounted and available on *all* nodes.

```
# mount
# ls /global/nfs
lost+found
```

Task 10 – Creating a Global web File System

Perform the following steps on Node 2 to create and mount a demonstration file system on the webdg disk group volume:

1. On Node 2, create a file system on webvol in the webdg disk group.

```
# newfs /dev/vx/rdsk/webdg/webvol
```

2. On *all nodes*, create a global mount point for the new file system.

```
# mkdir /global/web
```

3. On *all nodes*, add a mount entry in the `/etc/vfstab` file for the new file system with the `global` and `logging` mount options.

```
/dev/vx/dsk/webdg/webvol /dev/vx/rdsk/webdg/webvol \
/global/web ufs 2 yes global,logging
```

Note – Do not use the line continuation character (\) in the `vfstab` file.

4. On Node 2, mount the `/global/web` file system.

```
# mount /global/web
```

5. Verify that the file system is mounted and available on *all* nodes.

```
# mount
# ls /global/web
lost+found
```

Task 11 – Testing Global File Systems

Perform the following steps to confirm the general behavior of globally available file systems in the Sun Cluster 3.1 software environment:

1. On Node 2, move into the `/global/nfs` file system.

```
# cd /global/nfs
```

2. On Node 1, try to unmount the `/global/nfs` file system (**`umount /global/nfs`**). You should get an error that the file system is busy.
3. On Node 2, move out of the `/global/nfs` file system (**`cd /`**) and try to unmount it again on Node 1.
4. Mount the `/global/nfs` file system again on Node 1.
5. Try unmounting and mounting `/global/nfs` from all nodes.

Task 12 – Managing Disk Device Groups

In the Sun Cluster 3.1 software environment, VERITAS disk groups become *disk device groups* when they are registered. In most cases, they should *not* be managed using VERITAS commands. Some administrative tasks are accomplished using a combination of Sun Cluster and VERITAS commands. Common tasks are:

- Adding volumes to a disk device group
- Removing volumes from a disk device group

Adding a Volume to a Disk Device Group

Perform the following steps to add a volume to an existing device group:

1. Make sure the *device group* is online (to the Sun Cluster software).

   ```
   # scstat -D
   ```

Note – You can bring it online to a selected node as follows:

```
        # scswitch -z -D nfsdg -h node1
```

2. On the node that is primary for the device group, create a 50-Mbyte test volume in the nfsdg disk group.

   ```
   # vxassist -g nfsdg make testvol 50m layout=mirror
   ```

3. Verify the status of the volume testvol.

   ```
   # vxprint -g nfsdg testvol
   ```

4. Perform the following steps to register the changes to the nfsdg disk group configuration:

   a. Start the scsetup utility.

   b. Select menu item 5, “Device groups and volumes.”

   c. Select menu item 2, “Synchronize volume information.”

   d. Supply the name of the disk group and quit the scsetup utility when the operation is finished.

Note – The command-line equivalent is:

```
        scconf -c -D name=nfsdg,sync
```

Removing a Volume From a Disk Device Group

To remove a volume from a disk device group, perform the following steps on the node that currently has the related disk group imported:

1. Unmount any file systems that are related to the volume.
2. On Node 1, recursively remove the test volume, testvol, from the nfsdg disk group.

   ```
   # vxedit -g nfsdg -rf rm testvol
   ```

3. Synchronize the `nfsdg` disk group configuration change with the Sun Cluster software framework.

Note – You can use either the `scsetup` utility or `scconf` as follows:

```
scconf -c -D name=nfsdg,sync
```

Migrating Device Groups

The `scconf -p` command is the best method of determining current device group configuration parameters. Perform the following steps to verify device group behavior:

1. Verify the current demonstration device group configuration.

```
# scconf -p |grep group
Device group name:                                  nfsdg
  Device group type:                                VxVM
  Device group failback enabled:                    yes
  Device group node list:                     node1,node2
  Device group ordered node list:                   yes
  Device group desired number of secondaries:       1
  Device group diskgroup name:                   nfsdg

Device group name:                                   webdg
  Device group type:                                 VxVM
  Device group failback enabled:                     yes
  Device group node list:                     node2,node1
  Device group ordered node list:                    yes
  Device group desired number of secondaries:          1
  Device group diskgroup name:                    webdg
```

2. From either node, switch the `nfsdg` device group to Node 2.

```
# scswitch -z -D nfsdg -h node2 (use your node name)
```

Type the **`init 0`** command to shut down Node 1.

3. Boot Node 1. The `nfsdg` disk group should automatically migrate back to Node 1.

4. Disable the device group failback feature. It is inconsistent with how the device groups work with the applications we configure in Modules 9 and 10:

```
# scconf -c -D name=nfsdg,failback=disabled
# scconf -c -D name=webdg,failback=disabled
# scconf -p | grep group
```

Task 13 – Viewing and Managing VxVM Device Groups Through SunPlex Manager

In this task you view and manage VxVM device groups through SunPlex Manager. Perform the following steps on your administration workstation or display station.

1. In a web browser, log in to Sun Java Web Console on any cluster node:

   https://*nodename*:6789
2. Enter the SunPlex Manager Application.
3. Click on the `Shared Storage` folder on the left.
4. Investigate the status information and graphical topology information that you can see regarding VxVM device groups.
5. View the details of a device group by clicking on its name.
6. Use the Switch Primaries button to switch the primary for one of your device groups.

Task 14 (Optional) – Encapsulating the Boot Disk on a Cluster Node

If you choose to, on one of your cluster nodes, you can encapsulate the root disk. Recall that you will no longer be able to have a logging root file system. The encapsulation process automatically places `nologging` in the options field for your root file system in the `/etc/vfstab`.

Normally, after encapsulation and reboot, you would mirror your OS disk (there is no other reason to encapsulate root).

For the purposes of this class, you might like to leave a local (non-shared) disk free so that you can still perform the Solaris Volume Manager exercises in the next module.

If you do this task on just one node, you could leave both local disks on the other node completely free for Solaris Volume manger.

On any node on which you want to encapsulate root (do them one at time and wait for reboot if you want to do it on more than one node):

1. Run `scvxinstall` (yes, again) and choose to encapsulate the boot disk.
2. Let `scvxinstall` reboot your node for you. It will end up rebooting twice.
3. Verify that the OS has successfully been encapsulated.

   ```
   # df -k
   # swap -l
   # grep /dev/vx /etc/vfstab
   ```

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 7

Managing Volumes With Solaris™ Volume Manager (Solstice DiskSuite™ Software)

Objectives

Upon completion of this module, you should be able to:

- Explain Solaris Volume Manager and Solstice DiskSuite software
- Describe the most important concepts of Solaris Volume Manager
- Describe Solaris Volume Manager soft partitions
- Differentiate between shared disksets and local disksets
- Describe Solaris Volume Manager multiowner disksets
- Describe volume database (`metadb`) management issues and the role of Solaris Volume Manager mediators
- Install and Configure the Solaris Volume Manager software
- Create the local `metadbs`
- Add disks to shared disksets
- Use shared diskset disk space
- Build Solaris Volume Manager mirrors with soft partitions in disksets
- Use Solaris Volume Manager status commands
- Use hot spare pools in the disk set
- Perform Sun Cluster software level device group management
- Perform cluster-specific changes to Solaris Volume Manager device groups
- Create global file systems
- Mirror the boot disk with Solaris Volume Manager

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- Is it easier to manage Solaris Volume Manager devices with or without soft partitions?
- How many different collections of `metadbs` will you need to manage?
- What is the advantage of using DID devices as Solaris Volume Manager building blocks in the cluster?
- How do Solaris Volume Manager disksets get registered in the cluster?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Microsystems, Inc. *Solaris Volume Manager Administration Guide,* part number 816-4520.

Describing Solaris Volume Manager and Solstice DiskSuite Software

Solaris 9 OS and Solaris 10 OS integrate the Solaris Volume Manager into the base Solaris OS. If you are running Solaris 8 OS in the Sun Cluster 3.1 software environment, installing Solstice DiskSuite 4.2.1 software with patch 108693-06 or higher gives you the equivalent capabilities of Solaris Volume Manager. This includes soft partitioning, which is described in the module and used in the labs. You must have a SunSolve^{SM} program account to access this patch.

In the Solstice DiskSuite software, virtual volume structures have always been referred to as *metadevices*.

As its new name indicates, the Solaris Volume Manager prefers the use of the word *volumes* to refer to the same structures. This module follows this latter convention.

Viewing Solaris Volume Manager in the Sun Cluster Software Environment

This module does not intend to replace the full three-day course in disk management using Solaris Volume Manager. Instead, this module briefly introduces the most important concepts of Solaris Volume Manager, and then focuses on the issues for using Solaris Volume Manager to manage disk space in the cluster environment. Among the topics highlighted in this module are:

- Using Solaris Volume Manager with and without soft partitions
- Identifying the purpose of Solaris Volume Manager disksets in the cluster
- Managing local diskset database replicas (`metadb`) and shared diskset `metadb`s
- Initializing Solaris Volume Manager to build cluster disksets
- Building cluster disksets
- Mirroring with soft partitions in cluster disksets
- Encapsulating and mirroring the cluster boot disk with Solaris Volume Manager

Exploring Solaris Volume Manager Disk Space Management

Solaris Volume Manager has two distinct ways of managing disk space. Until patch 108693-06, the Solstice DiskSuite software (Solaris Volume Manager's precursor product) had the restriction that Solaris OS partitions were the smallest granularity building blocks for volumes. The current Solaris Volume Manager supports both the traditional way of managing space and the new method called *soft partitioning*.

Solaris Volume Manager Partition-Based Disk Space Management

Figure 7-1 demonstrates Solaris Volume Manager disk space management that equates standard disk partitions with building blocks for virtual volumes.

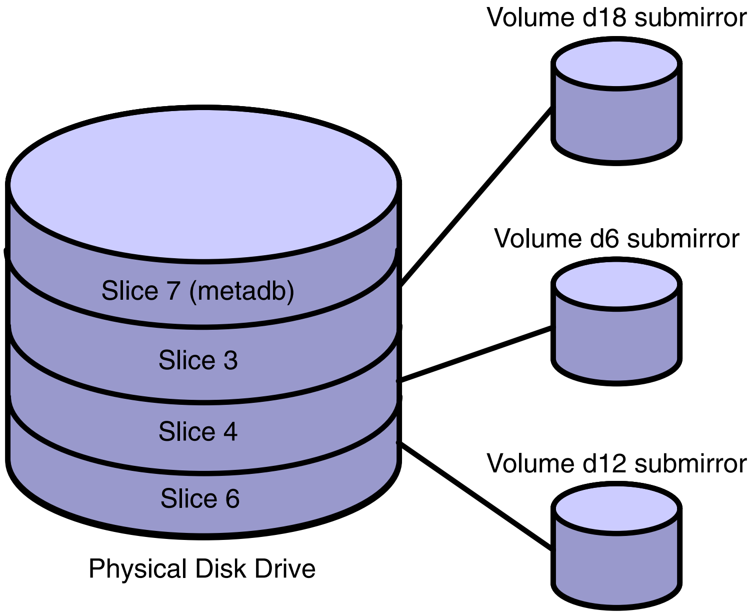


Figure 7-1 Solstice Volume Manager Space Management

The following are limitations with using only partitions as Solaris Volume Manager building blocks:

- The number of building blocks per disk or LUN is limited to the traditional seven (or eight, if you want to push it) partitions. This is particularly restrictive for large LUNs in hardware RAID arrays.
- Disk repair is harder to manage because an old partition table on a replacement disk must be recreated or replicated manually.

Solaris Volume Manager Disk Space Management With Soft Partitions

Soft partitioning supports the creation of *virtual partitions* within Solaris OS disk partitions or within other volumes. These virtual partitions can range in size from one block to the size of the base component. You are limited in the number of soft partitions only by the size of the base component.

Soft partitions add an enormous amount of flexibility to the capability of Solaris Volume Manager:

- They allow you to manage volumes whose size meets your exact requirements without physically repartitioning Solaris OS disks.
- They allow you to create more than seven volumes using space from a single large drive or LUN.
- They can grow as long as there is space left inside the parent component. The space does not have to be physically contiguous. Solaris Volume Manager will find whatever space it can (in the same parent component) to allow your soft partition to grow.

There are two ways to deal with soft partitions in Solaris Volume Manager:

- Make soft partitions on disk partitions, and then use those to build your submirrors as shown in Figure 7-2.

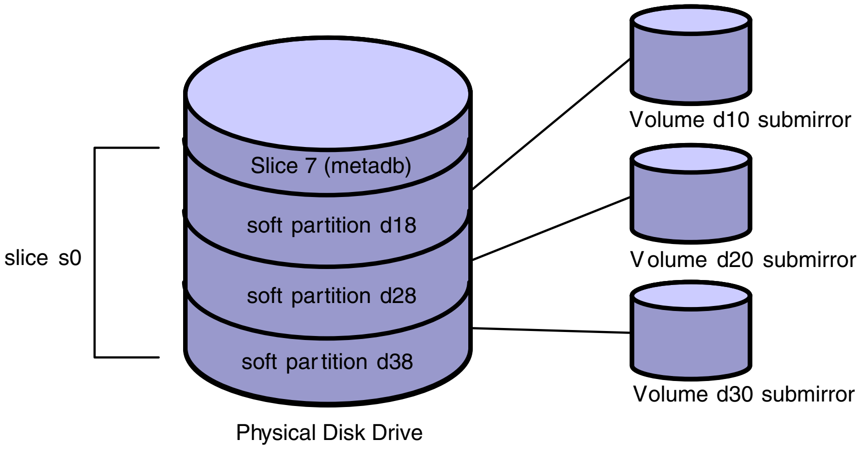


Figure 7-2 Building Submirrors From Soft Partitions

- Build mirrors using entire partitions, and then use soft partitions to create the volumes with the size you want as shown in Figure 7-3.

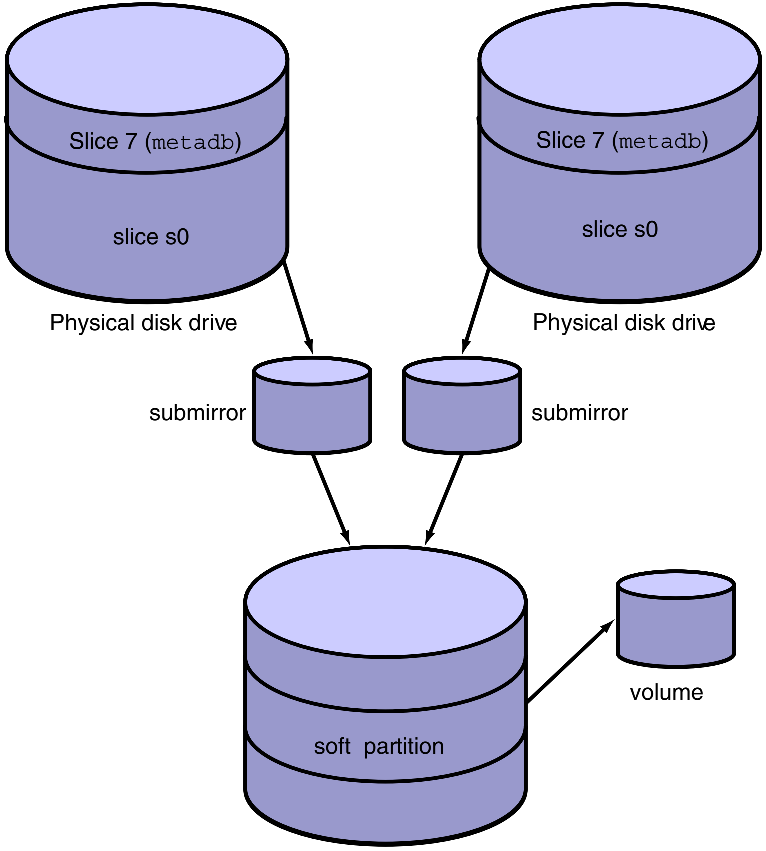


Figure 7-3 Using Soft Partitions to Create Volumes

All the examples later in this module use the second strategy. The advantages of this are the following:

- Volume management is simplified.
- Hot spare management is consistent with how hot spare pools work.

The only disadvantage of this scheme is that you have large mirror devices even if you are only using a small amount of space. This slows down resynching of mirrors.

Exploring Solaris Volume Manager Disksets

When using Solaris Volume Manager to manage data in the Sun Cluster 3.1 software environment, all disks that hold the data for cluster data services must be members of Solaris Volume Manager shared disksets.

Only disks that are physically located in the multiported storage will be members of the shared disksets. Only disks that are in the same diskset operate as a unit. They can be used together to build mirrored volumes, and primary ownership of the diskset transfers as a whole from node to node. You will always have disks from different controllers (arrays) in the same diskset, so that you can mirror across controllers.

Shared disksets are given a name that often reflects the intended usage of the diskset (for example, `nfsds`).

To create any shared disksets, Solaris Volume Manager requires that each host (node) must have a local diskset on non-shared disks. The only requirement for these local disks is that they have local diskset `metadbs`, which are described later. It is likely, however, that if you are using Solaris Volume Manager for your data service data, that you will also mirror your boot disk using Solaris Volume Manager. Figure 7-4 shows the contents of two shared disksets in a typical cluster, along with the local disksets on each node.

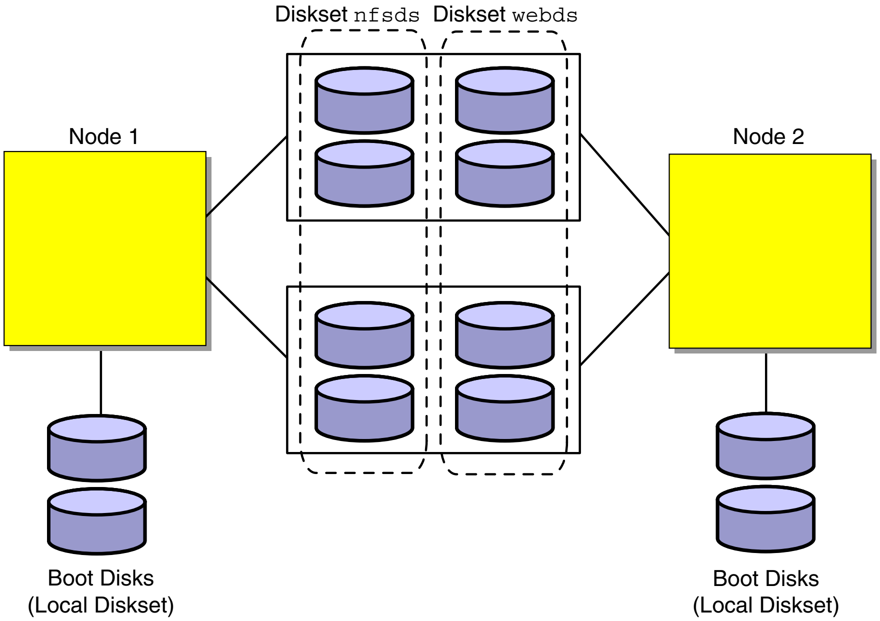


Figure 7-4 Two Shared Disksets

Solaris Volume Manager Multiowner Disksets (for ORACLE RAC)

Starting in Solaris 9 OS 9/04 (Update 7) and Sun Cluster 3.1 9/04 (Update 3), Solaris Volume Manager has a multiowner diskset feature that is analogous to the VERITAS Cluster Volume Manager (CVM) feature. That is, a multiowner diskset allows more than one node to physically access the storage, simultaneously.

In the current implementation, multiowner disksets are *only* for use for with Oracle RAC software. In fact, you cannot create a multiowner diskset until you have enabled a layer of software underlying ORACLE RAC known as the RAC framework. This is discussed in more detail, and you will get a chance to work with it, if you choose, in one of the optional exercises in Module 11.

Managing multiowner disksets is identical to managing other disksets, except that you use a `-M` option when you create the diskset.

Using Solaris Volume Manager Database Replicas (metadb)

Solaris Volume Manager requires that certain partitions serve as volume database replicas, storing the Solaris Volume Manager configuration and state information in a raw (non-file system) format. These are normally small dedicated Solaris OS partitions. Formerly, in the local disksets, Solaris Volume Manager allowed the creation of metadbs on a large partition and then using the same partition as a component in a volume. Reasons why you would never want to do this on the boot disks are described later in this module.

There are separate sets of metadbs for the local disksets (local metadbs) and for each shared diskset. For example, in Figure 7-4 on page 7-10 (which shows two nodes and two shared disksets), there are *four* distinct collections of metadbs.

It is possible to put several copies of the metadb on the same partition. You might do this to balance the numbers of metadbs across disks or partitions, for reasons described in the following subsections.

Local Replicas Management

When managing local replicas:

- You must add local replicas manually.
- You can put local replicas on any dedicated partitions on the local disks. You might prefer to use slice 7 (s7) as a convention, because the shared diskset replicas have to be on that partition.
- You must spread local replicas evenly across disks and controllers.
- If you have less than three local replicas, Solaris Volume Manager logs warnings. You can put more than one copy in the same partition to satisfy this requirement. For example, with two local disks, set up a dedicated partition on each disk and put three copies on each disk.

Local Replica Mathematics

The math for local replicas is as follows:

- If less than 50 percent of the defined metadb replicas are available, Solaris Volume Manager will cease to operate.
- If less than or equal to 50 percent of the defined metadb replicas are available at boot time, you *cannot* boot the node. However, you can boot to single user mode, and just use the metadb command to delete ones that are not available.

Shared Diskset Replicas Management

Consider the following issues when managing shared diskset replicas:

- There are separate metadb replicas for each diskset.
- They are automatically added to disks as you add disks to disk sets. They will be, and must remain, on slice 7.
- You *will* have to use the metadb command to remove and add replicas if you repair a disk containing one (remove broken ones and add them back onto replaced disks).

Shared Diskset Replica Quorum Mathematics

The math for shared diskset replica quorums is as follows:

- If less than 50 percent of the defined replicas for a diskset are available, the diskset will cease to operate.
- If less than or equal to 50 percent of the defined replicas for a diskset are available, the diskset cannot be “taken” or switched over.

Shared Diskset Mediators

When you have two nodes connected to two storage arrays, the implication of the replica mathematics described in the previous section is that your diskset can keep operating, but can not transfer primary control from node to node.

This is unacceptable in the Sun Cluster software environment, because it can take a while to fix a broken array or controller, and you still want to be able to gracefully survive a node failure during that period.

The Sun Cluster 3.1 software environment includes special add-ons to Solaris Volume Manager called *mediators*. Mediators allow you to identify the nodes themselves as “tie-breaking votes” in the case of failure of exactly 50 percent of the `metadbs` of a shared diskset. The mediator data is stored in the memory of a running Solaris OS process on each node. If you lose an array, the node mediators will transfer to “golden” status, indicating they count as two extra votes for the shared diskset quorum mathematics. This allows you to maintain normal diskset operations with exactly 50 percent of the `metadbs` surviving. You can also lose a node at this point (you would still have one tie-breaking, “golden” mediator).

Installing Solaris Volume Manager and Tuning the md.conf File

In Solaris 9 OS and Solaris 10 OS, the packages SUNWmdr and SUNWmdu are part of the base operating system.

In Solaris 8 OS these packages must be installed manually. Make sure you also install patch 108693-06 or later so the soft partitioning feature is enabled.

Support for shared diskset mediators is in the package SUNWmdm. This package is automatically installed as part of the cluster framework.

Modifying the md.conf File (Solaris 8 and 9 OS Only)

Based on your planned implementation, you might need to update Solaris Volume Manager's kernel configuration file, /kernel/drv/md.conf. There are two variables that might need to be updated. These maximums include your local disksets. The modifications are summarized in Table 7-1.

Table 7-1 Modifications to the md.conf File

Variable	Default Value	Description
nmd	128	The maximum number of volumes. Solaris Volume Manager uses this setting to limit the *names* of the volumes as well. Note that setting this number too high can use a lot of inodes for device files in your /global/.devices/node@# file system. The maximum value is 8192.
md_nsets	4	The maximum number of disksets. This includes the local diskset. This number should be set to the number of shared disksets you plan to create in your cluster plus one. The maximum value for md_nsets is 32.

Keep this file identical on all nodes of the cluster. Changes to this file take effect after you call devfsadm or perform a boot -r.

Solaris 10 OS has no such limits. Device files for disksets and volumes are created dynamically as required.

Initializing the Local metadbs on Boot Disk and Mirror

No other Solaris Volume Manager management can be done until you initialize the local metadbs on each node.

These instructions assume that you have a small partition to use for the metadbs on your boot disk. In this example, the partition is “s7.”

The instructions presented in the following section assume that you are eventually mirroring your boot drive. As such, they instruct you to partition the eventual boot mirror identically to the boot drive and to add local metadbs there as well.

Make sure you initialize local metadbs correctly and separately on each node.

Using DIDs Compared to Using Traditional c#t#d#

In the Sun Cluster environment, you can add any metadb or partition component either using its cluster Device ID (/dev/did/rdsk/d#s#) or by using the traditional c#t#d#.

Use the DID naming for all shared disksets. Without it you are restricted to having identical controller numbers on each node. You might assume this will always be true, then be surprised when you add a new node or repair a node.

Use the traditional c#t#d# naming scheme for local metadbs and devices. This will make recovery easier in the case that you need to access these structures when booted in non-clustered mode. Omit the /dev/rdsk to abbreviate traditional names in all of the Solaris Volume Manager commands.

Adding the Local metadb Replicas to the Boot Disk

Use the metadb -a command to add local metadbs. As you add the first ones, you must use the -f (force) option. The following example creates three copies:

```
# metadb -a -f -c 3 c0t0d0s7
```

Repartitioning a Mirror Boot Disk and Adding metadb Replicas

This procedure assumes you have a second local disk which has identical geometry to the first. In that case, you can use the fmthard command to replicate the boot disks's partition table onto the mirror:

```
# prtvtoc /dev/rdsk/c0t0d0s0|fmthard -s - /dev/rdsk/c0t8d0s0
fmthard:  New volume table of contents now in place.

# metadb -a -c 3 c0t8d0s7
```

Using the metadb or metadb -i Command to Verify metadb Replicas

The output of the metadb command includes one line for each replica. The -i option adds the legend information about the status abbreviations.

```
# metadb -i
        flags           first blk       block count
    a m  p  luo        16              8192            /dev/dsk/c0t0d0s7
    a    p  luo        8208            8192            /dev/dsk/c0t0d0s7
    a    p  luo        16400            8192            /dev/dsk/c0t0d0s7
    a    p  luo        16              8192            /dev/dsk/c0t8d0s7
    a    p  luo        8208            8192            /dev/dsk/c0t8d0s7
    a    p  luo        16400            8192           /dev/dsk/c0t8d0s7
r - replica does not have device relocation information
 o - replica active prior to last mddb configuration change
 u - replica is up to date
 l - locator for this replica was read successfully
 c - replica's location was in /etc/lvm/mddb.cf
 p - replica's location was patched in kernel
 m - replica is master, this is replica selected as input
 W - replica has device write errors
 a - replica is active, commits are occurring to this replica
 M - replica had problem with master blocks
 D - replica had problem with data blocks
 F - replica had format problems
 S - replica is too small to hold current data base
 R - replica had device read errors
```

Creating Shared Disksets and Mediators

Use the metaset command to create new empty disksets and to add disk drives into an existing diskset. You must use the -a -h options of the metaset command first to create an empty diskset. Then you can add disks. The first host listed as you create a new diskset will be the first one to be the owner of the diskset. You can add mediators with the -a -m options.

All the examples shown in this module of diskset operations will use the DID names for disks instead of the c#t#d# names. You might have to run the scdidadm command often to map between the two.

```
# scdidadm -l c1t3d0
9        vincent:/dev/rdsk/c1t3d0      /dev/did/rdsk/d9
# scdidadm -l d17
17       vincent:/dev/rdsk/c2t3d0      /dev/did/rdsk/d17

# metaset -s nfsds -a -h vincent theo
# metaset -s nfsds -a -m vincent theo

# metaset -s nfsds -a /dev/did/rdsk/d9 /dev/did/rdsk/d17
# metaset

Set name = nfsds, Set number = 1

Host                Owner
  vincent            Yes
  theo

Mediator Host(s)    Aliases
  vincent
  theo

Drive Dbase

d9    Yes

d17   Yes

# metadb -s nfsds
        flags           first blk       block count
     a        u    r    16              8192 /dev/did/dsk/d9s7
     a        u    r    16              8192 /dev/did/dsk/d17s7

# medstat -s nfsds
```

```
Mediator                Status  Golden
vincent                 Ok      No
theo                    Ok      No
```

Mediators become *golden* if exactly 50 percent of the metadbs fail.

Shared Diskset Disk Automatic Repartitioning and `metadb` Placement

When a disk is added to a diskset, it is automatically repartitioned as follows:

- A small portion of the drive (starting at cylinder 0) is mapped slice 7 to be used for state database replicas (usually at least 4 Mbytes in size).
- `metadbs` are added to slice 7 as appropriate to maintain the balance across controllers.
- The rest of the drive is mapped to slice 0 (even slice 2 is deleted).

The drive is *not* repartitioned if the disk already has no slice 2 and slice 7 already has the following characteristics:

- It starts at cylinder 0
- It has at least 4 Mbytes (large enough to hold a state database)
- It has the `V_UNMT` flag set (unmountable flag)
- It is not read-only

Regardless of whether the disk is repartitioned, diskset `metadbs` are added automatically to slice 7 as appropriate as shown in Figure 7-5. If you have exactly two disk controllers you should always add equivalent numbers of disks or LUNs from each controller to each diskset to maintain the balance of `metadbs` in the diskset across controllers.

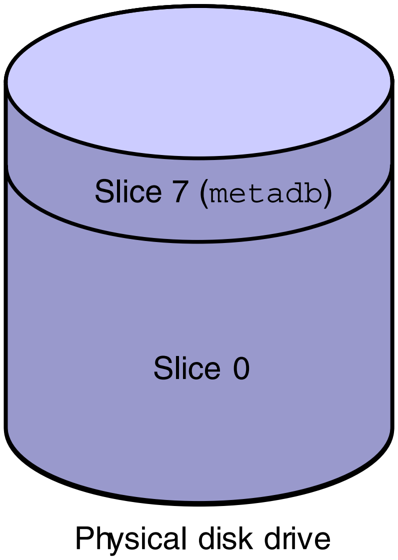


Figure 7-5 Automatically Adding `metadbs` to Slice 7

Using Shared Diskset Disk Space

This section shows the commands that implement the strategy described earlier in the module.

- Always use slice 0 "as is" in the diskset (almost the entire drive or LUN).
- Build mirrors out of slice 0 of disks across two different controllers.
- Use soft partitioning of the large mirrors to size the volumes according to your needs.

Figure 7-6 demonstrates the strategy again in terms of volume (`d#`) and DID (`d#s#`). While Solaris Volume Manager would allow you to use `c#t#d#`, always use DID numbers in the cluster to guarantee a unique, agreed-upon device name from the point of view of all nodes.

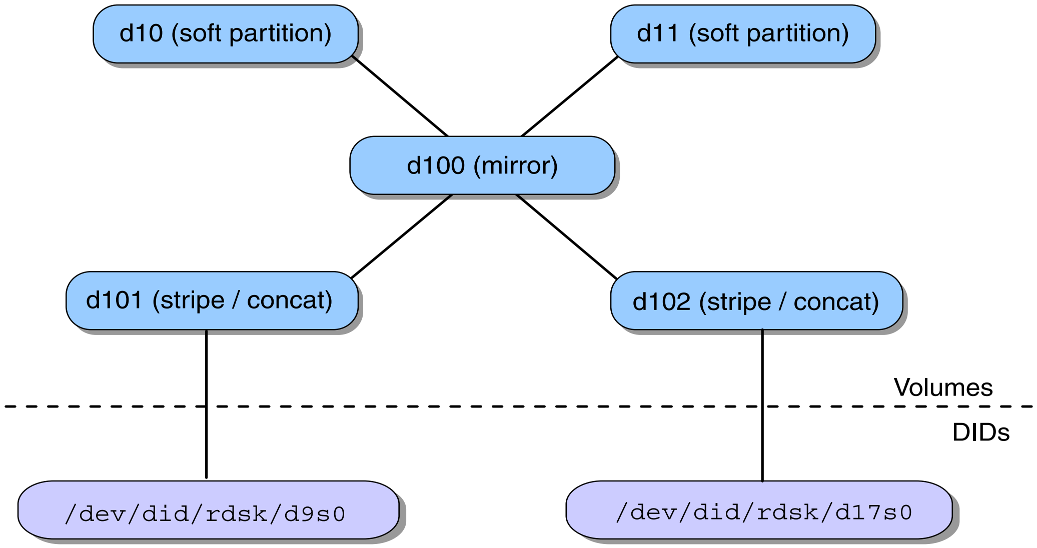


Figure 7-6 Strategy for Building Volumes

Building Volumes in Shared Disksets With Soft Partitions of Mirrors

The following are two ways to indicate which diskset you are referring to for each `metainit` command:

- Use `-s` *`disksetname`* with the command
- Use *`disksetname/d#`* for volume operands in the command.

This module uses the former model for all the examples.

The following is an example of the commands that are used to build the configuration described in Figure 7-6 on page 7-22:

```
# metainit -s nfsds d101 1 1 /dev/did/rdsk/d9s0
nfsds/d101: Concat/Stripe is setup
# metainit -s nfsds d102 1 1 /dev/did/rdsk/d17s0
nfsds/d102: Concat/Stripe is setup
# metainit -s nfsds d100 -m d101
nfsds/d100: Mirror is setup
# metattach -s nfsds d100 d102
nfsds/d100: submirror nfsds/d102 is attached
# metainit -s nfsds d10 -p d100 200m
d10: Soft Partition is setup
# metainit -s nfsds d11 -p d100 200m
d11: Soft Partition is setup
```

The following commands show how a soft partition can be grown as long as there is (even non-contiguous) space in the parent volume. You will see in the output of `metastat` on the next page how both soft partitions contain noncontiguous space, because of the order in which they are created and grown.

```
# metattach -s nfsds d10  400m
nfsds/d10: Soft Partition has been grown
/ metattach -s nfsds d11  400m
nfsds/d11: Soft Partition has been grown
```

Note – The volumes are being increased by 400 Mbytes, to a total size of 600 Mbytes.

Using Solaris Volume Manager Status Commands

There are no commands that display information about volumes or metadbs in multiple disksets at the same time. If you do not specify a diskset name (with -s), you will (possibly inadvertently) get output only about the local diskset on the node on which you entering the command.

Checking Volume Status

The following metastat command output is for the mirrored volume and soft partitions built in the previous example:

```
# metastat -s nfsds
nfsds/d11: Soft Partition
    Device: nfsds/d100
    State: Okay
    Size: 1228800 blocks (600 MB)
        Extent              Start Block              Block count
             0                   409664                   409600
             1                  1638528                   819200

nfsds/d100: Mirror
    Submirror 0: nfsds/d101
      State: Okay
    Submirror 1: nfsds/d102
      State: Resyncing
    Resync in progress: 0 % done
    Pass: 1
    Read option: roundrobin (default)
    Write option: parallel (default)
    Size: 71118513 blocks (33 GB)

nfsds/d101: Submirror of nfsds/d100
    State: Okay
    Size: 71118513 blocks (33 GB)
    Stripe 0:
        Device   Start Block  Dbase        State Reloc Hot Spare
        d9s0            0     No            Okay   No


nfsds/d102: Submirror of nfsds/d100
    State: Resyncing
    Size: 71118513 blocks (33 GB)
    Stripe 0:
        Device   Start Block  Dbase        State Reloc Hot Spare
        d17s0           0     No            Okay   No


nfsds/d10: Soft Partition
    Device: nfsds/d100
    State: Okay
    Size: 1228800 blocks (600 MB)
        Extent              Start Block              Block count
             0                       32                   409600
             1                   819296                   819200

Device Relocation Information:
Device   Reloc  Device ID
d17   No        -
d9    No        -
```

Using Hot Spare Pools

The hot spare pool feature allows you to put unused diskset partitions into a named pool. These partitions cannot then be used as building blocks to manually create volumes. Instead, the hot spare pool is associated with submirrors. The first partition in a pool that is big enough can replace a broken submirror with which it is associated.

Hot spares must be entire partitions. They *cannot* be soft partitions. Therefore, the model demonstrated in this module (using entire partitions as submirrors, building soft partitions on top of mirrors) works best with the hot spare model.

The following example adds a disk to the diskset, puts its partition s0 in a hot spare pool, and then associates the pool with a submirror. The association also could have been made when the submirror was built:

```
# metaset -s nfsds -a /dev/did/rdsk/d13
# metainit -s nfsds hsp001 /dev/did/rdsk/d13s0
nfsds/hsp001: Hotspare pool is setup

# metaparam -s nfsds -h hsp001 d101
# metastat -s nfsds d101
nfsds/d101: Concat/Stripe
    Hot spare pool: nfsds/hsp001
    Size: 71118513 blocks (33 GB)
    Stripe 0:
        Device   Start Block  Dbase        State Reloc Hot Spare
        d9s0            0     No            Okay   No


Device Relocation Information:
Device   Reloc  Device ID
d9   No         -
```

A good strategy, if you have extra drives in each array for hot sparing, is to do the following procedure (with different drives for each diskset):

1. Put extra drives from each array in diskset.
2. Put `s0` from each extra drive in two hot spare pools:
   a. In *pool one*, list drives from Array 1 first.
   b. In *pool two*, list drives from Array 2 first.
   c. Associate pool one with all submirrors from Array 1.
   d. Associate pool two with all submirrors from Array 2.

This results in sparing so that you are still mirrored across controllers, if possible, and sparing with both submirrors in the same controller as a last resort.

Managing Solaris Volume Manager Disksets and Sun Cluster Device Groups

When Solaris Volume Manager is run in the cluster environment, the commands are tightly integrated into the cluster environment. Creation of a shared diskset automatically registers that diskset as a cluster-managed device group:

```
# scstat -D

-- Device Group Servers --

                         Device Group        Primary Secondary
                         ------------        ------- ---------
  Device group servers:  nfsds               vincent theo


-- Device Group Status --

                              Device Group        Status
                              ------------        ------
  Device group status:        nfsds               Online


-- Multi-owner Device Groups --

                              Device Group        Online Status
                              ------------        -------------

# scconf -pv|grep nfsds
Device group name:                                 nfsds
  (nfsds) Device group type:                       SVM
  (nfsds) Device group failback enabled:           no
  (nfsds) Device group node list:                  vincent, theo
  (nfsds) Device group ordered node list:          no
  (nfsds) Device group desired number of secondaries: 1
  (nfsds) Device group diskset name:               nfsds
```

Addition or removal of a directly connected node to the diskset using `metaset -s nfsds -a -h` *`newnode`* automatically updates the cluster to add or remove the node from its list. There is no need to use cluster commands to resynchronize the device group if volumes are added and deleted.

In the Sun Cluster environment, use only the `scwitch` command (rather than the `metaset -[rt]` commands) to change physical ownership of the diskset. Note, as demonstrated in this example, if a mirror is in the middle of synching, this will force a switch anyway and restart the synch of the mirror all over again.

```
# scswitch -z -D nfsds -h theo
Dec 10 14:06:03 vincent Cluster.Framework: stderr: metaset: vincent:
Device busy
```

[a mirror was still synching, will restart on other node]

```
# scstat -D

-- Device Group Servers --

                         Device Group        Primary     Secondary
                         ------------        -------     ---------
  Device group servers:  nfsds               theo         vincent


-- Device Group Status --

                              Device Group        Status
                              ------------        ------
  Device group status:        nfsds               Online


-- Multi-owner Device Groups --

                              Device Group        Online Status
                              ------------        -------------
```

Managing Solaris Volume Manager Device Groups

While there is no need to register Solaris Volume Manager disksets with the `scconf` command (the registration is performed by the `metaset` command), the `scconf -c` command *can* be used to perform cluster-specific changes to Solaris Volume Manager device groups.

Device Groups Resynchronization

Solaris Volume Manager device groups are automatically resynchronized when new volumes are added to an existing diskset or volumes are deleted.

Other Changes to Device Groups

The properties of existing Solaris Volume Manager device groups *can* be changed. For example, the `failback` property of a group could be modified:

```
# scconf -c -D name=nfsds,failback=enabled
```

Maintenance Mode

You can take a Solaris Volume Manager device group "out of service," as far as the cluster is concerned, for emergency repairs.

To put the device group in maintenance mode, all of the Solaris Volume Manager volumes must be *unused* (unmounted, or otherwise not open). Then you can issue the following command:

```
    # scswitch -m -D nfsds
```

It is a rare event that will cause you to do this, as almost all repairs can still be done while the device group is in service.

To come back out of maintenance mode, just switch the device group onto a node:

```
    # scswitch -z -D nfsds -h new_primary_node
```

Using Global and Failover File Systems on Shared Diskset Devices

Sun Cluster 3.1 supports running data services on the following categories of file systems:

- *Global file systems* – These are accessible to all cluster nodes simultaneously, even those not physically connected to the storage.
- *Failover file systems* – These are mounted only on the node running the failover data service, which *must* be physically connected to the storage.

The file system type can be UFS or VxFS regardless of whether you are using global or failover file systems. The examples and the lab exercises in this modules assume you are using UFS.

Creating File Systems

The distinction between global and failover file system is *not* made at the time of file system creation. Use `newfs` as normal to create a UFS file system on the volume:

```
# newfs /dev/md/nfsds/rdsk/d10
```

Mounting File Systems

The distinction between global and failover file system is made in the `/etc/vfstab` "mount-at-boot" and "options" columns.

A global file system entry should look like the following, and it should be identical on all nodes that might run services which access the file system (including nodes not physically connected to the storage:)

```
/dev/md/nfsds/dsk/d10 /dev/md/nfsds/rdsk/d10 /global/nfs ufs 2 yes global,logging
```

A failover file system entry looks like the following, and it should be identical on all nodes that might run services which access the file system (they can only be nodes which are physically connected to the storage).

```
/dev/md/nfsds/dsk/d10 /dev/md/nfsds/rdsk/d10 /global/nfs ufs 2 no logging
```

Using Solaris Volume Manager to Mirror the Boot Disk

If Solaris Volume Manager is your volume manager of choice for the shared storage, you should—although this is not required—use Solaris Volume Manager to mirror the boot drive. The following example contains the following scenario:

- The boot drive is mirrored after cluster installation.
- All partitions on the boot drive are mirrored. The example has `root (/)` , swap, `/var`, and `/global/.devices/node@1`. That is, you will be manually creating four separate mirror devices.
- The geometry and partition tables of the boot disk and the new mirror are identical. Copying the partition table from one disk to the other was previously demonstrated in "Repartitioning a Mirror Boot Disk and Adding `metadb` Replicas" on page 7-18.
- Both the boot disk and mirror have three copies of the local `metadbs`. This was done previously.
- Soft partitions are *not* used on the boot disk. That way, if you need to back out, you can just go back to mounting the standard partitions by editing the `/etc/vfstab` manually.
- The `/global/.devices/node@1` is a special, *cluster-specific* case. For that one device you *must* use a different volume `d#` for the top level mirror on each node. The reason is that each of these file systems appear in the `/etc/mnttab` of each node (as global file systems) and the Solaris OS will not allow duplicate device names.

Verifying Partitioning and Local `metadbs`

The bold sections of the following output emphasize the partitions that you will mirror:

```
# df -k
Filesystem             kbytes    used   avail capacity  Mounted on
/dev/dsk/c0t0d0s0    26843000 1064410 25510160     5%    /
/proc                       0       0       0     0%    /proc
mnttab                      0       0       0     0%    /etc/mnttab
fd                          0       0       0     0%    /dev/fd
/dev/dsk/c0t0d0s3     4032504   43483 3948696     2%    /var
swap                  5471480     152 5471328     1%    /var/run
swap                  5471328       0 5471328     0%    /tmp
/dev/did/dsk/d1s5       95702    5317   80815 7% /global/.devices/node@1
/dev/did/dsk/d22s5      95702    5010   81122 6% /global/.devices/node@2
```

```
# swap -l
swapfile             dev  swaplo blocks   free
/dev/dsk/c0t0d0s1   32,1      16 8170064 8170064

# metadb
        flags           first blk       block count
    a m  p  luo        16              8192            /dev/dsk/c0t0d0s7
    a    p  luo        8208            8192            /dev/dsk/c0t0d0s7
    a    p  luo       16400            8192           /dev/dsk/c0t0d0s7
    a    p  luo        16              8192           /dev/dsk/c0t8d0s7
    a    p  luo        8208            8192           /dev/dsk/c0t8d0s7
    a    p  luo         16400         8192          /dev/dsk/c0t8d0s7
```

Building Volumes for Each Partition Except for Root

For all boot disk partitions except root, follow the same general strategy:

1. Create simple volumes for the existing partition and the other submirror. Use the `-f` option for the existing partition.
2. Create a mirror using *only* the existing partition. Do not attach the other half of the mirror until after a reboot. Make sure the `d#` chosen for the `/global/.devices/node@#` is different across your nodes.
3. Edit the `/etc/vfstab` file manually to use the new volume (mirror) instead of the original partition.
4. Wait to reboot until you do all partitions.

Building Volumes for Root Partition

For the root boot disk partition, follow the same general strategy:

1. Create simple volumes for the existing partition and the other submirror. Use the `-f` option for the existing partition.
2. Create a mirror using *only* the existing partition. Do not attach the other half of the mirror until after a reboot.
3. Use `metaroot` to edit the `vfstab` and `system` files.

Running the Commands

Examples are shown for `root` and `swap`. The procedure for the other partitions are just like the one for `swap`.

```
# metainit -f d11 1 1 c0t0d0s0
d11: Concat/Stripe is setup
# metainit d12 1 1 c0t8d0s0
d12: Concat/Stripe is setup
# metainit d10 -m d11
d10: Mirror is setup
# metaroot d10

# metainit -f d21 1 1 c0t0d0s1
d11: Concat/Stripe is setup
# metainit d22 1 1 c0t8d0s1
d12: Concat/Stripe is setup
# metainit d20 -m d21
d10: Mirror is setup
# vi /etc/vfstab
(change correct line manually)

/dev/md/dsk/d20 -       -       swap    -       no      -
```

Rebooting and Attaching the Second Submirror

After a reboot, you can attach the second submirror to each volume. The synchronizing of the mirrors runs in the background, and can take a long time.

```
# init 6
.
.
# df -k
/dev/md/dsk/d10      26843000 1064163 25510407     5%    /
/proc                      0       0       0     0%    /proc
mnttab                     0       0       0     0%    /etc/mnttab
fd                         0       0       0     0%    /dev/fd
/dev/md/dsk/d30      4032504   43506 3948673     2%    /var
swap                 5550648     152 5550496     1%    /var/run
swap                 5550496       0 5550496     0%    /tmp
/dev/md/dsk/d50        95702    5317   80815     7%
/global/.devices/node@1
/dev/did/dsk/d22s5     95702    5010   81122     6%
/global/.devices/node@2

# swap -l
swapfile             dev  swaplo blocks   free
/dev/md/dsk/d20     85,20     16 8170064 8170064

# metattach d10 d12
# metattach d20 d22
# metattach d30 d32
# metattach d50 d52
```

Exercise: Configuring Solaris Volume Manager

In this exercise, you complete the following tasks:

- Task 1 – Initializing the Solaris Volume Manager Local `metadbs`
- Task 2 – Selecting the Solaris Volume Manager Demo Volume Disk Drives
- Task 3 – Configuring Solaris Volume Manager Disksets
- Task 4 – Configuring Solaris Volume Manager Demonstration Volumes
- Task 5 – Creating a Global `nfs` File System
- Task 6 – Creating a Global `web` File System
- Task 7 – Testing Global File Systems
- Task 8 – Managing Disk Device Groups
- Task 9 – Viewing and Managing Solaris Volume Manager Device Groups
- Task 10 (Optional) – Mirroring the Boot Disk With Solaris Volume Manager

Preparation

This exercise assumes you are running the Sun Cluster 3.1 software environment on Solaris 10 OS.

At least one local disk on each node must have a small unused slice that you can use for the local `metadbs`. The examples in this exercise use slice 7.

If you have encapsulated (but not mirrored) your root disk in a previous VERITAS Volume Manager lab, you can still do this lab assuming that you have a second local disk. In order to run Solaris Volume Manager, you *must* have at least one local disk that has room for a local `metdb` and is *not* under control of VERITAS Volume Manager.

During this exercise, you create two data service disksets that each contain a single mirrored volume as shown in Figure 7-7.

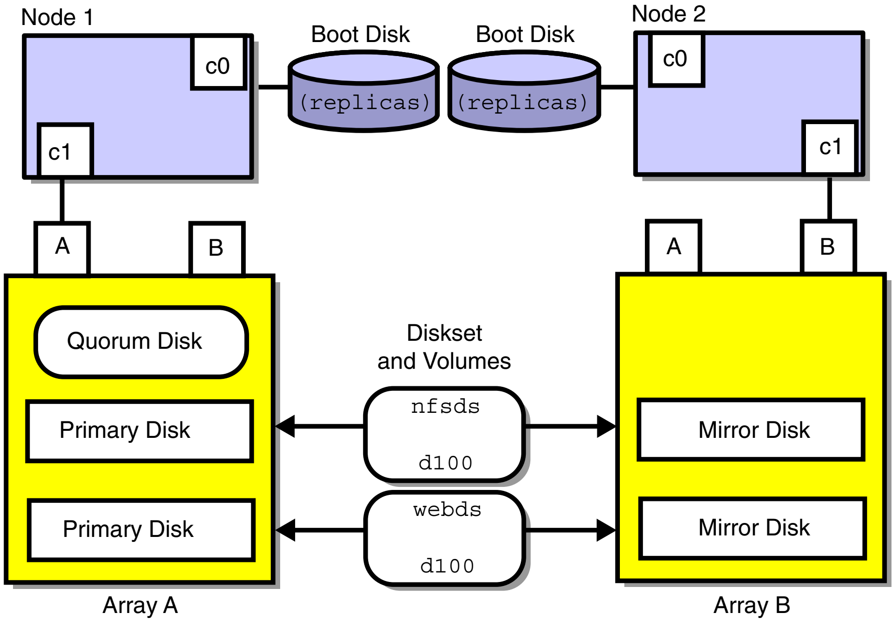


Figure 7-7 Configuring Solaris Volume Manager

Note – During this exercise, when you see italicized names, such as ***`IPaddress`***, ***`enclosure_name`***, ***`node1`***, or ***`clustername`*** embedded in a command string, substitute the names appropriate for your cluster.

Task 1 – Initializing the Solaris Volume Manager Local `metadbs`

Before you can use Solaris Volume Manager to create disksets and volumes, you must initialize the state database and create one or more replicas.

Configure the system boot disk (or other local disk) on each cluster host with a small unused partition. This should be slice 7.

Perform the following steps:

1. On each node in the cluster, verify that the local disk has a small unused slice available for use. Use the `format` command to verify the physical path to the unused slice. Record the paths of the unused slice on each cluster host. A typical path is `c0t0d0s7`.

   Node 1 Replica Slice:_____________

   Node 2 Replica Slice:________________

Caution – You must ensure that you are using the correct slice. A mistake can corrupt the system boot disk. If in doubt, check with your instructor.

2. On each node in the cluster, use the `metadb` command to create three replicas on the unused boot disk slice.

```
# metadb -a -c 3 -f replica_slice
```

Caution – Make sure you reference the correct slice address on each node. You can destroy your boot disk if you make a mistake.

3. On all nodes, verify that the replicas are configured and operational.

```
# metadb
flags    first blk   block count
 a  u      16          8192      /dev/dsk/c0t0d0s7
 a  u      8208        8192      /dev/dsk/c0t0d0s7
 a  u      16400       8192      /dev/dsk/c0t0d0s7
```

Task 2 – Selecting the Solaris Volume Manager Demo Volume Disk Drives

Perform the following steps to select the Solaris Volume Manager demo volume disk drives:

1. On Node 1, type the **`scdidadm -L`** command to list all of the available DID drives.
2. Record the logical path and DID path numbers of four disks that you will use to create the demonstration disksets and volumes in Table 7-2. Remember to mirror across arrays.

Note – You need to record only the last portion of the DID path. The first part is the same for all DID devices: `/dev/did/rdsk`.

Table 7-2 Logical Path and DID Numbers

Diskset	Volumes	Primary Disk	Mirror Disk
example	*d100*	*c2t3d0 d4*	*c3t18d0 d15*
nfsds	d100		
webds	d100		

Note – Make sure the disks you select are *not* local devices. They must be dual-hosted and available to more than one cluster host.

Task 3 – Configuring Solaris Volume Manager Disksets

Perform the following steps to create demonstration disksets and volumes for use in later exercises:

1. On Node 1, create the nfsds diskset, and configure the nodes that are physically connected to it.

```
# metaset -s nfsds -a -h node1 node2
```

2. On Node 1, create the webds diskset and configure the nodes that are physically connected to it.

```
# metaset -s webds -a -h node1 node2
```

3. Add the diskset mediators to each diskset, the same nodes.

```
# metaset -s nfsds -a -m node1 node2
# metaset -s webds -a -m node1 node2
```

4. Add the primary and mirror disks to the nfsds diskset.

```
# metaset -s nfsds -a /dev/did/rdsk/primary \
/dev/did/rdsk/mirror
```

5. Add the primary and mirror disks to the webds diskset.

```
# metaset -s webds -a /dev/did/rdsk/primary \
/dev/did/rdsk/mirror
```

6. Verify the status of the new disksets.

```
# metaset -s nfsds
# metaset -s webds
# medstat -s nfsds
# medstat -s webds

# scstat -D
```

Task 4 – Configuring Solaris Volume Manager Demonstration Volumes

Perform the following steps on Node 1 to create a 500-Mbyte mirrored volume in each diskset, as shown for the two following disksets.

The `nfsds` Diskset

Follow these steps to create the volume on the `nfsds` diskset:

1. Create a submirror on each of your disks in the `nfsds` diskset.

   ```
   # metainit -s nfsds d0 1 1 /dev/did/rdsk/primarys0
   # metainit -s nfsds d1 1 1 /dev/did/rdsk/mirrors0
   ```

2. Create a mirror volume, `d99`, using the `d0` submirror.

   ```
   # metainit -s nfsds d99 -m d0
   ```

3. Attach the second submirror, `d1`, to the volume `d99`.

   ```
   # metattach -s nfsds d99 d1
   ```

4. Create a 500 Mbyte soft partition, `d100`, on top of your mirror. This is the volume you will actually use for your file system data.

   ```
   # metainit -s nfsds d100 -p d99 500m
   ```

5. Verify the status of the new volume.

   ```
   # metastat -s nfsds
   ```

The `webds` Diskset

Follow these steps to create the volume on the `webds` diskset:

1. Create a submirror on each of your disks in the `webds` diskset.

   ```
   # metainit -s webds d0 1 1 /dev/did/rdsk/primarys0
   # metainit -s webds d1 1 1 /dev/did/rdsk/mirrors0
   ```

2. Create a mirror volume, `d99`, using the `d0` submirror.

   ```
   # metainit -s webds d99 -m d0
   ```

3. Attach the second submirror, `d1`, to the volume `d99`.

   ```
   # metattach -s webds d99 d1
   ```

4. Create a 500-Mbyte soft partition on top of your mirror. This is the volume you will actually use for your file system data.

   ```
   # metainit -s webds d100 -p d99 500m
   ```

5. Verify the status of the new volume.

   ```
   # metastat -s webds
   ```

Task 5 – Creating a Global nfs File System

Perform the following steps on Node 1 to create a global file system in the nfsds diskset:

1. On Node 1, create a file system on d100 in the nfsds diskset.

   ```
   # newfs /dev/md/nfsds/rdsk/d100
   ```

2. On *all nodes*, create a global mount point for the new file system.

   ```
   # mkdir /global/nfs
   ```

3. On *all nodes*, add a mount entry in the /etc/vfstab file for the new file system with the global and logging mount options.

   ```
   /dev/md/nfsds/dsk/d100 /dev/md/nfsds/rdsk/d100 \
   /global/nfs ufs 2 yes global,logging
   ```

Note – Do not use the line continuation character (\) in the vfstab file.

4. On Node 1, mount the /global/nfs file system.

   ```
   # mount /global/nfs
   ```

5. Verify that the file system is mounted and available on *all* nodes.

   ```
   # mount
   # ls /global/nfs
   lost+found
   ```

Task 6 – Creating a Global web File System

Perform the following steps on Node 1 to create a global file system in the webds diskset:

1. On Node 1, create a file system on d100 in the webds diskset.

   ```
   # newfs /dev/md/webds/rdsk/d100
   ```

2. On *all nodes*, create a global mount point for the new file system.

   ```
   # mkdir /global/web
   ```

3. On *all nodes*, add a mount entry in the `/etc/vfstab` file for the new file system with the `global` and `logging` mount options.

   ```
   /dev/md/webds/dsk/d100 /dev/md/webds/rdsk/d100 \
   /global/web ufs 2 yes global,logging
   ```

Note – Do not use the line continuation character (\) in the `vfstab` file.

4. On Node 1, mount the `/global/web` file system.

   ```
   # mount /global/web
   ```

5. Verify that the file system is mounted and available on *all* nodes.

   ```
   # mount
   # ls /global/web
   lost+found
   ```

Task 7 – Testing Global File Systems

Perform the following steps to confirm the general behavior of globally available file systems in the Sun Cluster 3.1 software environment:

1. On Node 2, move into the `/global/nfs` file system.

   ```
   # cd /global/nfs
   ```

2. On Node 1, try to unmount the `/global/nfs` file system. You should get an error that the file system is busy.
3. On Node 2, move out of the `/global/nfs` file system (`cd /`) and try to unmount it again on Node 1.
4. Mount the `/global/nfs` file system on Node 1.
5. Try unmounting and mounting `/global/nfs` from all nodes.

Task 8 – Managing Disk Device Groups

Perform the following steps to migrate a disk device group (diskset) between cluster nodes:

1. Make sure the *device groups* are online (to the Sun Cluster software).

   ```
   # scstat -D
   ```

Note – You can bring a device group online to a selected node as follows:

```
        # scswitch -z -D nfsds -h node1
```

---

2. Verify the current demonstration device group configuration.

   ```
   # scconf -p |grep group
   ```

3. Shut down Node 1.

   The `nfsds` and `webds` disksets should automatically migrate to Node 2 (verify with the `scstat -D` command).

4. Boot Node 1. Both disksets should remain mastered by Node 2.

5. Use the `scswitch` command from either node to migrate the `nfsds` diskset to Node 1.

   ```
   # scswitch -z -D nfsds -h node1
   ```

Task 9 – Viewing and Managing Solaris Volume Manager Device Groups

In this task you view and manage Solaris Volume Manager device groups through SunPlex manager.

Perform the following steps on your administration workstation:

1. In a web browser, log in to Sun Java Web Console on any cluster node:

   https://*nodename*:6789

2. Enter the SunPlex Manager application.

3. Click on the `Shared Storage` folder on the left.

4. Investigate the status information and graphical topology information that you can see regarding Solaris Volume Manager device groups.

5. View the details of a Solaris Volume Manager device group by clicking on its name.
6. Use the Switch Primaries button to switch the primary for one of your device groups.

Task 10 (Optional) – Mirroring the Boot Disk With Solaris Volume Manager

You can try this procedure if you have time. Normally, you would run it on all nodes but feel free to run it on one node only as a "proof of concept."

This task assumes you have a second drive with identical geometry to the first. If you do not, you can still perform the parts of the lab that do not refer to the second disk. That is, create mirrors with only one sub-mirror for each partition on your boot disk.

The example uses `c0t8d0` as the second drive.

Repartitioning the Second Drive and Adding `metadb` Replicas

Use the following commands to repartition the second drive and add `metadbs`:

```
# prtvtoc /dev/rdsk/c0t0d0s0|fmthard -s -
/dev/rdsk/c0t8d0s0
fmthard:  New volume table of contents now in place.

# metadb -a -c 3 c0t8d0s7
```

Using the `metadb` Command to Verify `metadb` Replicas

Use the following command to verify the `metadbs`:

```
# metadb
        flags           first blk       block count
     a        u         16              8192            /dev/dsk/c0t0d0s7
     a        u         8208            8192            /dev/dsk/c0t0d0s7
     a        u         16400           8192            /dev/dsk/c0t0d0s7
     a        u         16              8192            /dev/dsk/c0t8d0s7
     a        u         8208            8192            /dev/dsk/c0t8d0s7
     a        u         16400           8192            /dev/dsk/c0t8d0s7
```

Making Mirrors for Non-Root Slices

For *each* non-root slice on your drive (including `swap` but not including the metadb partition), make submirrors from the partition on each of your two disks. Do *not* attach the second submirror. Edit the `vfstab` file manually to reference the mirror volume instead of the partition or DID device. Do not reboot yet.

The example shows the procedure for the `swap` partition. If you are doing this on two nodes, make sure the volume number chosen for the `/global/.devices/node@#` mirror is different on each node.

```
# metainit -f d21 1 1 c0t0d0s1
d11: Concat/Stripe is setup
# metainit d22 1 1 c0t8d0s1
d12: Concat/Stripe is setup
# metainit d20 -m d21
d10: Mirror is setup
# vi /etc/vfstab
(change correct line manually)

/dev/md/dsk/d20 -       -       swap    -       no      -
```

Making the Mirror for the Root Slice

You do not have to edit the `vfstab` or `system` files manually for the root volume; the `metaroot` command takes care of it for you.

```
# metainit -f d11 1 1 c0t0d0s0
d11: Concat/Stripe is setup
# metainit d12 1 1 c0t8d0s0
d12: Concat/Stripe is setup
# metainit d10 -m d11
d10: Mirror is setup
# metaroot d10
```

Rebooting and Attaching the Second Submirror

Use the following commands to reboot and attach the second submirror:

```
# init 6
# df -k
# swap -l
# metattach d10 d12
(etc)
# metattach d50 d52
```

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 8

Managing the Public Network With IPMP

Objectives

Upon completion of this module, you should be able to:

- Define the purpose of IPMP
- Define the concepts for an IPMP group
- List examples of network adapters in IPMP groups on a single Solaris OS server
- Describe the operation of the `in.mpathd` daemon
- List the new options to the `ifconfig` command that support IPMP, and configure IPMP with `/etc/hostname.xxx` files
- Perform a "forced failover" of an adapter in an IPMP group
- Configure IPMP manually with IPMP commands
- Describe the integration of IPMP into the Sun Cluster software environment

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- Should you configure IPMP before or after you install the Sun Cluster software?
- Why is IPMP required even if you do not have redundant network interfaces?
- Is the configuration of IPMP any different in the Sun Cluster software environment than on a standalone Solaris OS?
- Is the behavior of IPMP any different in the Sun Cluster software environment?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Microsystems, Inc. *Solaris™ Administration Guide: IP Services (Solaris 10)*, Part Number 816-4554

Introducing IPMP

IPMP has been a standard part of the base Solaris OS since Solaris 8 OS Update 3 (01/01).

IPMP enables you to configure redundant network adapters, on the same server (node) and on the same subnet, as an IPMP *failover group.*

IPMP detects failures and repairs of network connectivity for adapters, and provides failover and failback of IP addresses among members of the same group. Existing TCP connections to the IP addresses that fail over as part of an IPMP group are interrupted for short amounts of time, and are *not* disconnected.

In the Sun Cluster 3.1 software environment, it is *required* to use IPMP to manage any public network interfaces on which you will be placing the IP addresses associated with applications running in the cluster.

These application IP addresses are implemented by mechanisms known as `LogicalHostname` and `SharedAddress` resources. The validation methods associated with these resources will not allow their configuration unless the public network adapters being used are already under control of IPMP. Module 9, "Introducing Data Services, Resource Groups, and HA-NFS," and Module 10, "Configuring Scalable Services and Advanced Resource Group Relationships," provide detail about the proper configuration of applications and their IP addresses in the Sun Cluster software environment.

Describing General IPMP Concepts

IPMP allows you to group network adapters for redundancy. Members of the same IPMP group are identified by a common group name. This can be any alphanumeric name. The group name is only meaningful inside a single Solaris OS server.

Defining IPMP Group Requirements

You must observe the following rules when configuring an IPMP group:

- A network adapter can be a member of only one group.
- When both IPv4 and IPv6 are configured on a physical adapter, the group names are *always* the same (and thus need not be specified explicitly for IPv6).
- All members of a group must be on the same subnet. The members, if possible, should be connected to physically separate switches on the same subnet.
- Adapters on different subnets *must* be in different groups.
- It is possible to have multiple groups on the same subnet.
- You *can* have a group with only one member. However, there cannot be any failover associated with this adapter.
- Each Ethernet network interface must have a unique MAC address. This is achieved by setting the `local-mac-address?` variable to `true` in the OpenBoot PROM.
- Network adapters in the same group must be the same type. For example, you cannot combine Ethernet interfaces and Asynchronous Transfer Mode (ATM) interfaces in the same group.
- In Solaris 8 OS and Solaris 9 OS, when more than one adapter is in an IPMP group, each adapter requires a dedicated *test interface*. This is an extra static IP for each group member specifically configured for the purposes of testing the health of the adapter using `ping` traffic.

  The test interface enables test traffic on *all* the members of the IPMP group. This is the reason that `local-mac-address?=true` is required for IPMP.

- Solaris 10 OS does not require test addresses, even with multiple adapters in a group. If you set up adapters in Solaris 10 IPMP groups without test addresses, the health of the adapter is determined solely by the link state of the adapter.

Note – Using IPMP without test addresses reduces network traffic and reduces the administrative strain of allocating the addresses. However, the testing is less robust. For example, adapters with a valid link state may have broken receive logic. Such an adapter would be properly faulted using test addresses. The remaining examples in this module and its exercises focus on creating IPMP configurations with test addresses.

- If both IPv4 and IPv6 are configured on the same adapters, it is possible to have test addresses for both IPv4 and IPv6, but you do not have to. If a failure is detected (even if you have only an IPv4 test address), all IPv4 and IPv6 addresses (except the test address) will fail over to the other physical adapter.

  If you *do* choose to have an IPv6 test address, that test address will always be the link-local IPv6 address (the address automatically assigned to the adapter, that can only be used on the local subnet). See “Configuring Adapters for IPV6 (Optional)” on page 8-16 to see how to set up IPv6 on your adapters.

Configuring Standby Interfaces in a Group

In an IPMP group with two or more members, it is optional to configure an adapter as a *standby* adapter for the group. Standby adapters have the following properties:

- You are allowed (and must) configure *only* the test interface on the adapter. Any attempt to manually configure any other addresses will fail.
- Additional IP addresses will be added to the adapter *only* as a result of failure of another member of the group.
- The standby interface will be preferred as a failover target if another member of the group fails.
- You must have at least one member of the group which is not a standby adapter.

Note – The examples of two-member IPMP groups in the Sun Cluster software environment will *not* use any standby adapters. This will allow the Sun Cluster software to place additional IP addresses associated with applications on both members of the group simultaneously.

If you had a three-member IPMP group in the Sun Cluster software environment, setting up one of the members as a standby is still a valid option.

Examining IPMP Group Examples

The following sections describe and illustrate examples of network adapters in IPMP groups on a single Solaris OS server.

Single IPMP Group With Two Members and No Standby

Figure 8-1 shows a server with two member adapters in a single IPMP group. These two adapters must be on the same subnet and provide failover for each other.

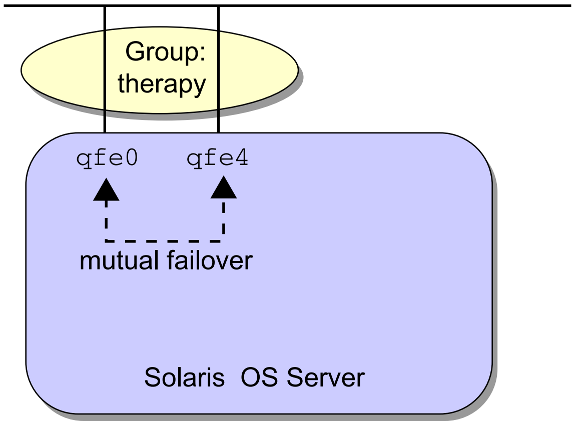


Figure 8-1 Server With Two Member Adapters in a Single IPMP Group

Single IPMP Group With Three Members Including a Standby

Figure 8-2 shows a server with three members of a single IPMP group. One of the adapters in the group is configured as a standby interface.

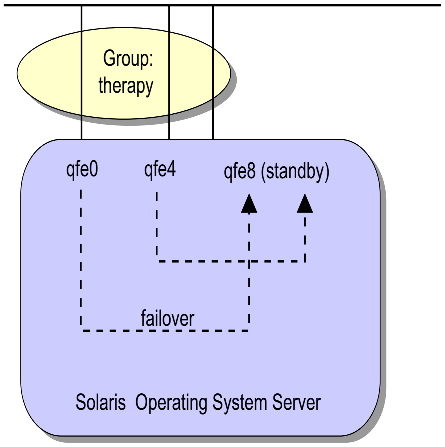


Figure 8-2 Server With Three Members of a Single IPMP Group

Two IPMP Groups on Different Subnets

Figure 8-3 shows how different IPMP groups must be used for adapters on different subnets.

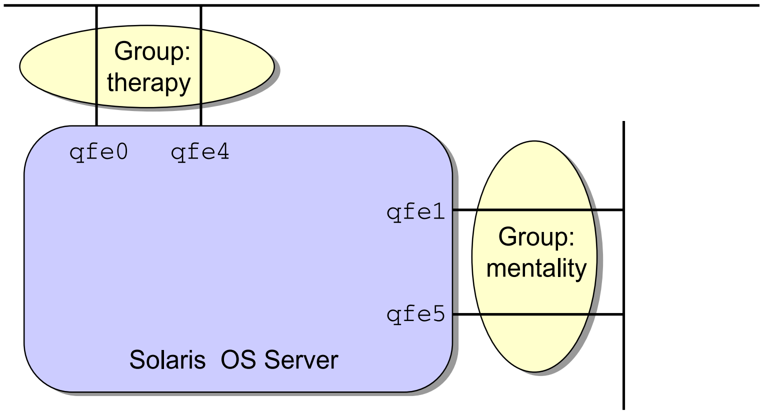


Figure 8-3 Two IPMP Groups on Different Subnets

Two IPMP Groups on the Same Subnet

Figure 8-4 shows two different IPMP groups configured on the same subnet. Failover will still occur only within each particular group.

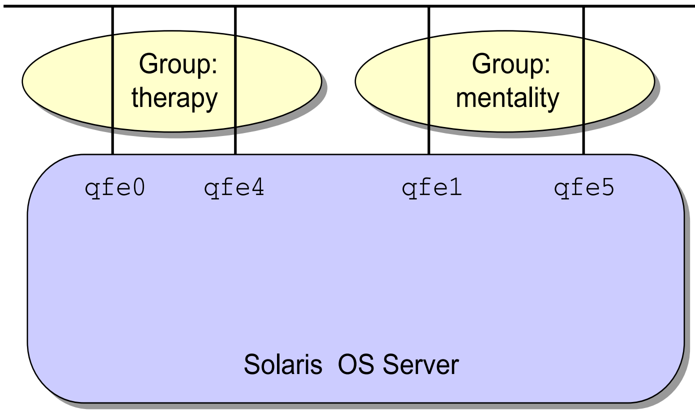


Figure 8-4 Two IPMP Groups on the Same Subnet

Describing IPMP

The in.mpathd daemon controls the behavior of IPMP. This behavior can be summarized as a three-part scheme:

- Network path failure detection
- Network path failover
- Network path failback

The in.mpathd daemon starts automatically when an adapter is made a member of an IPMP group through the ifconfig command.

Network Path Failure Detection

The following paragraphs describe the functionality of the in.mpathd daemon in a configuration using test addresses. In Solaris 10, without test addresses, adapter failure detection and repair is based solely on the link state of the adapter.

When test addresses are used, the in.mpathd daemon sends Internet Control Message Protocol (ICMP) echo probes (pings) to the targets connected to the link on all adapters that belong to a group, in order to detect failures and repair. The test address is used as the source address of these pings.

Because the in.mpathd daemon determines what targets to probe dynamically, you cannot configure the targets. Targets will be chosen in the following order:

1. Default router (if you have a single default router it will be the only target)
2. Other targets discovered by a ping command to the 224.0.0.1 (all-hosts) multicast IP address

To ensure that each adapter in the group functions properly, the in.mpathd daemon probes all the targets separately through all the adapters in the multipathing group, using each adapter’s test address. If there are no replies to five consecutive probes, the in.mpathd daemon considers the adapter as having failed. The probing rate depends on the failure detection time (FDT). The default value for failure detection time is 10 seconds. For a failure detection time of 10 seconds, the probing rate is approximately one probe every two seconds.

Network Path Failover

After a failure is detected, failover of all network access occurs from the failed adapter to another functional adapter in the group. If you have configured a standby adapter, the `in.mpathd` daemon chooses that for failover of IP addresses and multicast memberships. If you have not configured a standby adapter, `in.mpathd` chooses the adapter with the least number of IP addresses. The new adapter assumes all of the IP addresses of the failed adapter, except for the test address.

Network Path Failback

The `in.mpathd` daemon detects (using continued `ping` traffic on the test address of the failed adapter) if the failed path has been repaired. At this time the IP addresses moved in the failover will be returned to their original path, assuming failback is enabled.

Configuring IPMP

This section outlines how IPMP is configured directly through options to the `ifconfig` command.

Most of the time, you will just specify the correct IPMP-specific `ifconfig` options in the `/etc/hostname.`*`xxx`* files. After these are correct, you never have to change them again.

Examining New `ifconfig` Options for IPMP

Several new `ifconfig` options have been added for use with IPMP. They are the following:

- `group` *`groupname`* – Adapters on the same subnet are placed in a failover group by configuring them with the same groupname. The groupname can be any name and must be unique only within a single Solaris OS (node). It is irrelevant if you have the same or different groupnames across the different nodes of a cluster.
- `-failover` – Use this option to demarcate the test interface for each adapter. The test interface is the only interface on an adapter which will not fail over to another adapter when a failure occurs.
- `deprecated` – This option, while not required, is generally always used on the test interfaces. Any IP address marked with this option will not be used as the source IP address for any client connections initiated on this machine.
- `standby` – This option, when used with the physical adapter (`-failover deprecated standby`), turns that adapter into a standby-only adapter. No other virtual interfaces can be configured on that adapter until a failure occurs on another adapter on the group.
- `addif` – Use this option to create the next available virtual interface for the specified adapter. The maximum number of virtual interfaces per physical interface is 8192.

  The Sun Cluster 3.1 software automatically uses the `ifconfig` *`xxx`* `addif` command to add application-specific IP addresses as virtual interfaces on top of the physical adapters that you have placed in IPMP groups. Details about configuring these IP addresses are in Module 9, "Introducing Data Services, Resource Groups, and HA-NFS," and Module 10, "Configuring Scalable Services and Advanced Resource Group Relationships."

Putting Test Interfaces on Physical or Virtual Interfaces

IPMP was designed so that the required test interface for each member of an IPMP group can be either on the physical interface (for example, `qfe0`) or on a virtual interface (for example, `qfe0:1`, created with the `ifconfig` *xxx* `addif` command).

There were at one time many existing IPMP documents written with the test IP on the physical interface because it looks "cleanest" to have only the virtual interfaces failing over to other adapters and the IP on the physical interface always staying where it is.

The convention changed due to a bug relating to having more than one non-deprecated IP address on the same interface. While a test address normally is deprecated, it was still safer to guarantee that if there were a non-deprecated address (the physical address), that it be on the physical interface.

The examples in this module follow the convention of placing the test interface on a virtual interface, when given the choice.

Using `ifconfig` Commands to Configure IPMP Examples

While most of the time you will set up your `/etc/hostname.`*xxx* files once and never have to worry about IPMP configuration again, these examples show using the `ifconfig` command directly with the IPMP-specific options.

The `/etc/hosts` fragment shows the dedicated IP address for the test interface for each adapter. While it is not required to have host names for these, remember that these IPs are reserved.

```
# cat /etc/hosts
.
 #physical host address
 172.20.4.192    vincent

 # test addresses for vincent (node 1)
 172.20.4.194    vincent-qfe1-test
 172.20.4.195    vincent-qfe2-test

# ifconfig qfe1 vincent group therapy netmask + broadcast + up
# ifconfig qfe1 addif vincent-qfe1-test -failover deprecated \
netmask + broadcast + up

# ifconfig qfe2 plumb
# ifconfig qfe2 vincent-qfe2-test group therapy -failover deprecated \
netmask + broadcast + up

# ifconfig -a
.
qfe1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
         inet 172.20.4.192 netmask ffffff00 broadcast 172.20.4.255
         groupname therapy
         ether 8:0:20:f1:2b:d
qfe1:1:flags=9040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAI
          LOVER> mtu 1500 index 2
         inet 172.20.4.194 netmask ffffff00 broadcast 172.20.4.255
qfe2:flags=9040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILO
          VER> mtu 1500 index 3
         inet 172.20.4.195 netmask ffffff00 broadcast 172.20.4.255
         groupname therapy
         ether 8:0:20:f1:2b:e
```

Configuring the /etc/hostname.*xxx* Files for IPMP

Usually, you will not need to manually use ifconfig commands. Instead, the ifconfig commands get run automatically at Solaris OS boot time by way of their presence in the /etc/hostname.*xxx* files.

The new format of these files (since Solaris 8 OS) allows you to put all the options to the ifconfig command in these files.

You can omit the netmask + broadcast + from these files as this is always automatically applied by the Solaris OS boot process as a group for all of the interfaces that have been configured.

```
# cat /etc/hostname.qfe1
 vincent group therapy up
 addif vincent-qfe1-test -failover deprecated up


# cat /etc/hostname.qfe2
 vincent-qfe2-test group therapy -failover deprecated up
```

Configuring Adapters for IPV6 (Optional)

If you want to support application-specific IP addresses that use IPv6, you must manually configure your adapters for IPv6. If you do not want to use any IPv6 on your public network at all, you do not need to configure IPv6 on them at all.

You can choose to configure IPV6 with or without an IPV6 test address. Since you already have an IPV4 test address it is not absolutely required to also have an IPV6 test address. If you do not want an IPV6 test address you can just create empty filenames for the IPV6 interfaces, and if you reboot, IPV6 will be enabled on the interfaces with the same group name `therapy` as the IPV4 interfaces, without an IPV6 test address.

```
# touch /etc/hostname6.qfe1
# touch /etc/hostname6.qfe2
# init 6
```

Alternatively, you can choose to have an IPV6 test address. Since the only address that can serve this purpose is the link-local address, which is automatically assigned to the interface and can be used only on the local subnet, you can still have very simple IPV6 adapter configuration files:

```
# vi /etc/hostname6.qfe1
-failover up
# vi /etc/hostname6.qfe2
-failover up
# init 6
```

Multipathing Configuration File

The in.mpathd daemon uses the settings in the /etc/default/mpathd configuration file to invoke multipathing. Changes to this file are read by the in.mpathd daemon at startup and on a SIGHUP. This file contains the following default settings and information:

```
#
# Time taken by mpathd to detect a NIC failure in ms.
# The minimum time that can be specified is 100 ms.
#
FAILURE_DETECTION_TIME=10000
#
# Failback is enabled by default. To disable failback
# turn off this option
#
FAILBACK=yes
#
# By default only interfaces configured as part of
# multipathing groups are tracked. Turn off this
# option to track all network interfaces on the system
#
TRACK_INTERFACES_ONLY_WITH_GROUPS=yes
```

Note – Generally, you do not need to edit the default /etc/default/mpathd configuration file.

The three settings you can alter in this file are:

- FAILURE_DETECTION_TIME – You can lower the value of this parameter. If the load on the network is too great, the system cannot meet the failure detection time value. Then the in.mpathd daemon prints a message on the console, indicating that the time cannot be met. It also prints the time that it can meet currently. If the response comes back correctly, the in.mpathd daemon meets the failure detection time provided in this file.
- FAILBACK – After a failover, failbacks take place when the failed adapter is repaired. However, the in.mpathd daemon does not fail back the addresses if the FAILBACK option is set to no.
- TRACK_INTERFACES_ONLY_WITH_GROUPS – In standalone servers, you can set this to no so that in.mpathd monitors traffic even on adapters not in groups. In the cluster environment make sure you leave the value yes so that private transport adapters are not probed.

Performing Failover and Failback Manually

You can do a “forced failover” of an adapter in an IPMP group. The following is an example:

```
# if_mpadm -d qfe1
```

This command causes IPMP to behave as if qfe1 had failed. All IP addresses, except the test interface, are failed over to another member of the group. The interface is marked down and will not be accessed by in.mpathd until re-enabled.

To re-enable an adapter after this operation, use:

```
# if_mpadm -r qfe1
```

This command allows the adapter to take back the IP addresses to which it was originally assigned, assuming FAILBACK=yes is set in the /etc/default/mpathd file.

Configuring IPMP in the Sun Cluster 3.1 Software Environment

There are no special cluster-related tools for configuring IPMP in the Sun Cluster 3.1 software environment. IPMP is configured on each node exactly as in a non-clustered environment.

Configuring IPMP Before or After Cluster Installation

If you have `/etc/hostname.`*xxx* files without IPMP groups, `scinstall` will automatically rewrite the files so that each such adapter is a member of an IPMP singleton group, as in the following example:

```
vincent:/# cat /etc/hostname.qfe1
vincent netmask + broadcast + group sc_ipmp0 up
```

Since you always need to configure a multiadapter IPMP group by hand, you can choose to wait until after `scinstall`, as you are doing in this module (where you will be replacing some files rewritten by `scinstall)`, or you can configure IPMP before `scinstall.`

You might choose to make IPMP configuration part of your Solaris OS JumpStart software installation, by copying in the correct `/etc/hostname.`*xxx* files as part of a JumpStart software finish script.

Using Same Group Names on Different Nodes

It makes no difference to IPMP whether you use the same group names or different group names for IPMP across nodes. IPMP itself is aware only of what is going on in the local Solaris OS.

It is a helpful convention to use the same group names to indicate groups of adapters on different nodes that are connected to the same subnet.

Understanding Standby and Failback

It is unlikely that you want to use a standby interface in a two-member IPMP group in the cluster environment. If you do, the Sun Cluster software will be unable to load-balance additional application-related IP addresses across the members of the group.

Keep `FAILBACK=yes` in the `/etc/default/mpathd` file. Because the Sun Cluster software is automatically trying to load-balance additional IP addresses across the members of the group, it makes sense that you want them to “rebalance” when a repair is detected.

Integrating IPMP Into the Sun Cluster 3.1 Software Environment

There is nothing special about IPMP itself within the cluster. In the cluster environment, as in a non-cluster environment, `in.mpathd` is concerned with probing network adapters only on a single Solaris OS, and manages failovers and failbacks between the adapters in a group.

The Sun Cluster software environment, however, needs additional capability *wrapped around* IPMP to do the following:

- Store cluster-wide status of IPMP groups on each node in the CCR, and enable retrieval of this status with the `scstat` command.
- Facilitate application failover in the case where all members of an IPMP group on one node have failed, but the corresponding group on the same subnet on another node has a healthy adapter.

Clearly, IPMP itself is unaware of any of these cluster requirements. Instead, the Sun Cluster 3.1 software uses a cluster-specific public network management daemon (`pnmd`) to perform this cluster integration.

Capabilities of the `pnmd` Daemon in Sun Cluster 3.1 Software

In the cluster environment, the `pnmd` daemon has the following capabilities:

- Populate CCR with public network adapter status
- Facilitate application failover

When `pnmd` detects that all members of a local IPMP group have failed, it consults a file named `/var/cluster/run/pnm_callbacks`. This file contains entries that would have been created by the activation of `LogicalHostname` and `SharedAddress` resources. (There is more information about this in Module 9, “Introducing Data Services, Resource Groups, and HA-NFS,” and Module 10, “Configuring Scalable Services and Advanced Resource Group Relationships.”

It is the job of the `hafoip_ipmp_callback`, in the following example, to decide whether to migrate resources to another node.

```
# cat /var/cluster/run/pnm_callbacks
therapy orangecat-nfs.mon  \
/usr/cluster/lib/rgm/rt/hafoip/hafoip_ipmp_callback mon nfs-rg orangecat-
nfs
```

Summary of IPMP Cluster Integration

Figure 8-5 summarizes the IPMP and `pnmd` elements of public network management in the cluster.

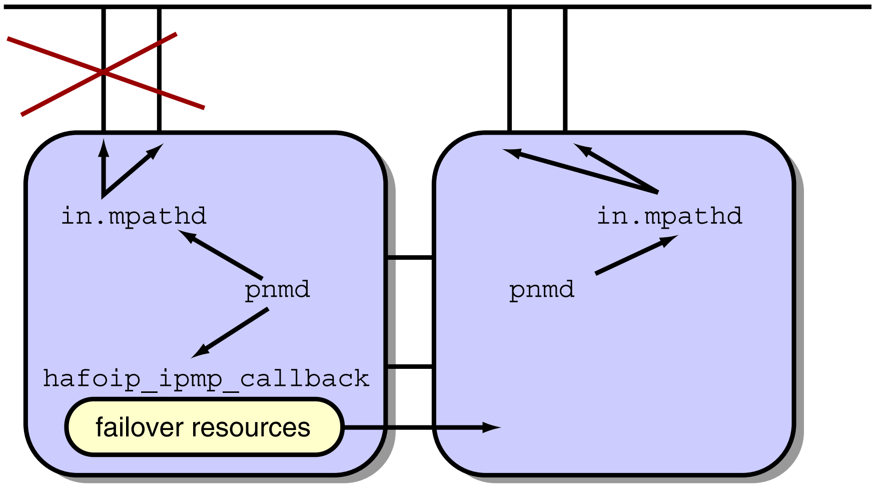


Figure 8-5 IPMP Cluster Integration

Viewing Cluster-Wide IPMP Information With the `scstat` Command

In the Sun Cluster 3.1 software environment, running the `scstat -i` command from any node shows the status of IPMP group members on all the nodes. If you use the `scstat` command without any options, then the IPMP information is included among all the other cluster status information.

```
# scstat -i

-- IPMP Groups --

              Node Name            Group   Status          Adapter   Status
              ---------            -----   ------          -------   ------
  IPMP Group: vincent             therapy Online          qfe2      Online
  IPMP Group: vincent             therapy Online          qfe1      Online

  IPMP Group: theo                therapy Online          qfe2      Online
  IPMP Group: theo                therapy Online          qfe1      Online
```

Exercise: Configuring and Testing IPMP

Perform the following tasks on all cluster nodes. It is assumed that the Sun Cluster software is installed and operational, and that the only IPMP configuration is the single-adapter group called `sc_ipmp0` automatically created by `scinstall`.

In this exercise, you complete the following tasks:

- Task 1 – Verifying the `local-mac-address?` Variable
- Task 2 – Verifying the Adapters for the IPMP Group
- Task 3 – Verifying or Entering Test Addresses in the `/etc/hosts` File
- Task 4 – Creating `/etc/hostname.xxx` Files
- Task 5 – Rebooting and Verifying That IPMP Is Configured
- Task 6 – Verifying IPMP Failover and Failback

Preparation

No preparation is required for this exercise.

Task 1 – Verifying the `local-mac-address?` Variable

Perform the following steps on each node in the cluster

1. Verify that the EEPROM `local-mac-address?` variable is set to `true`:

   ```
   # eeprom "local-mac-address?"
   ```

   It is set to `true` by the `scinstall` utility at cluster install time, so the only reason it should be `false` now is if somebody changed it back manually.

2. If for some reason you need to modify the variable, do so and reboot the node.

Task 2 – Verifying the Adapters for the IPMP Group

Perform the following steps on each node of the cluster:

1. Make sure you know which are the redundant adapters on the public network. You might have already written this down in the exercises for Module 3, "Preparing for Installation and Understanding Quorum Devices."
2. Your primary public network adapter should be the only one currently configured on the public net. You can verify this with:

```
# ls -l /etc/hostname.*
# ifconfig -a
```

3. You can verify your secondary public network adapter by:
   a. Making sure it is not configured as a private transport
   b. Making sure it can snoop public network broadcast traffic:

```
# ifconfig ifname plumb
# snoop -d ifname
(other window or node)# ping -s pubnet_broadcast_addr
```

Task 3 – Verifying or Entering Test Addresses in the `/etc/hosts` File

It is a good idea, although not required, to have test IP addresses in the hosts file. While one node does not need to know anything about another node's test addresses, it is advisable to have all test addresses in the hosts file for all nodes to indicate that these addresses are reserved and what they are reserved for.

Perform the following steps:

1. Verify with your instructor which IP addresses should be used for the test interfaces for each adapter on each node.
2. Enter the IP addresses in `/etc/hosts` on each node if they are not already there.

   The following is only an example. It really does not matter if you use the same names as another group working on another cluster, as long as the IP addresses are different:

```
# IPMP TEST ADDRESSES
172.20.4.194    vincent-qfe1-test
172.20.4.195    vincent-qfe2-test
172.20.4.197    theo-qfe1-test
```

```
172.20.4.198    theo-qfe2-test
```

Task 4 – Creating /etc/hostname.*xxx* Files

Perform the following:

On each node, create the appropriate /etc/hostname.*xxx* files to place adapters in IPMP groups. The group name is unimportant. The following are just examples, so use the adapter names that are configured on your public network.

Note – The scinstall utility, starting in Sun Cluster 3.1 8/05, should have already modified the /etc/hostname.*xxx* file (one of the two below) that already existed at the time scinstall was used to configure your cluster. Your adapter was placed in a singleton IPMP group called sc_ipmp0. You will be completely overwriting this file to place the adapter in a real, multiadapter IPMP group, as in the examples.

```
# vi /etc/hostname.qfe1
 vincent group therapy up
 addif vincent-qfe1-test -failover deprecated up
# vi /etc/hostname.qfe2
 vincent-qfe2-test group therapy -failover deprecated up
```

Task 5 – Rebooting and Verifying That IPMP Is Configured

The following steps can be performed on one node at a time, so that the cluster stays active the whole time:

1. Reboot the node.
2. Verify the new IPMP configuration with **ifconfig -a.**
3. Verify IPMP cluster-wide status with **scstat -i.**

Task 6 – Verifying IPMP Failover and Failback

Perform the following steps on at least one of your nodes:

1. From outside of the cluster, launch **ping -s *nodename***, and keep it running.
2. If you have physical access to your cluster hardware, unplug the Ethernet cable from the network adapter which currently has the node physical interface on it.

   If you have no physical access, you can sabotage your adapter with:

   ```
   # ifconfig adapter_name modinsert ldterm@2
   ```
3. Observe the node messages (on the console or in the `/var/adm/messages` file).
4. Observe the output of the **scstat -i** command.
5. Observe the behavior of your command from step 1 (keep it running).

   If you have physical access, reattach the broken cable. If you have no physical access, use the following to repair your sabotage:

   ```
   # ifconfig adapter_name modremove ldterm@2
   ```
6. Observe the messages and the behavior of your command from Step 1.
7. Observe the output of the **scstat -i** command.

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 9

Introducing Data Services, Resource Groups, and HA-NFS

Objectives

Upon completion of this module, you should be able to:

- Describe how data service agents enable a data service in a cluster to operate properly
- List the components of a data service agent
- Describe data service packaging, installation, and registration
- Describe the primary purpose of resource groups
- Differentiate between failover and scalable data services
- Describe how to use special resource types
- List the components of a resource group
- Differentiate between standard, extension, and resource group properties
- Describe the resource group configuration process
- List the primary functions of the `scrgadm` command
- Use the `scswitch` command to control resources and resource groups
- Use the `scstat` command to view resource and group status
- Use the `scsetup` utility for resources and resource group operations

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- What is a data service agent for the Sun Cluster 3.1 software?
- What is involved in fault monitoring a data service?
- What is the difference between a resource and a resource group?
- What are the different types of properties?
- What are the specific requirements for setting up NFS in the Sun Cluster 3.1 software environment?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)

Introducing Data Services in the Cluster

The Sun Cluster 3.1 software framework makes applications highly available, minimizing application interruptions after any single failure in the cluster.

In addition, some applications, such as Apache Web Server software, are supported not only with high availability features but also in a scalable configuration. This configuration allows the service to run on multiple nodes of the cluster simultaneously while providing a single IP address to the client.

Off-the-Shelf Applications

For most applications supported in the Sun Cluster software environment, software customization is not required to enable an application to run correctly in the cluster. A Sun Cluster 3.1 software agent is provided to enable the data service to run correctly in the cluster environment.

Application Requirements

Identify requirements for all of the data services *before* you begin Solaris OS and Sun Cluster software installation. Failure to do so might result in installation errors that require you to completely reinstall the Solaris OS and Sun Cluster software.

Determining the Location of the Application Binaries

You can install the application software and application configuration files on one of the following locations.

- The local disks of each cluster node

  Placing the software and configuration files on the individual cluster nodes lets you upgrade application software later without shutting down the service. The disadvantage is that you have several copies of the software and configuration files to maintain and administer.

- A global file system or failover file system in the shared storage

  If you put the application binaries on a shared file system, you have only one copy to maintain and manage. However, you must shut down the data service in the entire cluster to upgrade the application software. If you can spare a small amount of downtime for upgrades, place a single copy of the application and configuration files on a shared global or failover file system.

Sun Cluster 3.1 Software Data Service Agents

The word *agent* is an informal name for a set of components, written specifically for Sun Cluster 3.1 software, that enable a data service in a cluster to operate properly. Figure 9-1 shows the relationship between a standard application and the data service agent.

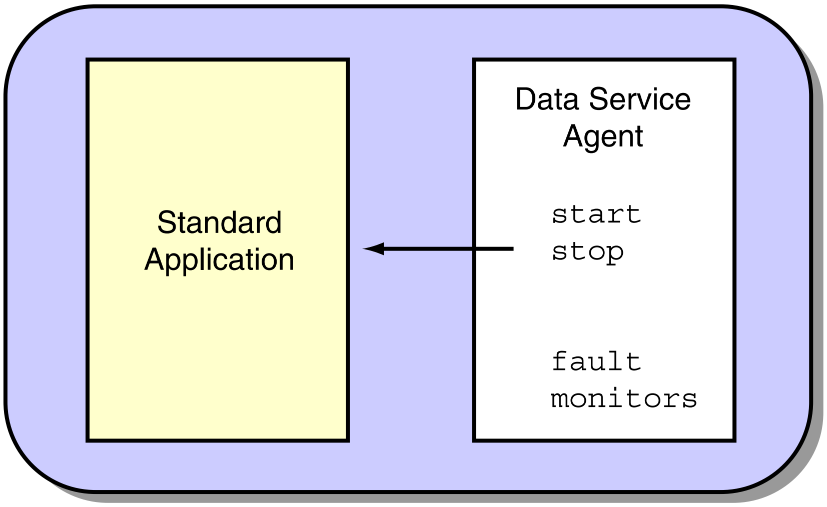


Figure 9-1 Standard Application and Data Service Agent

Reviewing Components of a Data Service Agent

Typical components of a data service agent include the following:

- Methods to start and stop the service in the cluster
- Fault monitors for the service
- Methods to start and stop the fault monitoring
- Methods to validate configuration of the service in the cluster
- A registration information file which allows the Sun Cluster software to store all the information about the methods into the CCR

  You then only need to reference a *resource type* to refer to all the components of the agent.

Sun Cluster software provides an API that can be called from shell programs, and an API which can be called from C or C++ programs. Most of the program components of data service methods that are supported by Sun products are actually compiled C and C++ programs.

Fault Monitor Components

Fault monitoring components specific to data services in Sun Cluster 3.1 software are run on the local node only. This is the same node which is running the data service. Fault monitoring components are intended to detect application failure, and can suggest either application restarts or failovers in the cluster. Unlike Sun Cluster 2.*x* software, no remote fault monitoring is running on backup nodes that could suggest a takeover.

While the actual capabilities of these fault monitors is application-specific and often poorly documented, the general strategy for fault monitors in the Sun Cluster 3.1 environment is to monitor the health of the following:

- The daemons, by placing them under control of the *process monitoring facility* (`rpc.pmfd`). This facility calls action scripts if data service daemons stop unexpectedly.
- The service, by using client commands.

Note – Data service fault monitors do *not* need to monitor the health of the public network itself, as this is already done by the combination of `in.mpathd` and `pnmd`, as described in Module 8, “Managing the Public Network With IPMP.”

Introducing Data Service Packaging, Installation, and Registration

You install most agents for Sun Cluster 3.1 software data services as separate Solaris OS packages using the `pkgadd` command. Agent packages released along with Sun Cluster, whether or not Sun Cluster software is bundled with other Java System applications, come on two different CD's. These are the following:

- Agents for the Java System Applications are on the CDROM 2 of 2, in the `Product/sun_cluster_agents` directory.
- Agents for other applications are on the Agents CDROM.

The `scinstall` utility includes a menu item which allows you to install these packages from the menu interface rather than using the `pkgadd` command, from either of the above two arenas.

Some agents are supplied with the application software, rather than with the cluster software.

Data Service Packages and Resource Types

Each data service agent encapsulates all the information about the agent as a resource type.

When this resource type is registered with the cluster software, you do not need to know the location or names of the components of the agent. You only need to reference an application instance's resource type to determine all the correct information about methods and fault monitors for that component.

Note – The package that you add by using the `pkgadd` command to install an agent, and the corresponding resource type might have different names. For example, when you install the `SUNWschtt` package, a resource type called `SUNW.iws` becomes available.

Introducing Resources, Resource Groups, and the Resource Group Manager

Data services are placed under the control of the cluster by configuring the services as *resources* within *resource groups*. The rgmd daemon is the resource group manager, which controls all activity having to do with resources and resource groups, as shown in Figure 9-2. That is, the rgmd daemon controls all data service activity within the cluster.

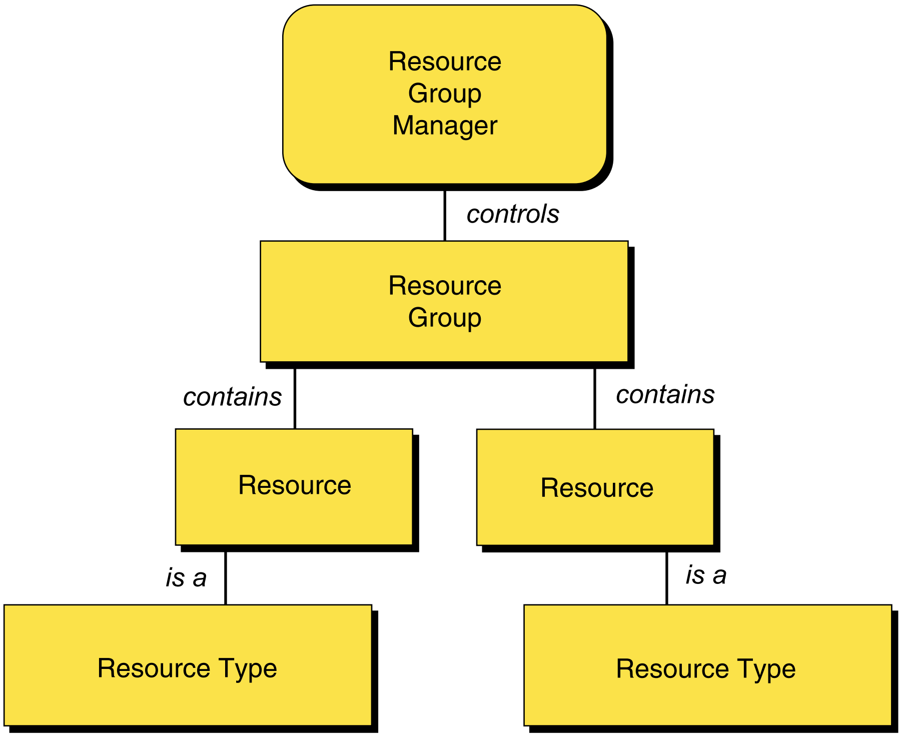


Figure 9-2 Resource Group Manager

Resources

In the context of clusters, the word *resource* refers to any element above the layer of the cluster framework which can be turned on or off, and can be monitored in the cluster.

The most obvious example of a resource is an instance of a running data service. For example, an Apache Web Server, with a single httpd.conf file, counts as one resource.

Each resource is an instance of a specific type. For example, the type for Apache Web Server is `SUNW.apache`.

Other types of resources represent IP addresses and storage that are required by the cluster.

A particular resource is identified by the following:

- Its type, though the type is not unique for each instance
- A unique name which is used as input and output within utilities
- A set of properties, which are parameters that define a particular resource

You can configure multiple resources of the same type in the cluster, either in the same or different resource groups.

For example, you might want to run two Apache Web Server application services on different nodes in the cluster. In this case, you have two resources, both of type `SUNW.apache`, in two different resource groups.

Resource Groups

Resource groups are collections of resources. Resource groups are either failover or scalable.

Failover Resource Groups

For failover applications in the cluster, the *resource group* becomes the unit of failover. That is, the resource group is the collection of services that always run together on one node of the cluster at one time, and simultaneously failover or switchover to another node.

Scalable Resource Groups

For scalable applications, the resource group describes the collection of services that simultaneously run on one or more nodes.

You must configure two separate resource groups for scalable applications. Scalable applications are described in detail in Module 10, "Configuring Scalable Services and Advanced Resource Group Relationships."

Number of Data Services in a Resource Group

You can put multiple data services in the same resource group.

For example, you could run two Sun Java System Web Server (formerly known as Sun ONE Web Server) application services, a Sun Java System Directory Server (formerly known as Sun ONE Directory Server) software identity management service, and a Sybase database as four separate resources in the same failover resource group. These applications always run on the same node at the same time, and always simultaneously migrate to another node.

On the other hand, you can put all four of the preceding services in separate resource groups. The services could still run on the same node, but would failover and switchover independently.

Benefits of Putting All Services in a Single Resource Group

If you put all data services in the same failover resource group of a two-node cluster, the node that is *not* currently hosting the resource group is in a pure backup mode. Different services will not run on different nodes.

Some customers prefer to deploy services in a single resource group because this configuration provides more predictable behavior. If the node currently running all the data services is performing optimally, and there is a failover, you can predict that the new node will have the same performance (assuming equivalent servers) because the node was not doing anything else to start with.

Resource Group Manager

The rgmd daemon maintains all the information about data services that are known to the cluster as resources in resource groups. The rgmd daemon launches all scripts that start resources in resource groups and performs all failovers.

Unlike the rest of the cluster framework, the rgmd daemon is a typical daemon that does not add any special range of operations to the Solaris OS kernel. Most switching and failing services occur through typical Solaris OS activity. For example, the Solaris OS calls methods that directly or indirectly launch daemons. In addition, the ifconfig addif command and the ifconfig removeif command migrate IP addresses that are used for services.

Describing Failover Resource Groups

The following example is provided to help you understand the general concept of resources as part of specific resource groups.

A resource name must be globally unique, not merely unique inside a particular resource group as shown in Figure 9-3.

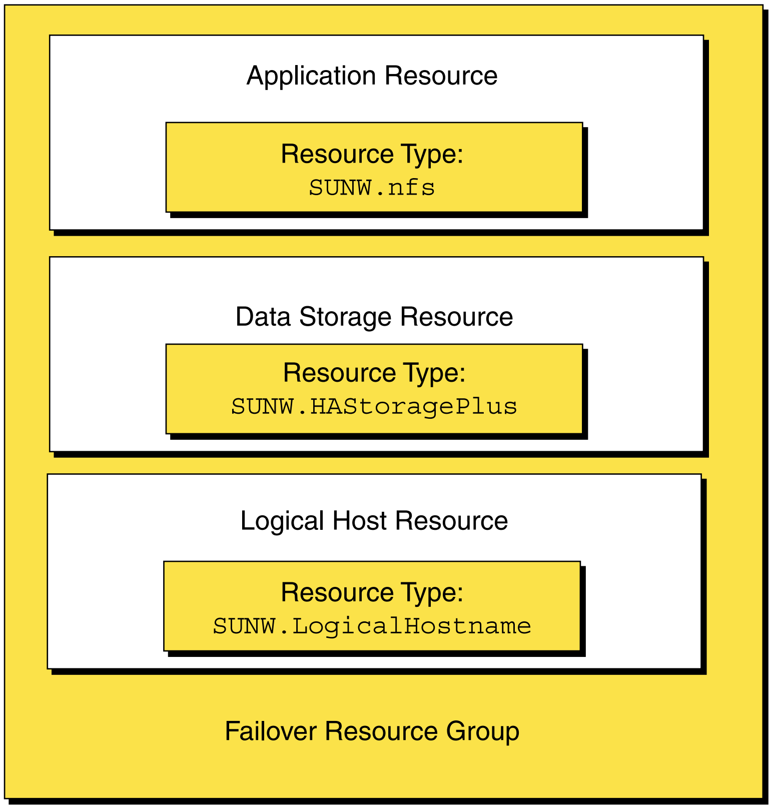


Figure 9-3 Failover Resource Group

The previous failover resource group has been defined with a unique name in the cluster. Configured resource types are placed into the empty resource group. These are known as resources. By placing these resources into the same failover resource group, they failover together.

Resources and Resource Types

Each instance of a data service under cluster control is represented by a resource within a resource group.

Resources exist only inside resource groups. There is no such thing as a disembodied resource that is not a member of a resource group.

Each resource has a *resource type* that describes the type of resource it is (for example, `SUNW.nfs` for an NFS resource).

At least one defined resource type exists for each supported service in the cluster. Some applications that are typically considered to be a single entity, such as an instance of the ORACLE database, actually require two different resources with different types: the ORACLE server and the ORACLE listener.

In addition to resource types that relate to data services, there are a few other special resource types that relate to IP addresses and storage. These resource types are described in "Using Special Resource Types" on page 9-14.

Resource Type Versioning

Sun Cluster 3.1 software gives you the ability to use different versions of resource types. For example, old and new versions of data service agents can co-exist as separate types. Individual resources of an original resource type can be upgraded to a new type on a resource-by-resource basis.

Most data service resources in the Sun Cluster 3.1 software have 3.1 as their version number. In fact, this version number is appended to the resource type name. For example, the official name of the NFS resource type is `SUNW.nfs:3.1`.

When using a resource type name, the version suffix can be dropped if there is no ambiguity. For that matter, the vendor prefix can also be dropped. So, for example, when you initially install Sun Cluster 3.1 software, all of the following names can be used to refer to the `nfs` resource type:

- `SUNW.nfs:3.1`
- `SUNW.nfs`
- `nfs:3.1`
- `nfs`

Using Special Resource Types

The following sections describe special resources that you can include in resource groups to complete the capability of those groups.

The SUNW.LogicalHostname Resource Type

Resources of type SUNW.LogicalHostname represent one or more IP addresses on a particular subnet that will be the logical IP addresses for services in that resource group.

That is, each IP address described by a SUNW.LogicalHostname resource migrates from node to node, along with the services for that group. The client uses these IP addresses to access the services in the cluster.

A large part of what makes cluster failover relatively transparent to the client is that IP addresses migrate along with services of the group. The client always uses the same logical IP address to contact a particular instance of a data service, regardless of which physical node is actually running the service.

The SUNW.SharedAddress Resource Type

Resources of type SUNW.SharedAddress represent a special type of IP address that is required by scalable services. This IP address is configured on the public net of only one node with failover capability, but provides a load-balanced IP address that supports scalable applications that run on multiple nodes simultaneously. This subject is described in greater detail in Module 10, “Configuring Scalable Services and Advanced Resource Group Relationships.”

The SUNW.HAStorage Resource Type

The SUNW.HAStorage resource type is the original storage management type available in all versions of Sun Cluster 3.0 software.

Global device and file system management, including failover from a failed node, are part of the Sun Cluster software framework. So, it might seem redundant to provide a resource type that also manages global devices and file systems. In fact it is *not* redundant.

The resource type SUNW.HAStorage serves the following purposes:

- You use the START method of SUNW.HAStorage to check if the global devices or file systems in question are accessible from the node where the resource group is going online.
- You almost always use the Resource_dependencies standard property to place a dependency so that the real data services depend on the SUNW.HAStorage resource type. In this way, the resource group manager does not try to start services if the storage the services depend on is not available.
- You set the AffinityOn resource property to True, so that the SUNW.HAStorage resource type attempts co-location of resource groups and device groups on the same node, thus enhancing the performance of disk-intensive data services.

  If movement of the device group along with the resource group is impossible (moving service to a node not physically connected to the storage), that is still fine. AffinityOn=true moves the device group for you whenever it can.

The SUNW.HAStorage resource type provides support only for global devices and global file systems. The SUNW.HAStorage resource type does *not* include the ability to unmount file systems on one node and mount them on another. The START method of SUNW.HAStorage is only a check that a global device or file system is already available.

The SUNW.HAStoragePlus Resource Type

The SUNW.HAStoragePlus resource type supersedes the older SUNW.HAStorage type because it incorporates all of the older type's capabilities and adds support for a failover file system.

A failover file system *must* fail over to a node as the service fails over, but can failover *only* to nodes physically connected to the storage.

The advantage of a failover file system is performance. The disadvantage is that the failover file system cannot be used for scalable services or for failover services that need to failover to a node that is not physically connected to the storage.

The SUNW.HAStoragePlus resource type still supports global devices and file systems. The capability of SUNW.HAStoragePlus is a proper superset of SUNW.HAStorage.

The SUNW.HAStoragePlus resource type uses the FilesystemMountpoints property for global and failover file systems. The SUNW.HAStoragePlus resource type understands the difference between global and failover file systems by looking in the vfstab file, where:

- Global file systems have yes in the mount at boot column and global (usually global,logging) in the mount options column. In this case, SUNW.HAStoragePlus behaves exactly like SUNW.HAStorage. The START method is just a check. The STOP method does nothing except return the SUCCESS return code.
- Failover file systems have no in the mount at boot column, and do *not* have the mount option global (just logging). In this case, the STOP method actually unmounts the file system on one node, and the START method mounts the file system on another node.

Guidelines for Using the Storage Resources With Global and Failover File Systems

Although the SUNW.HAStorage resource type is still available for backward compatibility, you should use SUNW.HAStoragePlus in a fresh Sun Cluster 3.1 software installation because it includes all the capabilities of SUNW.HAStorage. Eventually, SUNW.HAStorage will be removed from Sun Cluster software.

When To Use a Global File System

Use a global file system if you want to support the following:

- A scalable service
- A failover service that must failover to a node not physically connected to the storage
- Data for different failover services that are contained in different resource groups
- Data from a node that is not currently running the service

You can still use AffinityOn=true to migrate the storage along with the service, if that makes sense, as a performance benefit.

When To Use a Failover File System

Use a failover file system if you need the following:

- The file system is for a failover service only
- The Nodelist for the resource group contains only nodes that are physically connected to the storage; that is, if the Nodelist for the resource groups is the same as the Nodelist for the device group
- Only services in a single resource group are using the file system

If these conditions are true, a failover file system provides a higher level of performance than a global file system, especially for services that are file system intensive.

Generic Data Service

The Generic Data Service (GDS) is a resource type for making simple applications highly available or scalable using the resource type SUNW.gds by plugging them into the Sun Cluster RGM framework. The GDS composes a fully functional Sun Cluster Resource Type complete with callback methods and a Resource Type Registration file.

Basically the concept is that many different applications can share the same resource type. All you have to do is have a resource type where “what is being launched” is not implied by the resource type itself. Instead the “what is being launched” and “how to probe this application” and “how to stop this application” are just properties that can be set differently for each instance of SUNW.gds.

Many Sun supported data services (for example, DHCP, Samba, and many more) *do not* actually supply new resource types. Rather, they supply configuration scripts that configure resources to represent the application of an instance of SUNW.gds.

In the lab exercises for this module you will get an opportunity to use an instance of the SUNW.gds type to put your own customized application under control of the Sun Cluster software.

Understanding Resource Dependencies and Resource Group Dependencies

You can declare dependency relationships between resources. Starting with Sun Cluster 3.1 9/04 (Update 3), relationships between individual resources can be between resources in the same or different resource groups. You can also configure group dependencies which take into account only the state of a group rather than the state of its resources.

There are three levels of resource dependencies.

Regular Resource Dependencies

If Resource A *depends on* Resource B, then:

- Resource B *must be added* first.
- Resource B *must be started* first (without the dependency, RGM may have been able to start them in parallel).
- Resource A *must be stopped* first (without the dependency, RGM may have been able to stop them in parallel).
- Resource A *must be deleted* first.
- The `rgmd` daemon will not try to start resource A if resource B fails to go online, and will not try to stop resource B if resource A fails to be stopped.

Weak Dependencies (Weaker than Regular Dependencies)

If resource A has a *weak dependency* on resource B, all of the previous bulleted items apply except for the last one. So the weak dependency causes an ordering of starts and stops but is not a strict dependency if the dependee (resource B) is failing to start or the dependant (resource A) fails to stop.

Restart Dependencies (Stronger than Regular Dependencies)

Restart Dependencies have all the attributes of regular dependencies, with the *additional* attribute that if the RGM is told to restart resource B, or is informed that some agent did the restart of resource B itself, the RGM will automatically restart resource A.

Resource Group Dependencies

Resource group dependencies imply an ordering relationship between two groups: If resource group G has a group dependency on resource group H, group G cannot be brought online unless group H is online. Group H cannot be brought offline unless group G is offline. This type of dependency considers the state of the group only rather than the resources inside it.

Note – Resource group dependencies are somewhat limited. They are enforced between groups running on different nodes when you manually try to start or stop resource groups in the wrong dependency order, regardless of which nodes the groups are mastered. However, when failovers occur, resource group dependencies are only strictly enforced between groups that are stopping and starting on the same node.

Configuring Resource and Resource Groups Through Properties

You configure specific resources within the cluster by defining values for resource properties and resource group properties. Properties consist of a set of `name=value` pairs.

These properties are used by the data service agent and by `rgmd`. There is no way they can be accessed directly by the application software itself, as it is cluster-unaware software.

Some properties are essential for running the service in the cluster. Scripts specifically used to run the service in the cluster can read these property values and use them to locate configuration files, or can use them to pass command-line parameters to the actual services.

Other properties are essential only for data service fault monitoring. Misconfiguring these properties might leave the service running fine, but cause fault monitoring to fail.

Multiple properties can be assigned to different resources, as shown in Figure 9-4 on page 9-21.

Each resource can literally have dozens of properties. Fortunately, many important properties for particular types of resource are automatically provided with reasonable default values. Therefore, most administrators of Sun Cluster software environments never need to deal with the properties.

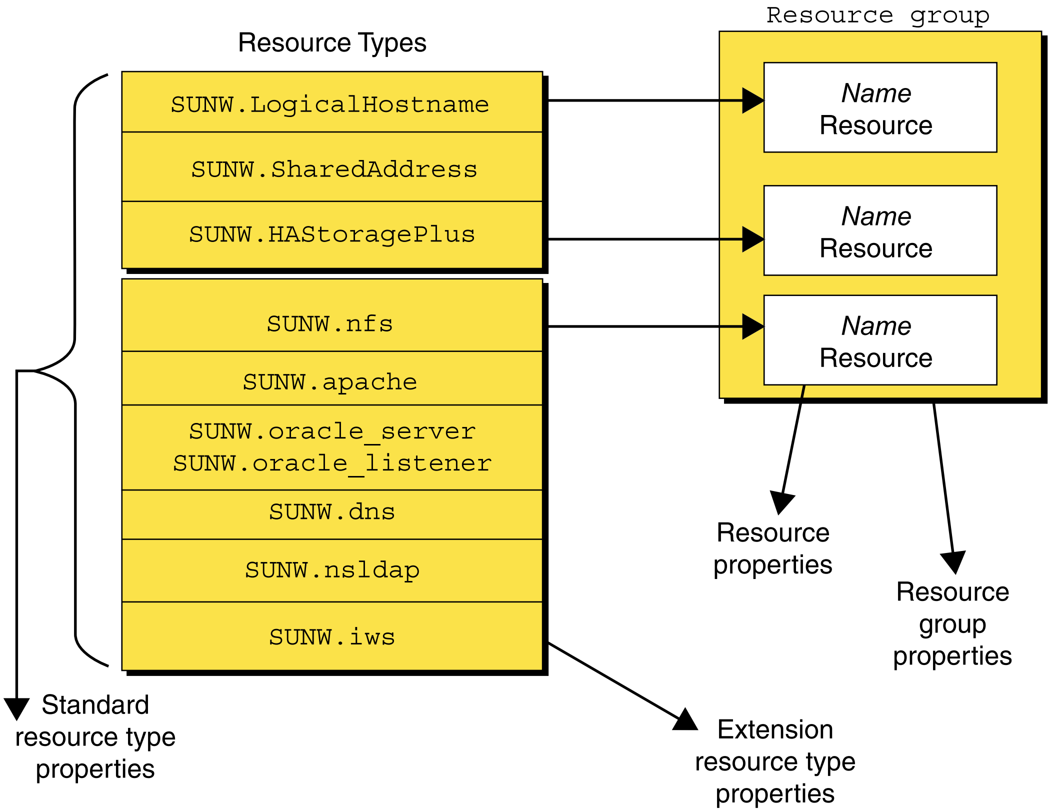


Figure 9-4 Standard and Extension Properties

Standard Resource Properties

The names of standard resource properties can be used with *any* type of resource.

You can access a full list of standard resource properties and their general meanings by typing:

```
# man r_properties
```

Of the dozens of properties listed, only a handful are critical for a particular resource type. Other properties can be ignored, or can have default values that are unlikely to be changed.

Extension Properties

Names of extension properties are specific to resources of a particular type. You can get information about extension properties from the man page for a specific resource type. For example:

```
# man SUNW.apache
# man SUNW.HAStoragePlus
```

If you are setting up the Apache Web Server, for example, you must create a value for the `Bin_dir` property, which points to the directory containing the `apachectl` script that you want to use to start the web server.

The storage resource has the following important extension properties:

```
FilesystemMountpoints=list_of_storage_mount_points
AffinityOn=True/False
```

Use the first of these extension properties to identify which storage resource is being described. The second extension property is a parameter which tells the cluster framework to switch physical control of the storage group to the node running the service. Switching control to the node which is running the service optimizes performance when services in a single failover resource group are the only services accessing the storage.

Many resource types (including LDAP and DNS) have an extension property called `Confdir_list` which points to the configuration.

```
Confdir_list=/global/dnsconflocation
```

Many have other ways of identifying their configuration and data. There is no hard and fast rule about extension properties.

Resource Group Properties

Certain properties apply to an entire resource group. You can get information about these properties by typing:

```
# man rg_properties
```

Examining Some Interesting Properties

The following sections examine the usage of some of the more interesting standard resource properties and resource group properties.

Some Standard Resource Properties

The following properties are standard resource properties, that is, they can have meaningful values for many different types of resources.

The `Resource_dependencies` Property

Resource dependencies, which imply ordering, are configured using standard properties.

```
Resource_dependencies=nfs-stor
```

The `Resource_dependencies_weak` Property

To set up a *weak* dependency, so that resource A has a weak dependency on resource B, set the `Resource_dependencies_weak` property on resource A. (Recall that this implies an ordering, but not a real dependency.)

```
Resource_dependencies_weak=resB
```

The `Resource_dependencies_restart` Property

To set up a *restart* dependency, so that resource A has the strongest type of dependency on resource B, set the `Resource_dependencies_restart` property on resource A. (Recall that this implies that resource A must be restarted if B is restarted, as well as all the other attributes of a regular dependency.)

```
Resource_dependencies_restart=resB
```

The Failover_mode Property

The Failover_mode property describes what should happen to a resource if the resource fails to start up or shut down properly. Table 9-1 describes how values of the Failover_mode property work.

Table 9-1 The Failover_mode Value Operation

Value of the Failover_mode property	**Failure to start**	**Failure to stop**	**Can fault monitor cause RGM to fail RG over?**	**Can fault monitor cause RGM to restart the resource?**
NONE	Other resources in the same resource group can still start (if non-dependent).	The STOP_FAILED flag is set on the resource.	yes	yes
SOFT	The whole resource group is switched to another node.	The STOP_FAILED flag is set on the resource.	yes	yes
HARD	The whole resource group is switched to another node.	The node reboots.	yes	yes
RESTART_ONLY	Other resources in the same resource group can still start (if non-dependent).	The STOP_FAILED flag is set on the resource.	No	Yes
LOG_ONLY	Other resources in the same resource group can still start (if non-dependent).	The STOP_FAILED flag is set on the resource.	No	No

If the STOP_FAILED flag is set, it must be manually cleared by using the scswitch command before the service can start again.

```
# scswitch -c -j resource -h node -f STOP_FAILED
```

Some Resource Group Properties

The following properties are associated with an entire resource group, rather than an individual resource.

The `RG_dependencies` Property

The `RG_dependencies` property will be set on a whole resource group to describe its dependency on another group. If resource group A has the property `RG_dependencies=rgB`, then resource group A cannot be brought online unless resource group B is online. Resource group B must be brought online somewhere, but not necessarily on the same node.

The `Nodelist` Property

The `Nodelist` property for a resource group describes what node or nodes the resource group can run on. If you have more than two nodes in the cluster, you can have as many or as few (but probably no less than two) potential nodes where a resource group can run.

The property is set up in order, from most preferred node to least preferred node.

The `Failback` Property

If the `Failback` property is `TRUE` (not the default), the resource group automatically switches back when a preferred node (earlier in the Nodelist) joins the cluster.

The `Implicit_network_dependencies` Property

The `Implicit_network_dependencies` property is `TRUE` by default. This property makes all the services in the resource group dependent on all the `LogicalHostName` and `SharedAddress` resources. In other words, no services can start if logical IP addresses cannot be brought online. You can set the `Implicit_Network_Dependencies` property to `FALSE` if you have a service which does not depend on logical IP addresses.

The Pingpong_interval Property

This property, whose default value is 3600 seconds (one hour), determines the interval inside which the cluster will refuse to start a resource group on a particular node if the resource group has previously failed on that node or failed to start on that node. The nomenclature derives from the fact that this property *prevents* rapid "ping-ponging" of a resource group back and forth between or among the nodes if something is misconfigured.

The Pathprefix Property

The Pathprefix property points to a directory in a shared-storage file system which will be used for the administrative purposes of services in the resource group.

Currently, the only data service that must have the Pathprefix property set is NFS, which uses the property to find its dfstab file. NFS needs the dfstab file so that it knows what it is supposed to be sharing. NFS also uses the same arena to store file lock state information. File lock state information is typically stored in the /etc directory on a standalone NFS server, but it must be in the shared storage in the cluster.

An NFS resource must have its dfstab file named:

value_of_Pathprefix_for_RG/SUNW.nfs/dfstab.*resource_name*

When you create an NFS resource, the resource checks that the Pathprefix property is set on its resource group, and that the dfstab file exists.

Using the scrgadm Command

You use the scrgadm command to add, remove, modify, and view the configuration of all resources, resource types, and resource groups. You run the command on any one node that is booted into the cluster.

Note – The scrgadm command does *not* show any status. Use the scstat -g command to the show status of resources in the cluster.

Show Current Configuration

Use the following command syntax to show the current resources that are configured:

```
scrgadm -p[v[v]] [-t resource_type_name] [-g resource_group_name] [-j
resource_name]
```

Registering, Changing, or Removing a Resource Type

You use the following command syntax to register (-a), change (-c), or remove (-r) a resource type:

```
scrgadm -a -t resource_type_name [-h RT_installed_node_list] [-f
registration_file_path]
scrgadm -c -t resource_type_name -h RT_installed_node_list
scrgadm -r -t resource_type_name
```

Adding, Changing, or Removing Resource Groups

You must add a resource group before putting resources in it. In addition, you can only delete a resource group which has no resources in it. Use the following command syntax to add, change or remove a resource group:

```
scrgadm -a -g RG_name [-h nodelist] [-y property […]]
scrgadm -c -g RG_name [-h nodelist] -y property [-y property […]]
scrgadm -r -g RG_name
```

The Nodelist property is set with the -h option.

Adding a LogicalHostname or a SharedAddress Resource

Use the following command syntax to add a LogicalHostname (-L) resource or a SharedAddress (-S) resource:

```
scrgadm -a -L -g RG_name [-j resource_name] -l hostnamelist [-n
netiflist] [-y property […]]
scrgadm -a -S -g RG_name -l hostnamelist [-j resource_name] [-n
netiflist][-X auxnodelist] [-y property […]]
```

The LogicalHostname and SharedAddress resources can have multiple IP addresses associated with them *on the same subnet*. Multiple logical IP resources on different subnets must be separate resources, although they can still be in the same resource group. If you do not specify a resource name, it will be the same as the first logical host name following the -l argument.

A logical name given after the -l argument can be either an IPV4 address (associated with an address in the /etc/hosts file), an IPV6 address (associated with an address in the /etc/inet/ipnodes file).

IPV6 addresses which can actually cross router boundaries, called site-local and global addresses, are normally provided automatically by the IPV6 router infrastructure to the server and then automatically configured as virtual IPV6 interfaces.

Ordinarily, you will only be able to configure IPV6 logical hostname resource and shared address resources that are on the same subnets as site-local or global IP addresses that have already been configured via the router infrastructure on your cluster nodes. Therefore it is very important to fully test the configuration of your routers, assuring that they are passing the correct addresses to the cluster nodes and passing the correct network traffic between the client and the node. Only at this time can you successfully configure an IPV6 logical hostname or shared address.

Adding, Changing, or Removing All Resources

Use the following command syntax to add and remove all other resource types, as well as to change properties of all resource types.

```
scrgadm -a -j resource_name -t resource_type_name -g RG_name [-y property
[…]] [-x extension_property […]]
scrgadm -c -j resource_name [-y property […]] [-x extension_property […]]
scrgadm -r -j resource_name
```

Using the scrgadm Command

The following example is based on the assumption that the NFS agent from the Sun Cluster 3.1 Data Services CD has been added by using the `scinstall` or `pkgadd` command. You should also assume that the `dfstab` file has been set up under the `/global/nfs/admin` directory.

1. Add (register) the resource types by typing:

   ```
   # scrgadm -a -t SUNW.nfs
   # scrgadm -a -t SUNW.HAStoragePlus
   ```

2. Add the resource group by typing:

   ```
   # scrgadm -a -g nfs-rg -h vincent,theo
      -y PathPrefix=/global/nfs/admin
   ```

3. Add the logical host name resource by typing:

   ```
   # scrgadm -a -L -l orangecat-nfs -g nfs-rg
   ```

4. Add the `SUNW.HAStoragePlus` resource by typing:

   ```
   # scrgadm -a -j nfs-stor -t SUNW.HAStoragePlus
      -g nfs-rg -x FilesystemMountpoints=/global/nfs
      -x AffinityOn=true
   ```

5. Add the NFS service. Although many services have properties that point to configuration files or binaries, NFS does not because it uses the `Pathprefix` property of the resource group. Dependency on the storage is expressed by using the standard property. Dependency on the `logicalhostname` resource is implied.

   ```
   # scrgadm -a -j nfs-res -t SUNW.nfs -g nfs-rg
      -y Resource_dependencies=nfs-stor
   ```

Setting Standard and Extension Properties

Sun Cluster 3.1 software allows the developer of a resource type to place restrictions on when particular properties can be set or changed. Each property for a particular type has a "tunability" characteristic that can be one of the following:

- `at_creation` – You can only set the property as you do the `scrgadm -a` command to add the resource.
- `when_disabled` – You can change with the `scrgadm -c` command if the resource is disabled.
- `anytime` – You can change anytime with the `scrgadm -c` command.

Controlling Resources and Resource Groups By Using the `scswitch` Command

The following examples demonstrate how to use the `scswitch` command to control the state of resource groups, resources, and data service fault monitors.

Resource Group Operations

Use the following commands to do the following:

- Take a resource group offline:

  ```
  # scswitch -F -g nfs-rg
  ```

- Online an offline or unmanaged resource group. Enable all disabled resources:

  ```
  # scswitch -Z -g nfs-rg
  ```

- Switch a failover resource group to another node or from offline state to a node. If no node mentioned, it goes to the preferred node:

  ```
  # scswitch -z -g nfs-rg [-h node ]
  ```

- Restart a resource group:

  ```
  # scswitch -R -h node -g nfs-rg
  ```

- Evacuate all resources and resource groups from a node:

  ```
  # scswitch -S -h node
  ```

Resource Operations

Use the following commands to do the following:

- Disable a resource and its fault monitor:

  ```
  # scswitch -n -j nfs-res
  ```

- Enable a resource and its fault monitor:

  ```
  # scswitch -e -j nfs-res
  ```

- Clear the `STOP_FAILED` flag:

  ```
  # scswitch -c -j nfs-res -h node -f STOP_FAILED
  ```

Fault Monitor Operations

Use the following commands to do the following:

- Disable the fault monitor for a resource:

```
# scswitch -n -M -j nfs-res
```

- Enable a resource fault monitor:

```
# scswitch -e -M -j nfs-res
```

Note – Do not manually stop any resource group operations that are underway. Operations, such as those initiated by the `scswitch` command, must be allowed to complete.

Resource Group and Resource Transitions

The diagram in Figure 9-5 summarizes the resource and resource group transitions. Note the following:

- The `scswitch -Z` always enables any disabled resources in the group.
- The `scswitch -z` preserves the state of the disabled/enabled flag for a resource. It can be used instead of `scswitch -Z` to bring a group from Offline to Online if you want to preserve that flag.

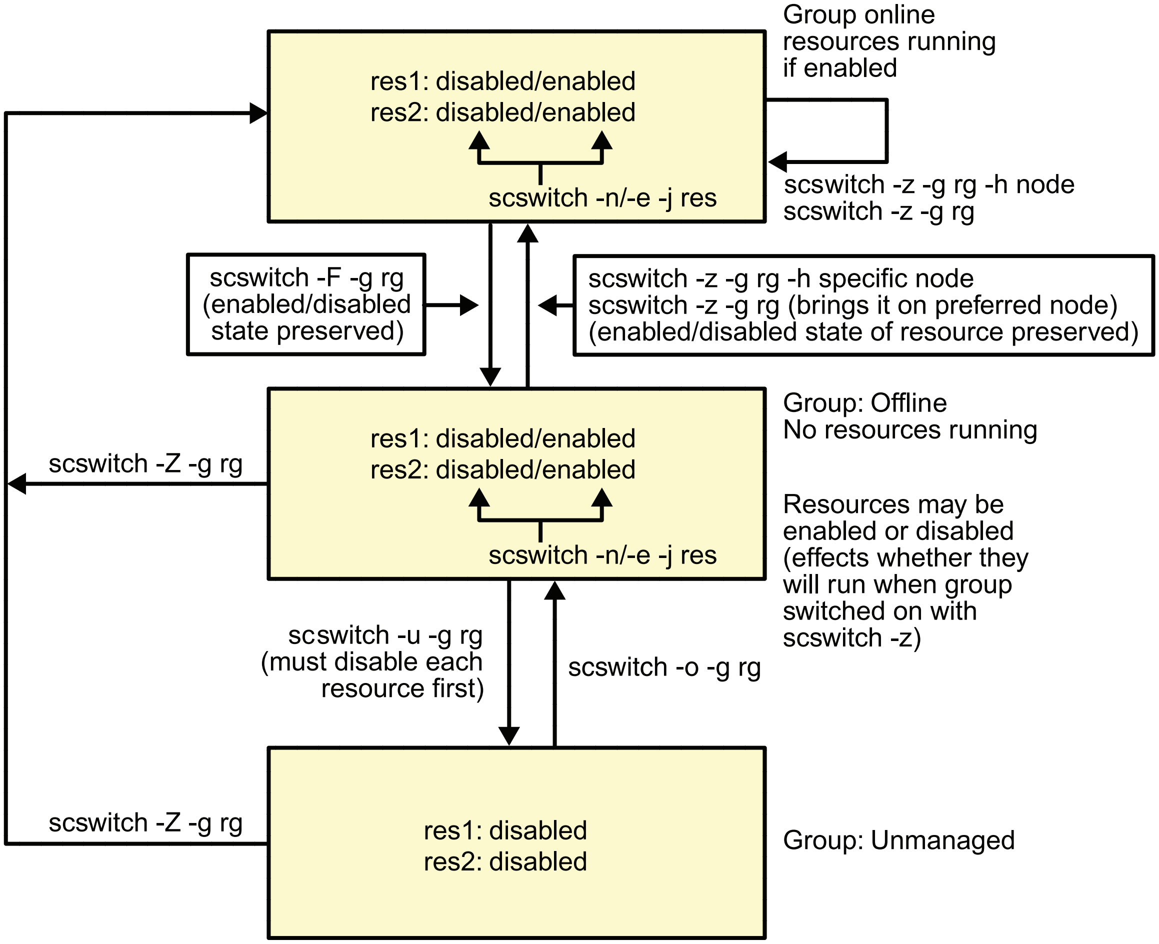


Figure 9-5 Resource Group and Resource Transitions

Confirming Resource and Resource Group Status Using the scstat Command

The scstat -g command and option show the state of all resource groups and resources. If you use the scstat command without the -g, it shows the status of everything, including disk groups, nodes, quorum votes, and transport. For failover resources or resource groups, the output always shows *offline* for a node which the group is not running on, even if the resource is a global storage resource type that is accessible everywhere.

Using the scstat Command for a Single Failover Application

Use the scstat -g command as follows to show the state of resource groups and resources for a single failover application:

```
# scstat -g

-- Resource Groups and Resources --

            Group Name          Resources
            ----------          ---------
 Resources: nfs-rg              orangecat-nfs nfs-stor nfs-res


-- Resource Groups --

            Group Name          Node Name           State
            ----------          ---------           -----
     Group: nfs-rg              vincent             Online
     Group: nfs-rg              theo                Offline


-- Resources --

            Resource Name       Node Name   State     Status Message
            -------------       ---------   -----     --------------
  Resource: orangecat-nfs       vincent    Online    Online -
LogicalHostname online.
  Resource: orangecat-nfs       theo       Offline   Offline

  Resource: nfs-stor            vincent    Online    Online
  Resource: nfs-stor            theo          Offline   Offline

  Resource: nfs-res             vincent       Online    Online - Service
is online.
  Resource: nfs-res             theo          Offline   Offline
```

Using the scsetup Utility for Resource and Resource Group Operations

The scsetup utility has an extensive set of menus pertaining to resource and resource group management. These menus are accessed by choosing Option 2 (Resource Group) from the main menu of the scsetup utility.

The scsetup utility is intended to be an intuitive, menu-driven interface which guides you through the options without having to remember the exact command-line syntax. The scsetup utility calls the scrgadm, scswitch, and scstat commands that have been described in previous sections of this module.

The Resource Group menu for the scsetup utility looks like the following:

```
*** Resource Group Menu ***

    Please select from one of the following options:

        1) Create a resource group
        2) Add a network resource to a resource group
        3) Add a data service resource to a resource group
        4) Resource type registration
        5) Online/Offline or Switchover a resource group
        6) Enable/Disable a resource
        7) Change properties of a resource group
        8) Change properties of a resource
        9) Remove a resource from a resource group
       10) Remove a resource group
       11) Clear the stop_failed error flag from a resource

        ?) Help
        s) Show current status
        q) Return to the main menu

    Option:
```

Exercise: Installing and Configuring the Sun Cluster HA for the NFS Software Package

In this exercise, you complete the following tasks:

- Task 1 – Preparing to Register and Configure the Sun Cluster HA for NFS Data Service
- Task 2 – Registering and Configuring the Sun Cluster HA for NFS Data Service Software Package
- Task 3 – Verifying Access by NFS Clients
- Task 4 – Observing Sun Cluster HA for NFS Failover Behavior
- Task 5 – Generating Cluster Failures and Observing Behavior of the NFS Failover
- Task 6 – Configuring NFS to Use a Failover File System
- Task 7 – Making a Customized Application Fail Over With a Generic Data Service Resource
- Task 8 – Viewing and Managing Resources and Resource Groups Through SunPlex Manager

Preparation

The following tasks are explained in this section:

- Preparing to register and configure the Sun Cluster HA for NFS data service
- Registering and configuring the Sun Cluster HA for NFS data service
- Verifying access by NFS clients
- Observing Sun Cluster HA for NFS failover behavior

Note – During this exercise, when you see italicized terms, such as ***`IPaddress`***, ***`enclosure_name`***, ***`node1`***, or ***`clustername`***, embedded in a command string, substitute the names appropriate for your cluster.

Task 1 – Preparing to Register and Configure the Sun Cluster HA for NFS Data Service

In earlier exercises in Module 6, "Using VERITAS Volume Manager for Volume Management," or Module 7, "Managing Volumes With Solaris™ Volume Manager (Solstice DiskSuite™ Software)," you created the global file system for NFS. Confirm that this file system is available and ready to configure for Sun Cluster HA for NFS.

Perform the following steps:

1. Install the Sun Cluster HA for NFS data service software package on *all* nodes by running the `scinstall` command. Select Option 3 from the interactive menus.

Note – You must furnish the full path to the data service software. The full path name is the location of the `.cdtoc` file and not the location of the packages.

---

2. Modify the `/etc/default/nfs` file on *all* nodes. Uncomment the line that currently reads:

   ```
   #NFS_SERVER_VERSMAX=4
   ```

   and change the maximum version offered by the server to 3. The line should end up reading (note that the comment is *removed):*

   ```
   NFS_SERVER_VERSMAX=3
   ```

Note – At the time of writing of this course, NFS version 4 is *not* supported in the cluster.

---

3. Log in to Node 1 as user `root`.
4. Verify that your cluster is active.

   ```
   # scstat -p
   ```

5. Verify that the `/global/nfs` file system is mounted and ready for use.

   ```
   # df -k
   ```

6. Add an entry to the `/etc/hosts` file on each cluster node and on the administrative workstation for the logical host name resource *`clustername`*`-nfs`. Substitute the IP address supplied by your instructor.

```
IP_address    clustername-nfs
```

Note – In the RLDC, the `/etc/hosts` file on the vnchost already contains the appropriate entry for each cluster. Verify that the entry for your cluster exists on the vnchost, and use the same IP address to create the entry on your cluster nodes.

Perform the remaining steps on just *one node* of the cluster.

7. Create the administrative directory which will contain the `dfstab.nfs-res` file for the NFS resource.

```
# cd /global/nfs
# mkdir admin
# cd admin
# mkdir SUNW.nfs
```

8. Create the `dfstab.nfs-res` file in the `/global/nfs/admin/SUNW.nfs` directory. Add the entry to share the `/global/nfs/data` directory.

```
# cd SUNW.nfs
# vi dfstab.nfs-res

share -F nfs -o rw /global/nfs/data
```

9. Create the directory specified in the `dfstab.nfs-res` file.

```
# cd /global/nfs
# mkdir /global/nfs/data
# chmod 777 /global/nfs/data
# touch /global/nfs/data/sample.file
```

Note – You are changing the mode of the data directory only for the purposes of this lab. In practice, you would be more specific about the share options in the `dfstab.nfs-res` file.

Task 2 – Registering and Configuring the Sun Cluster HA for NFS Data Service Software Package

Perform the following steps to register the Sun Cluster HA for NFS data service software package:

1. From one node, register the NFS and `SUNW.HAStorage` resource types.

```
# scrgadm -a -t SUNW.nfs
# scrgadm -a -t SUNW.HAStoragePlus
# scrgadm -p
```

Note – The `scrgadm -a` command only produces output to the window if there are errors executing the commands. If the command executes and returns, that indicates successful creation.

2. Create the failover resource group. Note that on a three node cluster the nodelist (specified with `-h`) can include nodes not physically connected to the storage if you are using a global file system.

```
# scrgadm -a -g nfs-rg -h node1,node2,[node3]
-y Pathprefix=/global/nfs/admin
```

3. Add the logical host name resource to the resource group.

```
# scrgadm -a -L -g nfs-rg -l clustername-nfs
```

4. Create the `SUNW.HAStoragePlus` resource. If all of your nodes are connected to storage (2-node cluster, for example), you should set the value of `AffinityOn` to `TRUE`. If you have a 3rd, non-storage node, you should set the value of `AffinityOn` to `FALSE`.

```
# scrgadm -a -j nfs-stor -g nfs-rg \
-t SUNW.HAStoragePlus \
-x FilesystemMountpoints=/global/nfs \
-x AffinityOn=[TRUE|FALSE]
```

5. Create the `SUNW.nfs` resource.

```
# scrgadm -a -j nfs-res -g nfs-rg -t SUNW.nfs \
-y Resource_dependencies=nfs-stor
```

6. Enable the resources and the resource monitors, manage the resource group, and switch the resource group into the online state.

```
# scswitch -Z -g nfs-rg
```

7. Verify that the data service is online.

```
# scstat -g
```

Task 3 – Verifying Access by NFS Clients

Perform the following steps to verify that NFS clients can access the file system of the Sun Cluster HA for NFS software package:

1. On the administration workstation, verify that you can access the cluster file system.

   ```
   (# or $) ls /net/clustername-nfs/global/nfs/data
   sample.file
   ```

2. On the administration workstation, copy the `test.nfs` file from the lab files arena into your home directory.

3. Edit the `$HOME/test.nfs` script and verify that the logical host name and NFS file system names are correct.

   When this script is running, it creates and writes a file containing a timestamp to your NFS-mounted file system. The script also displays the file to standard output (`stdout`). This script times how long the NFS data service is interrupted during switchovers and takeovers.

Task 4 – Observing Sun Cluster HA for NFS Failover Behavior

Now that the Sun Cluster HA for NFS environment is working properly, test its high-availability operation by performing the following steps:

1. On the administration workstation, start the `$HOME/test.nfs` script.
2. On one node of the cluster, determine the name of the node currently hosting the data services of the Sun Cluster HA for NFS software package.
3. On one node of the cluster, use the `scswitch` command to transfer control of the NFS service from one node to another.

   ```
   # scswitch -z -h dest-node -g nfs-rg
   ```

   Substitute the name of your offline node for *dest-node*.
4. Observe the messages displayed by the `test.nfs` script.
5. How long was the data service interrupted during the switchover from one physical host to another?

   ____
6. Use the `mount` and `share` commands on all nodes to verify which file systems the nodes are now mounting and exporting.

   ____

   ____

   ____
7. Use the `ifconfig` command on all nodes to observe the additional IP address associated with the Logical Hostname resource configured as a virtual interface on one of the adapters in your IPMP group.

   ```
   # ifconfig -a
   ```
8. On one node of the cluster, use the `scswitch` command to transfer control of the NFS service back to its preferred host.

   ```
   # scswitch -z -h dest-node -g nfs-rg
   ```

Task 5 – Generating Cluster Failures and Observing Behavior of the NFS Failover

Generate failures in your cluster to observe the recovery features of Sun Cluster 3.1 software. If you have physical access to the cluster, you can physically pull out network cables or power down a node. If you do not have physical access to the cluster, you can still bring a node to the OK prompt using a break signal through your terminal concentrator.

Try to generate the following failures:

- Single public network interface failure (observe IPMP failover)
- Total public network failure on a single node (pull both public network interfaces)
- Single transport failure
- Dual transport failure
- Node failure (power down a node or bring it to the `ok` prompt)

Task 6 – Configuring NFS to Use a Failover File System

In this task, you configure NFS to use a failover file system rather than a global file system.

Perform the following steps:

1. Disable the NFS resource group.

   ```
   # scswitch -F -g nfs-rg
   ```

2. If you have more than two nodes, make sure that the NFS resource group is enabled to run only on the nodes physically connected to the storage.

   ```
   # scrgadm -c -g nfs-rg -y Nodelist=node1,node2
   # scswitch -n -j nfs-res
   # scswitch -n -j nfs-stor
   # scrgadm -c -j nfs-stor -x AffinityOn=TRUE
   # scswitch -e -j nfs-stor
   # scswitch -e -j nfs-res
   ```

3. Unmount the global NFS file system.

   ```
   # umount /global/nfs
   ```

4. On each node, edit the `/etc/vfstab` file to make `/global/nfs` a local file system.
   a. Change `yes` to `no` in the mount-at-boot column.
   b. Remove the word `global` from the mount options (keep the word `logging`, or add logging if you forgot before).
5. Restart the NFS resource group.

```
# scswitch -z -g nfs-rg
```

6. Observe the file system behavior. The file system should be mounted only on the node running the NFS service, and should failover appropriately.

Task 7 – Making a Customized Application Fail Over With a Generic Data Service Resource

In this task you can see how easy it is to get any daemon to fail over in the cluster, by using the Generic Data Service (so that you do not have to invent your own resource type).

Perform the following steps on the nodes indicated:

1. On *all nodes* (or do it on one node and copy the file to other nodes in the same location), create a "daemon" which will represent your customized application:

```
# vi /var/tmp/myappdaemon
#!/bin/ksh
while :
do
   sleep 10
done
```

2. Make sure the file is executable on all nodes.
3. From *any one node*, create a new failover resource group for your application:

```
# scrgadm -a -g myapp-rg -h node1,node2,[node3]
```

4. From *one node*, register the Generic Data Service resource type:

```
# scrgadm -a -t SUNW.gds
```

5. From *one node*, create the new resource and enable the group:

```
# scrgadm -a -g myapp-rg -j myapp-res -t SUNW.gds \
-x Start_Command=/var/tmp/myappdaemon -x \
Probe_Command=/bin/true -x Network_aware=false

# scswitch -Z -g myapp-rg
```

6. Verify the behavior of your customized application.
   a. Verify that you can manually switch the group from node to node.
   b. Kill the daemon. Wait a little while and note that it restarts on the same node. Wait until `scstat -g` shows that the resource is fully online again.
   c. Repeat step b a few times. Eventually, the group will switch over to the other node.

Task 8 – Viewing and Managing Resources and Resource Groups Through SunPlex Manager

Perform the following steps on your administration workstation:

1. In a web browser, log in to Sun Java Web Console on any cluster node:

   `https://`*`nodename`*`:6789`

2. Log in as root and enter the SunPlex Manager Application.
3. Click on the `Resource Groups` folder on the left.
4. Investigate the status information and graphical topology information that you can see regarding resource groups.
5. View the details of a resource group by clicking on its name.
6. Use the Switch Primary button to switch the primary node for a resource group.

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 10

Configuring Scalable Services and Advanced Resource Group Relationships

Objectives

Upon completion of this module, you should be able to:

- Describe the characteristics of scalable services
- Describe the function of the load-balancer
- Create the failover resource group for the SharedAddress resource
- Create the scalable resource group for the scalable service
- Describe how the SharedAddress resource works with scalable services
- Add auxiliary nodes
- Use the scrgadm command to create these resource groups
- Use the scswitch command to control scalable resources and resource groups
- Use the scstat command to view scalable resource and group status
- Configure and run Apache as a scalable service in the cluster

Relevance

Discussion – The following questions are relevant to understanding the content of this module:

- What type of services can be made scalable?
- What type of storage must be used for scalable services?
- How does the load balancing work?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)

  Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)

Using Scalable Services and Shared Addresses

The scalable service architecture allows some services, such as Apache Web Server, to run simultaneously on multiple nodes as if it were a single server.

Clients connect to such a service using a single IP address called a *shared address*. Clients are not normally aware which node they connect to, nor do they care.

Figure 10-1 shows the architecture of scalable services and shared addresses.

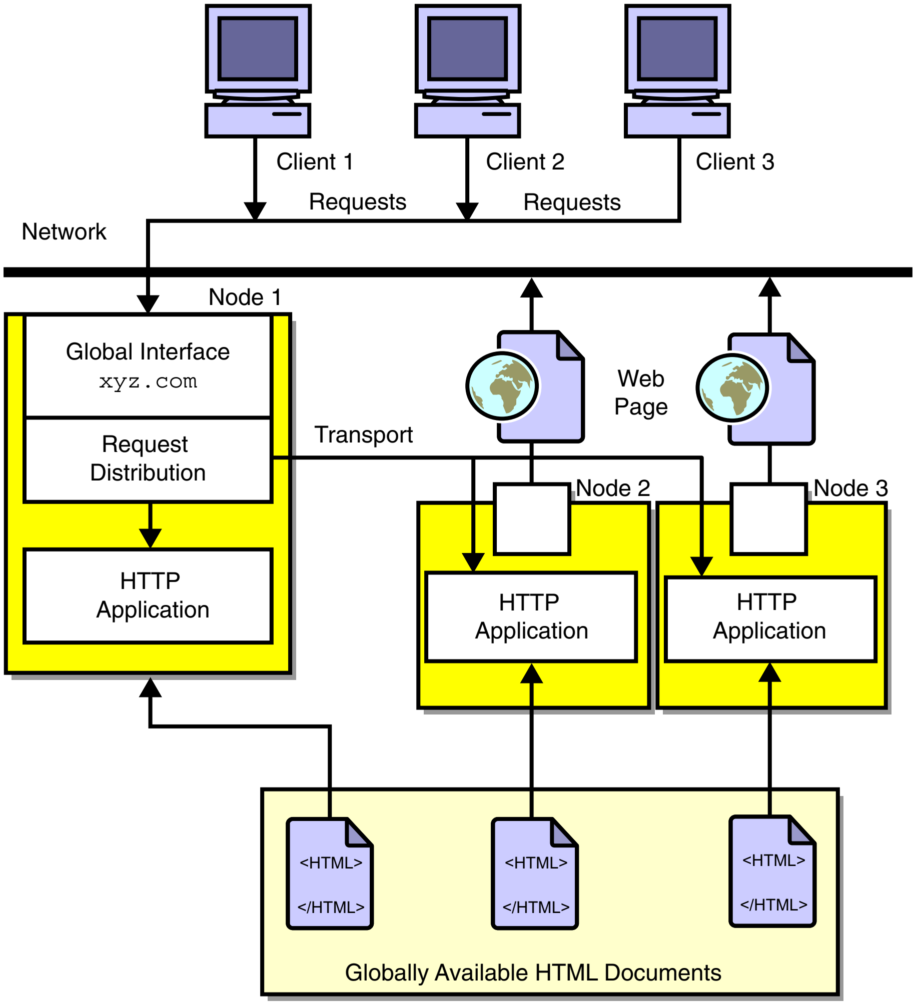


Figure 10-1 Scalable Services, Shared Addresses Architecture

Exploring Characteristics of Scalable Services

Like the failover services described in the previous modules, the applications used as scalable services in Sun Cluster 3.1 are generally off-the-shelf applications that are *not* specifically compiled or released to run in the cluster. The application binaries running on the various nodes are unaware of each other. All the “magic” that makes this work is in the Sun Cluster 3.1 `SharedAddress` mechanism.

File and Data Access

Like the failover services, all the data for the scalable service must be in the shared storage.

Unlike the failover service, a file system-oriented scalable service *must* use the global file system. A `SUNW.HAStoragePlus` resource can still be set up to manage dependencies between the service and the storage.

File Locking for Writing Data

One of the big barriers to taking “just any application” and turning it into a scalable service is that a service that is cluster-unaware may have been written in such a way as to ignore file locking issues.

In Sun Cluster 3.1 software, an application that does data modification without any type of locking or file synchronization mechanism generally cannot be used as a scalable service.

While Sun Cluster 3.1 software provides the global data access methods it does *not* automatically call any file-locking primitives for you.

Web servers that do file writing in common gateway interface (cgi) scripts although not written specifically for a scalable platform, are usually written in such a way that multiple instances can be launched simultaneously even on a standalone server. Thus, they already have the locking in place to make them ideal to work as scalable services in Sun Cluster 3.1 software.

Web servers that do file writing using Java servlets must be examined much more closely. A servlet might use thread synchronization rather than file locking to enforce serial write access to a critical resource. This will *not* translate properly into a scalable service.

Using the SharedAddress Resource

The “glue” that holds a scalable service together in Sun Cluster 3.1 software is the SharedAddress resource.

This resource provides not only a single IP address that makes the scalable service look like a “single server” from the point of view of the client, but also provides the load balancing of requests to all the nodes on which a Scalable Service is active.

Client Affinity

Certain types of applications require that load balancing in a scalable service be on a *per client* basis, rather than a *per connection* basis. That is, they require that the same client IP always have its requests forwarded to the same node. The prototypical example of an application requiring such *Client Affinity* is a shopping cart application, where the state of the client’s shopping cart is recorded only in the memory of the particular node where the cart was created.

A single SharedAddress resource can provide *standard* (per connection load balancing) and the type of *sticky* load balancing required by shopping carts. A scalable service’s agent “registers” which type of load balancing is required, based on a property of the data service.

Load-Balancing Weights

Sun Cluster 3.1 software also lets you control, on a scalable resource by scalable resource basis, the weighting that should be applied for load balancing. The default weighting is equal (connections or clients, depending on the stickiness) per node, but through the use of properties described in the following sections, a “better” node can be made to handle a greater percentage of requests.

Exploring Resource Groups for Scalable Services

A scalable service requires the creation of two resource groups. A failover resource group holds the SharedAddress resource. It is online, or *mastered*, by only one node at a time. The node that masters the SharedAddress resource is the only one which receives incoming packets for this address from the public network. A scalable resource group holds an HAStoragePlus resource and the actual service.

Remember, the HAStoragePlus resource is there to guarantee that the storage is accessible on *each node* before the service is started on that node. Figure 10-2 is a block diagram showing the relationship between scalable and failover resource groups.

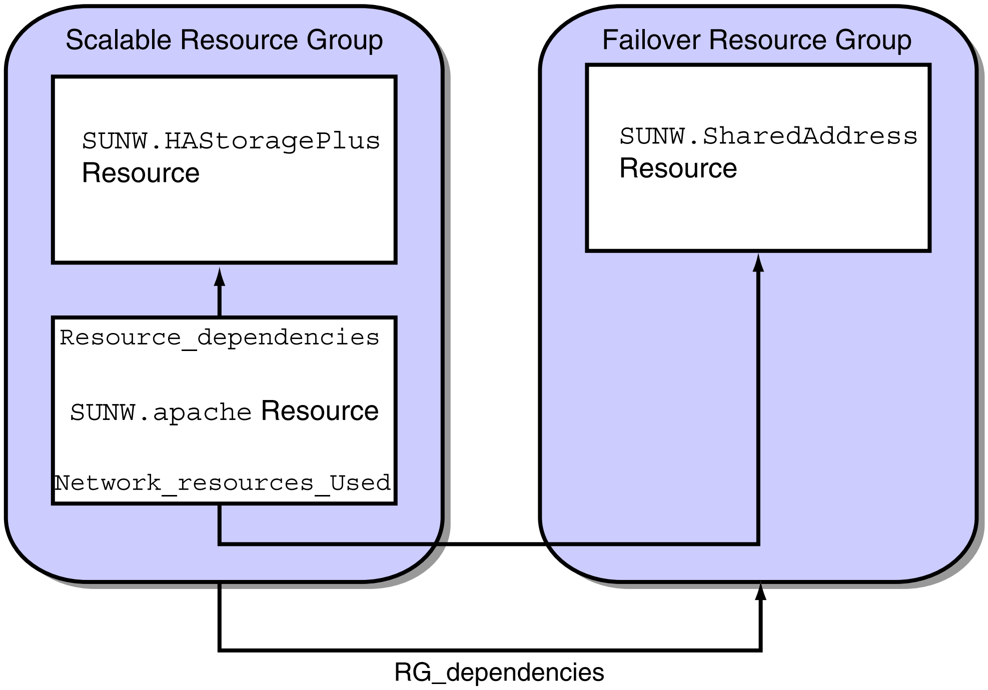


Figure 10-2 Scalable and Failover Resource Groups

Resources and Their Properties in the Resource Groups

Table 10-1 and Table 10-2 demonstrate the properties and contents of the two resource groups required to implement a scalable service.

- Resource Group Name: sa-rg
- Properties: `[Nodelist=vincent,theo Mode=Failover Failback=False...]`
- Resource Group Contents: See Table 10-1

Table 10-1 `sa-rg` Resource Group Contents

Resource Name	**Resource Type**	**Properties**

apache-lh	SUNW.SharedAddress	HostnameList=apache-lh
		Netiflist=therapy@vincent
		therapy@theo

- Resource Group Name: apache-rg
- Properties: `[Nodelist=vincent,theo Mode=Scalable Desired_primaries=2 Maximum_primaries=2]`

  `RG_dependencies=sa-rg`
- Resource Group Contents: See Table 10-2

Table 10-2 `apache-rg` Resource Group Contents

Resource Name	**Resource Type**	**Properties**

web-stor	SUNW.HAStoragePlus	FilesystemMountpoints=/global/web
		AffinityOn=False (ignored anyway)

apache-res	SUNW.apache	Bin_dir=/global/web/bin
		Load_balancing_policy=LB_WEIGHTED
		Load_balancing_weights=3@1,1@2
		Scalable=TRUE
		port_list=80/tcp
		Network_resources_used=apache-lh
		Resource_dependencies=web-stor

Understanding Properties for Scalable Groups and Services

There are certain resource group properties and certain resource properties of particular importance for scalable services.

The `Desired_primaries` and `Maximum_primaries` Properties

These are group properties that indicate how many nodes the service should run on. The `rgmd` will try to run the service on `Desired_primaries` nodes, but you can manually switch it on using `scswitch`, up to `Maximum_primaries`.

Note that if these values are greater than 1, the `Mode=Scalable` property is automatically set.

The `Load_balancing_policy` Property

This is a property of the data service resource. It has one of the following values:

- `Lb_weighted` – Client connections are all load-balanced. This is the default. Repeated connections from the same client might be serviced by different nodes.
- `Lb_sticky` – Connections from the same client IP to the same server port all go to the same node. Load balancing is only for different clients. This is only for all ports listed in the `Port_list` property.
- `Lb_sticky_wild` – Connections from the same client to *any* server port go to the same node. This is good when port numbers are generated dynamically and not known in advance.

The `Load_balancing_weights` Property

This property controls the weighting of the load balancer. Default is even for all nodes. A value such as `3@1,1@2` indicates three times as many connections (or clients, with one of the "sticky" policies) serviced by Node 1 in the cluster compared to the number serviced by Node 2.

The Network_resources_used Property

It is *required* to set this property on a scalable resource. Its value *must* point to the SharedAddress resource (in the other resource group). This is used by the data service agent to “register” the data service with the load balancer associated with the SharedAddress resource.

Adding Auxiliary Nodes for a SharedAddress Property

The SharedAddress logic includes a routine whereby the actual IP address associated with the resource is configured on the public network adapter (IPMP groups) on the primary node and is configured as a virtual address on the loopback network on all other nodes in its nodelist.

This enables scalable services that are running on nodes other than the primary SharedAddress node to still bind to the IP address used for the SharedAddress.

However, the SharedAddress resource *must* know about all possible nodes on which any scalable service associated with it might possibly run.

If the nodelist for the SharedAddress resource group is a superset of the Nodelist of every scalable service that might run on it, you are fine. But you may want to restrict which nodes might actually be the primary for the SharedAddress, while still allowing a larger nodelist for the scalable services dependent on it.

The SharedAddress resource has a special "auxiliary nodes" property which allows you to augment the Nodelist of its group, just for the purposes of supporting more nodes for scalable services. This is set with the -X option to scrgadm.

In the following example, you want only nodes vincent and theo to be the primaries for the SharedAddress (to actually host it on the public net and do the load balancing). But you may be supporting scalable services that run on vincent, theo, and apricot:

```
# scrgadm -a -g sa-rg -h vincent,theo
# scrgadm -a -S -l apache-lh -g sa-rg -X apricot
```

Reviewing scrgadm Command Examples for a Scalable Service

The following examples assume that the Apache Web Server agent was added from the Sun Cluster 3.1 Data Services CD using the scinstall or pkgadd commands.

1. Add (register) the resource types by typing:

```
# scrgadm -a -t SUNW.apache
# scrgadm -a -t SUNW.HAStoragePlus
```

2. Add the failover resource group for the SharedAddress by typing:

```
# scrgadm -a -g sa-rg -h vincent,theo
```

3. Add the SharedAddress resource by typing:

```
# scrgadm -a -S -l apache-lh -g sa-rg
```

4. Add the scalable resource group by typing:

```
# scrgadm -a -g apache-rg
   -y RG_dependencies=sa-rg
   -y Desired_primaries=2 -y Maximum_primaries=2
```

5. Add the HAStoragePlus resource by typing:

```
# scrgadm -a -j web-stor -t SUNW.HAStoragePlus
   -g apache-rg -x FilesystemMountpoints=/global/web
```

6. Add the Apache Service by typing:

```
# scrgadm -a -j apache-res -t SUNW.apache -g apache-rg
   -y Resource_dependencies=web-stor
   -x Bin_dir=/global/web/bin
   -x Port_list=80/tcp
   -y Network_resources_used=apache-lh
   -y Scalable=TRUE
```

Controlling Scalable Resources and Resource Groups Using the `scswitch` Command

The following examples demonstrate how to use the `scswitch` command to perform advanced operations on resource groups, resources, and data service fault monitors.

Resource Group Operations

Use the following commands to:

- Shut down a resource group:

```
# scswitch -F -g apache-rg
```

- Turn on a resource group:

```
# scswitch -Z -g apache-rg
```

- Switch the scalable resource group to these specific nodes (on other nodes it is turned off): If no `-h` option is given, it goes on the most preferred `Desired_primaries` number of nodes in the list, and off on other nodes:

```
# scswitch -z -g apache-rg [-h node,node,node]
```

- Restart a resource group:

```
# scswitch -R -h node,node -g apache-rg
```

- Evacuate all resources and resource groups from a node:

```
# scswitch -S -h node
```

Resource Operations

Use the following commands to:

- Disable a resource and its fault monitor:

```
# scswitch -n -j apache-res
```

- Enable a resource and its fault monitor:

```
# scswitch -e -j apache-res
```

- Clear the `STOP_FAILED` flag:

```
# scswitch -c -j apache-res -h node -f STOP_FAILED
```

Fault Monitor Operations

Use the following commands to:

- Disable the fault monitor for a resource:

```
# scswitch -n -M -j apache-res
```

- Enable a resource fault monitor:

```
# scswitch -e -M -j apache-res
```

Note – You should not manually stop any resource group operations that are underway. Operations, such as scswitch, must be allowed to complete.

Using the scstat Command for a Scalable Application

Use the scstat -g command and option as follows to show the state of resource groups and resources for a scalable application:

```
# scstat -g
-- Resource Groups and Resources --

            Group Name    Resources
            ----------    ---------
Resources:  sa-rg         apache-lh
Resources:  apache-rg     apache-res web-stor


-- Resource Groups --

            Group Name    Node Name     State
            ----------    ---------     -----
Group:      sa-rg         vincent       Offline
Group:      sa-rg         theo          Online

Group:      apache-rg     vincent       Online
Group:      apache-rg     theo          Online


-- Resources --

            Resource Name Node Name  State    Status Message
            ------------- ---------  -----    --------------
Resource:   apache-lh     vincent    Offline Offline
Resource:   apache-lh     theo       Online  Online -
SharedAddress online.

Resource:   web-stor      vincent    Online  Online
Resource:   web-stor      theo       Online  Online

Resource:   apache-res    vincent    Online  Online
Resource:   apache-res    theo       Online  Online -
Service is online.
```

Advanced Resource Group Relationships

Sun Cluster 3.1 offers, starting in Sun Cluster 3.1 9/04 (Update 3), a series of advanced resource group relationships called *resource group affinities*.

Resource group affinities provide a mechanism for specifying either a *preference* (weak affinities) or a *requirement* (strong affinities) that certain resource groups either run on the same node or do not run on the same node.

In this discussion, the words *source* and *target* are used to refer to the resource groups with an affinities relationships. The *source* is the group for which the value of `RG_affinities` is set, and the *target* is the group referred to by the value of the property. So, in the following example:

```
# scrgadm -c -g rg2 -y RG_affinities=++rg1
```

`rg2` is referred to as the *source* and `rg1` as the *target*.

Weak Positive Affinities and Weak Negative Affinities

The first two kinds of affinities place only a preference, not a requirement, on the location of the source group.

Setting a weak positive affinity says that the source group will *prefer* to switch to the node already running the target group, if any. If the target group is not running at all, that is fine. You can freely switch online and offline either group, and freely, explicitly, place either on any node that you want.

Setting a weak negative affinity just means the source group will prefer *any other node besides the target.* Once again, it is just a preference, you can freely, explicitly put either group on whatever node you want.

Weak positive and weak negative group affinities are denoted by a single plus or single minus sign, respectively.

In this example group `myrg2` is given a weak positive affinity for `myrg1`. Note that this does not affect the current location of either group.

```
pecan:/# scrgadm -c -g myrg2 -y RG_affinities=+myrg1
WARNING: resource group myrg2 declares a weak positive
affinity for resource group myrg1; but this affinity is not
satisfied by the current state of the two resource groups
```

The following will be affected by a weak affinity (if the target group is actually online):

- Failover of the source group
- `# scswitch -Z -g source-grp`
- `# scswitch -z -g source-grp`

For example, weak affinities have no effect on a command like:

```
# scswitch -z -g source-grp -h specific-node
```

There can be multiple resource groups as the value of the property. In other words a source can have more than one target. In addition, a source can have both weak positive and weak negative affinities. In these cases, the source prefers to choose a node satisfying the greatest possible number of weak affinities (for example, choose to pick a node that satisfies two weak positive affinities and two weak negative affinities rather than a node that satisfies 3 weak positive affinities and no weak negative affinities.

Strong Positive Affinities

Strong positive affinities (indicated with a ++ before the value of the target) place a *requirement* that the source run on the same node as the target. This example sets a strong positive affinity:

```
# scrgadm -c -g rg2 -y RG_affinities=++rg1
```

The following applies:

- The only node or nodes on which the source can be online are nodes on which the target is online.
- If the source and target are currently running on one node, and you switch the target to another node, it will *drag* the source with it.
- If you offline the target group, it will offline the source as well.
- An attempt to switch the source to a node where the target is not running will fail.
- If a resource in the source group fails the source group *still* cannot fail over to a node where the target is not running. (See the next section for the “solution” to this one.)

The source and target are closely tied together. If you have two failover resource groups with a strong positive affinity relationship, it *might* make sense to make them just one group. So why does strong positive affinity exist?

- The relationship can be between a failover group (source) and a scalable group (target). That is, you are saying the failover group must run on *some node* already running the scalable group.
- You might want to be able to offline the source group but leave the target group running, which "works" with this relationship.
- For some reason, you might not be able to put some resources in the same group (the resources might "check" and reject you if you try to put it in the same group as another resource). But you still want them all to be running on the same node or nodes.

Strong Positive Affinity With Failover Delegation

A slight variation on strong positive affinity is set with the +++ syntax:

```
# scrgadm -c -g rg2 -y RG_affinities=+++rg1
```

The only difference between the +++ and the ++ is that here (with +++), if a resource in the source group fails and its fault monitor suggests a failover, the failover *can* succeed. What happens is that the RGM will move the target group over to where the source wants to fail over to, and then the source gets dragged correctly.

Strong Negative Affinity

A strong negative affinity is set using the following syntax:

```
# scrgadm -c -g rg2 -y RG_affinities=--rg1
```

Here, the source target *cannot* run on the same node as the target. It will absolutely refuse to switch to any node where the target is running.

If you switch the target to the node where the source is running, it chases the source out of the way and the source switches to a different node if any. If there are no more nodes, the source switches off.

For example, if you have a two-node cluster with both the source and target groups online (on different nodes), and one node crashes (whichever it is), only the target group can remain running, since the source group absolutely, categorically refuses to run on the same node.

You need to be careful with strong negative affinities because obviously the high-availability of the source group can be compromised.

Exercise: Installing and Configuring Sun Cluster Scalable Service for Apache

In this exercise, you complete the following tasks:

- Task 1 – Preparing for Apache Data Service Configuration
- Task 2 – Configuring the Apache Environment
- Task 3 – Testing the Server on Each Node Before Configuring the Data Service Resources
- Task 4 – Registering and Configuring the Sun Cluster Apache Data Service
- Task 5 – Verifying Apache Web Server Access
- Task 6 – Observing Cluster Failures
- Task 7 – Configuring Advanced Resource Group Relationships

Preparation

The following tasks are explained in this section:

- Preparing for Sun Cluster HA for Apache registration and configuration
- Registering and configuring the Sun Cluster HA for Apache data service
- Verifying Apache Web Server access and scalable capability

Note – During this exercise, when you see italicized names, such as ***`IPaddress`***, ***`enclosure_name`***, ***`node1`***, or ***`clustername`*** embedded in a command string, substitute the names appropriate for your cluster.

Task 1 – Preparing for Apache Data Service Configuration

Perform the following steps on *each node* of the cluster:

1. Install the Sun Cluster Apache data service software package by running `scinstall` on each node. Use option 3.

```
# scinstall
```

2. Create an entry in `/etc/hosts` for the shared address you will be configuring with the Apache Web server.

```
IP_address      clustername-web
```

Note – In the RLDC, the vnchost already contains the appropriate entry for each cluster. Verify that the entry for your cluster exists on the vnchost, and use the same IP address to create the entry on your cluster nodes. In a non-RLDC environment create the `/etc/hosts` entry on your administrative workstation as well.

Task 2 – Configuring the Apache Environment

On (any) *one node* of the cluster, perform the following steps:

1. Make a resource-specific copy of the `/usr/apache2/bin/apachectl` script and edit it:

```
# mkdir /global/web/bin
# cp /usr/apache2/bin/apachectl /global/web/bin
# vi /global/web/bin/apachectl
```

a. Add a line to make an apache run time directory. This directory gets deleted every time you reboot. Letting this script create it will solve the problem. Just add the line in bold as the second line of the file:

```
#!/bin/sh
mkdir -p /var/run/apache2
```

b. Locate the line:

```
HTTPD='/usr/apache2/bin/httpd'
```

And change it to:

```
HTTPD='/usr/apache2/bin/httpd -f /global/web/conf/httpd.conf'
```

2. Copy the sample /etc/apache2/httpd.conf-example to /global/web/conf/httpd.conf.

```
# mkdir /global/web/conf
# cp /etc/apache2/httpd.conf-example /global/web/conf/httpd.conf
```

3. Edit the /global/web/conf/httpd.conf file, and change the following entries as shown in Table 10-3. The changes are shown in their order in the file, so you can search for the first place to change, change it, then search for the next, and so on

Table 10-3 Entries in the /global/web/conf/httpd.conf File

Old entry	New entry
KeepAlive On	KeepAlive Off
Listen 80	Listen *clustername*-web:80
ServerName 127.0.0.1	ServerName *clustername*-web
DocumentRoot "/var/apache2/htdocs"	DocumentRoot "/global/web/htdocs"
<Directory "/var/apache2/htdocs">	<Directory "/global/web/htdocs">
ScriptAlias /cgi-bin/ "/var/apache2/cgi-bin/" [one line]	ScriptAlias /cgi-bin/ "/global/web/cgi-bin/" [one line]
<Directory "/var/apache2/cgi-bin">	<Directory "/global/web/cgi-bin">

4. Create directories for the Hypertext Markup Language (HTML) and CGI files and populate with the sample files.

```
# cp -rp /var/apache2/htdocs /global/web
# cp -rp /var/apache2/cgi-bin /global/web
```

5. Copy the file called “test-apache.cgi” from the classroom server to /global/web/cgi-bin. You use this file to test the scalable service. Make sure that test-apache.cgi is executable by all users.

```
# chmod 755 /global/web/cgi-bin/test-apache.cgi
```

Task 3 – Testing the Server on Each Node Before Configuring the Data Service Resources

Perform the following steps. Repeat the steps on each cluster node (one at a time).

1. Temporarily configure the logical shared address (on one node).

```
# ifconfig pubnet_adapter addif clustername-web netmask + broadcast + up
```

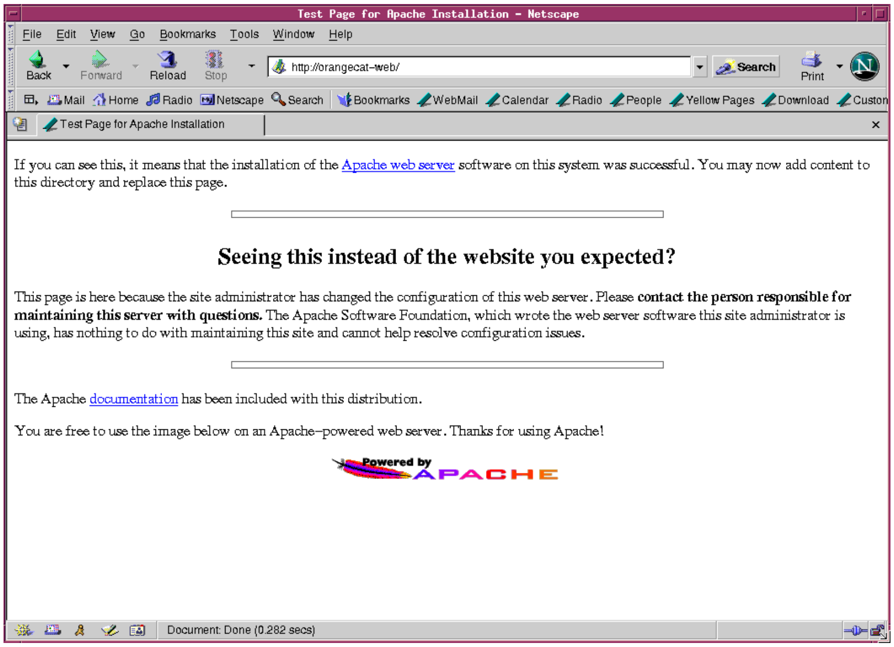
2. Start the server (on that node).

```
# /global/web/bin/apachectl start
```

3. Verify that the server is running.

```
vincent:/# ps -ef|grep apache2
webservd  4604  4601   0 10:20:05 ?         0:00 /usr/apache2/bin/httpd
-f /global/web/conf/httpd.conf -k start
webservd  4603  4601   0 10:20:05 ?         0:00 /usr/apache2/bin/httpd
-f /global/web/conf/httpd.conf -k start
webservd  4605  4601   0 10:20:05 ?         0:00 /usr/apache2/bin/httpd
-f /global/web/conf/httpd.conf -k start
    root  4601     1   0 10:20:04 ?         0:01 /usr/apache2/bin/httpd
-f /global/web/conf/httpd.conf -k start
webservd  4606  4601   0 10:20:05 ?         0:00 /usr/apache2/bin/httpd
-f /global/web/conf/httpd.conf -k start
webservd  4602  4601   0 10:20:05 ?         0:00 /usr/apache2/bin/httpd
-f /global/web/conf/httpd.conf -k start
```

4. Connect to the server from the Web browser on your administration or display station. Use `http://`*`clustername`*`-web` (Figure 10-3).


Test Page for Apache Installation - Netscape
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://orangecat-web/
Search Print
Mail Home Radio Netscape Search Bookmarks WebMail Calendar Radio People Yellow Pages Download Custom
Test Page for Apache Installation
Powered by APACHE
Document: Done (0.282 secs)

If you can see this, it means that the installation of the Apache web server software on this system was successful. You may now add content to this directory and replace this page.

Seeing this instead of the website you expected?

This page is here because the site administrator has changed the configuration of this web server. Please **contact the person responsible for maintaining this server with questions.** The Apache Software Foundation, which wrote the web server software this site administrator is using, has nothing to do with maintaining this site and cannot help resolve configuration issues.

The Apache documentation has been included with this distribution.

You are free to use the image below on an Apache-powered web server. Thanks for using Apache!

Figure 10-3 Apache Server Test Page

Note – If you can not connect, you may need to disable proxies or set a proxy exception in your web browser.

5. Stop the Apache Web server.

```
# /global/web/bin/apachectl stop
```

6. Verify that the server has stopped.

```
# ps -ef | grep apache2
root   8394   8393   0 17:11:14 pts/6     0:00 grep apache2
```

7. Take down the logical IP address.

```
# ifconfig pubnet_adapter removeif clustername-web
```

Task 4 – Registering and Configuring the Sun Cluster Apache Data Service

Perform the following steps only on (any) one node of the cluster:

1. Register the resource type required for the Apache data service.

   ```
   # scrgadm -a -t SUNW.apache
   ```

2. Create a failover resource group for the shared address resource. Use the appropriate node names for the `-h` argument. If you have more than two nodes, you can include more than two nodes after the `-h`.

   ```
   # scrgadm -a -g sa-rg -h node1,node2
   ```

3. Add the `SharedAddress` logical hostname resource to the resource group.

   ```
   # scrgadm -a -S -g sa-rg -l clustername-web
   ```

4. Create a scalable resource group to run on all nodes of the cluster. (The example assumes two nodes.)

```
# scrgadm -a -g web-rg -y Maximum_primaries=2 \
-y Desired_primaries=2 -y RG_dependencies=sa-rg
```

5. Create a storage resource in the scalable group.

```
# scrgadm -a -g web-rg -j web-stor -t SUNW.HAStoragePlus \
-x FilesystemMountpoints=/global/web -x AffinityOn=false
```

6. Create an application resource in the scalable resource group.

```
# scrgadm -a -j apache-res -g web-rg \
-t SUNW.apache -x Bin_dir=/global/web/bin \
-y Scalable=TRUE -y Network_resources_used=clustername-web \
-y Resource_dependencies=web-stor
```

7. Bring the failover resource group online.

```
# scswitch -Z -g sa-rg
```

8. Bring the scalable resource group online.

```
# scswitch -Z -g web-rg
```

9. Verify that the data service is online.

```
# scstat -g
```

Task 5 – Verifying Apache Web Server Access

Perform the following steps to verify the Apache Web Server access and scalable utility:

1. Connect to the web server using the browser on the administrator workstation using `http://`*`clustername`*`-web/cgi-bin/test-apache.cgi`.
2. Repeatedly press the Refresh or Reload button on the browser. The `test-apache.cgi` script shows the name of the cluster node that is currently servicing the request. The load-balancing is *not* done on a round-robin basis, so you may see several consecutive requests serviced by the same node. Over time, however, you should see the load-balancing be 50-50.

Task 6 – Observing Cluster Failures

Cause as many of the following failures (one at a time, and fix that one before going on to the next one) as you can, given your physical access to the cluster.

Perform the following steps to observe the behavior of the scalable service:

1. Fail a single public network interface on the node where the `SharedAddress` resource is online.
2. Fail all public network interfaces on that node.
3. Reboot one of the nodes running the scalable service.
4. Fail (or bring to the OK prompt) one of the nodes running the scalable service.

Task 7 – Configuring Advanced Resource Group Relationships

In this task, you will observe the effects of configuring variations of the RG_affinities property.

Perform the following steps:

1. If you have not yet done so, complete Task 7 in Module 9.
2. Put your `myapp-rg` resource group offline:
   ```
   # scswitch -F -g myapp-rg
   ```
3. Put a strong negative affinity with the apache resource group as the source and `myapp-rg` as the target. This means that apache will categorically refuse to run on any node where `myapp-rg` is running:
   ```
   # scrgadm -c -g web-rg -y RG_affinities=--myapp-rg
   ```
4. Switch the `myapp-rg` onto some node where apache is running. Observe what happens to apache. Observe the console messages as well.
   ```
   # scswitch -z -g myapp-rg -h somenode
   # scstat -g
   ```
5. Switch the `myapp-rg` another node where apache is running. Observe what happens to apache. Does it come back online on the first node?
   ```
   # scswitch -z -g myapp-rg -h othernode
   # scstat -g
   ```
6. Switch the `myapp-rg` offline. Can apache come back online on more nodes? Now remove the relationship:
   ```
   # scswitch -F -g myapp-rg
   # scstat -g
   # scrgadm -c -g web-rg -y RG_affinities=""
   ```
7. Set a weak positive affinity so that `myapp-rg` (source) always *prefers* to run on the same node as `nfs-rg` (target).
   ```
   # scrgadm -c -g myapp-rg -y RG_affinities=+nfs-rg
   ```
8. Print out the value of the `Nodelist` for `myapp-rg.`
   ```
   # scrgadm -pv -g myapp-rg |grep Nodelist
   ```
9. Switch your `nfs-rg` so that it is *not* on the preferred node of `myapp-rg`:
   ```
   # scswitch -z -g nfs-rg -h nonpreferrednode
   ```

10. Put myapp-rg online without specifying the node. Where does it end up? Why does it not end up on the first node of its own Nodelist?

    ```
    # scswitch -z -g myapp-rg
    # scstat -g
    ```

11. Switch the myapp-rg so it is no longer on the same node as nfs-rg. Can you do it? Is a weak affinity a preference or a requirement?

    ```
    # scswitch -z -g myapp-rg -h othernode
    # scstat -g
    ```

12. Switch the myapp-rg off and now change the affinity to a strong positive affinity. What is the difference between ++ and +++ (use +++)? The answer lies a few steps further on.

    ```
    # scswitch -F -g myapp-rg
    # scrgadm -c -g myapp-rg -y RG_affinities=+++nfs-rg
    ```

13. Put myapp-rg online without specifying the node. Where does it end up?

    ```
    # scswitch -z -g myapp-rg
    # scstat -g
    ```

14. Switch the myapp-rg so it is no longer on the same node as nfs-rg. Can you do it?

    ```
    # scswitch -z -g myapp-rg -h othernode
    # scstat -g
    ```

15. What happens if you switch the target group nfs-rg?

    ```
    # scswitch -z -g nfs-rg -h othernode
    # scstat -g
    ```

16. What happens if RGM wants to fail over the source myapp-rg because its fault monitor indicates application failure?

    Kill myappdaemon on whichever node it is running a few times (be patient with restarts) and observe.

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Module 11

Performing Supplemental Exercises for Sun Cluster 3.1 Software

Objectives

Upon completion of this module, you should be able to:

- Configure a failover zone in the Sun Cluster 3.1 environment on Solaris 10 OS
- Configure HA-ORACLE in a Sun Cluster 3.1 software environment as a failover application
- Configure ORACLE Real Application Cluster (RAC) 10g in a Sun Cluster 3.1 software environment

Relevance

Discussion – The following questions are relevant to understanding the content this module

- Why is it more difficult to install ORACLE software for a standard Oracle HA database on the local disks of each node rather than installing it in the shared storage?
- Is there any advantage to installing software on the local disks of each node?
- How is managing ORACLE RAC different that managing the other types of failover and scalable services presented in this course?

Additional Resources

Additional resources – The following references provide additional information on the topics described in this module:

- Sun Cluster 3.1 8/05 Software Collection for Solaris™ Operating System (x86 Platform Edition)
- Sun Cluster 3.1 8/05 Software Collection for Solaris Operating System (SPARC® Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (x86 Platform Edition)
- Sun Cluster 3.0-3.1 Hardware Collection for Solaris Operating System (SPARC® Platform Edition)

Exercise 1: Managing a Failover Solaris Zone Using the Sun Cluster Data Service for Solaris Containers

This exercise demonstrates creation and management of a failover zone in the Sun Cluster environment. The zone has a single IP address managed by a `SUNW.LogicalHostname` resource.

In this exercise, you complete the following tasks:

- Task 1 – Modifying `/etc/system` to Support Sparse Root Zones
- Task 2 – Creating a Failover File System That Will Be the Zonepath (VxVM)
- Task 3 – Creating a Failover File System That Will Be the Zonepath (SVM)
- Task 4 – Allocating a Failover IP Address for the Zone
- Task 5 – Creating a Failover Resource Group
- Task 6 – Configuring and Installing the Zone
- Task 7 – Copying the Zone Configuration to Other Nodes
- Task 8 – Booting the Zone and Performing System Configuration
- Task 9 – Testing the Zone on Other Nodes
- Task 10 – Adding the Data Service Agent
- Task 11 – Configuring the `sczbt` Zone Boot Resource Instance
- Task 12 – Copying the Zone Parameter File to Other Nodes
- Task 13 – Enabling and Testing the Zone Boot Resource

Preparation

Before continuing with this exercise, read the background information in this section.

Background – Failover Zones and Multiple Master Zones

The Sun Cluster 3.1 zones agent for Solaris 10 OS provides two different models for booting and controlling zones.

Failover zones have the following characteristics:

- The *zonepath* must be in shared storage configured as a failover file system.
- The zone is configured and installed manually only from one physical node, and then the zone's configuration is manually copied over to the other node.
- The control of the zone is managed by a failover resource group.
- The zone can have a combination of IP address types:
  - Some configured by the `zonecfg` utility and *not* controlled by the zone agent
  - Others configured as `SUNW.LogicalHostname` instances and controlled by the zone agent

  Both types of IP addresses will appear to fail over along with the zone from node to node. The advantage of `SUNW.LogicalHostname` addresses is that their presence will also *cause* the zone to fail over from node to node if all physical adapters in the IPMP group fail. The advantage of addresses configured by `zonecfg` is that they will be present if you need to boot the zone manually for debugging purposes; however if you have only this kind of address then your zone will *not* fail over even if all network adapters in the IPMP group on that node are broken.

Multiple-master zones have the following characteristics:

- The zones are created manually, separately, on each node.
- The zonepath for the zones must be storage local to each node.
- The zone name must be the same on each node but the zonepath may be the same or different.
- The zone is controlled by a scalable resource group, as described in detail in the following sections.
- IP addresses can be configured only by the `zonecfg` command and must be different on each physical node.
- The agent does *not* provide any internal load balancing or make any use of the global interface feature in any way.
- The agent boots the same zone name on all nodes simultaneously. The idea is that they may contain instances of some application that needs to be load-balanced externally.

Zone Boot (sczbt), Zone Script (sczsh), and Zone SMF (sczsmf) Resources

The agent provides three new kinds of resources. The word “resource types” is not being used here since each of these is implemented using the generic data service type, `SUNW.gds`. The three new resources are as follows:

- The `sczbt` resource provides booting and halting of the zone and, for failover zones, manages placing any (optional) IP addresses managed by `SUNW.LogicalHostname` resources into the zone. An instance of the `sczbt` resource is *required* in every resource group (both failover and multiple master) used to manage zones.
- The `sczsh` resource provides the ability to launch any software in the zone via a user-provided start script (that lives in the zone). Any instances of this resource are *optional*, and if present, must depend on the `sczbt` resource living in the same group.
- The `sczsmf` resource provides the ability to enable an SMF service in the zone. Note that this resource does *not* configure the SMF service, it only enables it. Any instances of this resource are *optional*, and if present, must depend on the `sczbt` resource living in the same group.

The `sczsh` and `sczsmf` resources are not required, and it is perfectly legal to just configure your zone manually to run all the boot scripts and SMF services that you want. However, by using the zone agent resources you gain the following benefits:

- You can have a fault-monitoring component. You can provide a custom fault probe for your resource; and its exit code can indicate a desire to have the entire resource group fail over (exit code 201) or restart (other non-zero exit code).
- You can have dependencies, even on resources that live in other resource groups that may be online on a different node.

Task 1 – Modifying `/etc/system` to Support Sparse Root Zones

Perform the following steps on all cluster nodes physically connected to the storage:

1. Edit `/etc/system` and remove the line:

```
exclude: lofs
```

Note – This line was inserted automatically as part of the cluster install, but it is incompatible with sparse root zones (the default when you create a zone), in which certain non-volatile parts of the global zones file tree are shared with a new local zone through a loopback file system mount.

2. Reboot for changes to take effect. In production, of course, you would do this one node at a time so that you don't jeopardize any highly-available applications that are already running.

```
# reboot
```

Task 2 – Creating a Failover File System That Will Be the Zonepath (VxVM)

Perform the following steps on *any one* node physically connected to the storage. Note that only step 6 is done on all nodes.

1. Select two disks from shared storage (one from one array and one from the other array) for a new disk group. Make sure you do not use any disks already in use in existing device groups. Note the logical device name (referred to as `c`*A*`t`*A*`d`*A* and `c`*B*`t`*B*`d`*B* in step 2). The following example checks against both VxVM and Solaris Volume Manager disks, just in case you are running both.

```
# vxdisk -o alldgs list
# metaset
# scdidadm -L
```

2. Create a disk group using these disks.

```
# /etc/vx/bin/vxdisksetup -i cAtAdA format=sliced
# /etc/vx/bin/vxdisksetup -i cBtBdB format=sliced
# vxdg init myzone-dg cds=off zone1=cAtAdA \
zone2=cBtBdB
```

3. Create a mirrored volume to hold the zone.

```
# vxassist -g myzone-dg make zonevol 1g layout=mirror
```

4. Register the new disk group (and its volume) with the cluster. The `nodelist` property contains all nodes physically connected to the storage.

```
# scconf -a -D \
type=vxvm,name=myzone-dg,nodelist=node1:node2
```

5. Create a file system.

```
# newfs /dev/vx/rdsk/myzone-dg/zonevol
```

6. Create a mount point and an entry in `/etc/vfstab` on *all* storage-connected nodes.

```
# mkdir /myzone
# vi /etc/vfstab
/dev/vx/dsk/myzone-dg/zonevol /dev/vx/rdsk/myzone-dg/zonevol /myzone ufs 2 no -
```

Task 3 – Creating a Failover File System That Will Be the Zonepath (SVM)

Perform the following steps on *any* node physically connected to the storage. Note that only step 5 is done on all nodes.

1. Select two disks from shared storage (one from one array and one from the other array) for a new disk group. Make sure you do not use any disks already in use in existing device groups. Note the DID device names (referred to as `dA` and `dB` in step 2). The following example checks against both VxVM and Solaris Volume Manager disks, just in case you are running both.

```
# vxdisk -o alldgs list
# metaset
# scdidadm -L
```

2. Create a diskset using these disks. In the first command, list nodes physically connected to the storage.

```
# metaset -s myzone-ds -a -h node1 node2
# metaset -s myzone-ds -a /dev/did/rdsk/dA
# metaset -s myzone-ds -a /dev/did/rdsk/dB
```

3. Create a soft partition (d100) of a mirrored volume hold the zone.

```
# metainit -s myzone-ds d11 1 1 /dev/did/rdsk/dAs0
# metainit -s myzone-ds d12 1 1 /dev/did/rdsk/dBs0
# metainit -s myzone-ds d10 -m d11
# metattach -s myzone-ds d10 d12
# metainit -s myzone-ds d100 -p d10 1g
```

4. Create a file system.

```
# newfs /dev/md/myzone-ds/rdsk/d100
```

5. Create a mount point and an entry in `/etc/vfstab` on *all* storage-connected nodes.

```
# mkdir /myzone
# vi /etc/vfstab
/dev/md/myzone-ds/dsk/d100 /dev/md/myzone-ds/rdsk/d100 /myzone ufs 2 no -
```

Task 4 – Allocating a Failover IP Address for the Zone

Perform the following on *all nodes* physically connected to storage.

Make an `/etc/hosts` entry for a new logical IP address for the zone. Things will work out most logically if you just give the IP the literal name of the zone:

```
# vi /etc/hosts
x.y.x.w    myzone
```

Task 5 – Creating a Failover Resource Group

Perform the following from *any one node* connected to the storage:

Create and enable failover resource group containing just the zone logical IP address and the failover storage. Note that the resource group will come online (and the storage will switch to) whichever node you list first in the following command:

```
# scrgadm -a -g myzone-rg -h node1,node2
# scrgadm -a -g myzone-rg -L -l myzone
# scrgadm -a -t HAStoragePlus
# scrgadm -a -g myzone-rg -t HAStoragePlus -j myzone-stor -x \
FilesystemMountpoints=/myzone
# scswitch -Z -g myzone-rg
```

Task 6 – Configuring and Installing the Zone

Perform the following steps from the *one node* where the new resource group is online.

Note – You *must not* set the `autoboot` parameter for the zone to `true`. The default is `false`, which is correct, and which is why you do not see it mentioned.

---

1. Configure the zone.

```
# zonecfg -z myzone
myzone: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:myzone> create
zonecfg:myzone> set zonepath=/myzone
zonecfg:myzone> commit
zonecfg:myzone> exit
```

2. Install the zone. You need to set the permissions of the zonepath directory to 700.

```
# chmod 700 /myzone
# zoneadm -z myzone install
Preparing to install zone <myzone>.
Creating list of files to copy from the global zone.
Copying <2826> files to the zone.
Initializing zone product registry.
Determining zone package initialization order.
Preparing to initialize <1005> packages on the zone.
Initialized <1005> packages on zone.
Zone <myzone> is initialized.
Installation of these packages generated warnings: <SUNWmconr>
The file </myzone/root/var/sadm/system/logs/install_log> contains a log
of the zone installation.
```

Task 7 – Copying the Zone Configuration to Other Nodes

Perform the following:

Copy the following files from the node where you created the zone to the other nodes physically connected to the storage:

- `/etc/zones/myzone.xml`
- `/etc/zones/index`

Note – In production you need to be careful about copying `/etc/zones/index`. It may have different entries on different nodes for zones not under control of the cluster agent, and in this case you should just copy the *entry* for your failover zone.

Task 8 – Booting the Zone and Performing System Configuration

On the node from which you installed the zone, perform the following steps:

1. Boot the zone.

```
# zoneadm -z myzone boot
```

2. Connect to the zone console and configure the zone. It will look similar to a standard Solaris OS that is booting after a `sys-unconfig`:

```
# zlogin -C myzone
[Connected to zone 'myzone' console]
```

Wait until the SMF services are all loaded, and navigate through the configuration screens. Get your terminal type correct, or you may have trouble with the rest of the configuration screens. When you have finished system configuration on the zone, it will reboot automatically. You can stay connected to the zone console.

3. Log in and perform other zone post-installation steps:

```
myzone console login: root
Password: ***

# vi /etc/default/login
[ Comment out the CONSOLE=/dev/console line]

# exit
```

4. Disconnect from the zone console using `~.`

Task 9 – Testing the Zone on Other Nodes

Perform the following steps to make sure you can boot the zone on other potential nodes:

1. On the node where the zone is running, halt the zone.

```
# zoneadm -z myzone halt
```

2. Switch the IP address and storage to another node.

```
# scswitch -z -g myzone-rg -h othernode
```

3. On the other node, verify that `/myzone` is mounted, and that you can boot and use the zone:

```
# df -k [verify filesystem switched here correctly]
# zoneadm -z myzone boot
# zlogin -C myzone [ disconnect with ~. when happy ]
# zoneadm -z myzone halt
```

4. Repeat steps 2 and 3 for any other nodes.

Task 10 – Adding the Data Service Agent

Perform the following on *all nodes* to add the agent:

```
# scinstall -i -d path-to-dataservices -s container
```

Task 11 – Configuring the sczbt Zone Boot Resource Instance

On *any one connected node*, perform the following steps:

1. Register the generic data service type:

   ```
   # scrgadm -a -t SUNW.gds
   ```

2. Put the correct values in the configuration file for the zone boot resource:

```
# cd /opt/SUNWsczone/sczbt/util
# vi sczbt_config
.
.
RS=myzone-rs
RG=myzone-rg
PARAMETERDIR=/etc/zoneagentparams
SC_NETWORK=true
SC_LH=myzone
FAILOVER=true
HAS_RS=myzone-stor

#
# The following variable will be placed in the parameter file
#
# Parameters for sczbt (Zone Boot)
#
# Zonename      Name of the zone
# Zonebootopt   Zone boot options ("-s" requires that Milestone=single-
user)
# Milestone     SMF Milestone which needs to be online before the zone is
considered as booted
#

Zonename=myzone
Zonebootopt=
Milestone=multi-user-server
```

3. Make an empty directory (the registration script will automatically create a parameter file in this directory), and run the registration script to create the new zone resource.

   ```
   # mkdir /etc/zoneagentparams
   # ./sczbt_register
   ```

Task 12 – Copying the Zone Parameter File to Other Nodes

Perform the following:

> Create a directory `/etc/zoneagentparams` on other potential nodes, and copy over the file `/etc/zoneagentparams/sczbt_myzone-rs` to these nodes.

Note – The file is automatically created on the node where you ran `sczbt_register`. It is possible to specify a globally mounted directory for this file; then you would not have to copy it. However, you are unlikely to create an entire global file system just for this purpose, since the zone storage itself for a failover zone must be in a failover file system.

Task 13 – Enabling and Testing the Zone Boot Resource

Enable the zone boot resource. This will also automatically move the logical `myzone` IP address into the zone.

Perform the following steps:

1. From any nodes, enable the zone boot resource.

```
# scswitch -e -j myzone-rs
```

2. From the node that owns the resource group, verify that the zone has booted.

```
# scstat -g
# zlogin -C myzone [exit by typing ~. when satsified ]
```

3. From the admin workstation, verify that you can access the zone using its logical IP address.

```
# telnet zone-logical-IP-address
login: root
Password: ****
# uname -a
# exit
```

4. From any node, switch the resource group to a different node:

```
# scswitch -z -g myzone-rg -h othernode
```

5. Repeat step 3 from the admin workstation to verify that your zone is now accessible. While it now lives on the other node, it looks just the same!

6. Generate hardware failures (public network failure, node stop) on the node currently running the zone and verify proper failover of the zone.

Exercise 2: Integrating ORACLE 10g Into Sun Cluster 3.1 Software as a Failover Application

In this exercise you integrate ORACLE 10g into Sun Cluster 3.1 software as a failover application.

In this exercise, you complete the following tasks:

- Task 1 – Creating a Logical IP Entry in the `/etc/hosts` File
- Task 2 – Creating `oracle` and `dba` Accounts
- Task 3 – Creating a Shared Storage File System for ORACLE Software (VxVM)
- Task 4 – Creating a Shared Storage File System for ORACLE Software (SVM)
- Task 5 – Preparing the `oracle` User Environment
- Task 6 – Disabling Access Control of X Server on the Admin Workstation
- Task 7– Running the `runInstaller` Installation Script
- Task 8– Preparing an ORACLE Instance for Cluster Integration
- Task 9 – Verifying ORACLE Software Runs on Other Nodes
- Task 10 – Registering the `SUNW.HAStoragePlus` Type
- Task 11 – Installing and Registering ORACLE Data Service
- Task 12 – Creating Resources and Resource Groups for ORACLE
- Task 13 – Verifying ORACLE Runs Properly in the Cluster

Preparation

Before continuing with this exercise, read the background information in this section.

Background

It is relatively straightforward to configure the ORACLE 10g Database software as a failover application in the Sun Cluster 3.1 environment. Like the majority of failover applications, the ORACLE software is “cluster-unaware.” You will be installing and configuring the software exactly as you would on a standalone system, and then manipulating the configuration so that it listens on a failover logical IP address.

Installing an Application on Local Storage (each node) or Shared Storage

For ease of installation and management, this lab takes the strategy of installing the ORACLE binaries themselves directly in shared storage. Thus, you only have to do the ORACLE installation once. The disadvantage is that there would be no way to do maintenance or patching on the software while keeping your application available on another node.

In production you might want to consider installing separate copies of the software locally on each node (with the data, of course, being in the shared storage). While it is more difficult to install, you get the advantage of being able to do "rolling maintenance" (including patching) of the software on one node while keeping your database alive on the other node.

Using a Failover File System or Global File System on Shared Storage

A failover file system is optimal for performance (although actual failover time is longer, the application runs faster once it is up and running).

However, you cannot use a failover file system if you want your application failover to a non-storage node.

This exercise uses a global filesystem because some lab environments may have non-storage nodes. You will notice that when we actually install Oracle we mount the file system with the `noglobal` flag to speed up the installation.

Task 1 – Creating a Logical IP Entry in the `/etc/hosts` File

Perform the following:

Create an entry for a logical IP address for ORACLE in the `/etc/hosts` file of all cluster nodes as follows:

```
# vi /etc/hosts
...
x.y.z.w    ora-lh
```

Task 2 – Creating oracle and dba Accounts

To create the oracle user and dba group accounts, perform the following steps on all cluster nodes:

1. Create the dba group account by typing:

   ```
   # groupadd -g 8888 dba
   ```

2. Create the oracle user. Note that you are specifying, but *not* creating, the home directory because it will live in shared storage that is not created yet.

```
# useradd -s /bin/ksh -g dba -u 8888 -d /global/oracle oracle
```

3. Create the oracle user password by typing:

   ```
   # passwd oracle
   New password: oracle
   Re-enter new Password: oracle
   ```

Task 3 – Creating a Shared Storage File System for ORACLE Software (VxVM)

Perform the following steps on *any* node physically connected to the storage. Note that only step 6 is done on all nodes.

1. Select two disks from shared storage (one from one array and one from the other array) for a new disk group. Make sure you do not use any disks already in use in existing device groups. Note the logical device name (referred to as c*A*t*A*d*A* and c*B*t*B*d*B* in step 2). The following example checks against both VxVM and Solaris Volume Manager disks, just in case you are running both.

```
# vxdisk -o alldgs list
# metaset
# scdidadm -L
```

2. Create a disk group using these disks.

```
# /etc/vx/bin/vxdisksetup -i cAtAdA format=sliced
# /etc/vx/bin/vxdisksetup -i cBtBdB format=sliced
# vxdg init ora-dg cds=off ora1=cAtAdA \
ora2=cBtBdB
```

3. Create a mirrored volume to hold the Oracle binaries and data.

```
# vxassist -g ora-dg make oravol 3g layout=mirror
```

4. Register the new disk group (and its volume) with the cluster. The `nodelist` property contains all nodes physically connected to the storage.

```
# scconf -a -D \
type=vxvm,name=ora-dg,nodelist=node1:node2
```

5. Create a UFS file system on the volume by typing:

```
# newfs /dev/vx/rdsk/ora-dg/oravol
```

6. Create a mount point and an entry in `/etc/vfstab` on *all* nodes:

```
# mkdir /global/oracle
# vi /etc/vfstab
/dev/vx/dsk/ora-dg/oravol /dev/vx/rdsk/ora-dg/oravol /global/oracle ufs 2 yes global
```

7. On any cluster node, mount the file system by typing:

```
# mount /global/oracle
```

8. From any cluster node, modify the owner and group associated with the newly mounted global file system by typing:

```
# chown oracle:dba /global/oracle
```

Task 4 – Creating a Shared Storage File System for ORACLE Software (SVM)

Perform the following steps on *any* node physically connected to the storage. Note that only step 5 is done on all nodes.

1. Select two disks from shared storage (one from one array and one from the other array) for a new diskset. Make sure you do not use any disks already in use in existing device groups. Note the DID device names (referred to as `dA` and `dB` in step 2). The following example checks against both VxVM and Solaris Volume Manager disks, just in case you are running both.

   ```
   # vxdisk -o alldgs list
   # metaset
   # scdidadm -L
   ```

2. Create a diskset using these disks.

   ```
   # metaset -s ora-ds -a -h node1 node2
   # metaset -s ora-ds -a /dev/did/rdsk/dA
   # metaset -s ora-ds -a /dev/did/rdsk/dB
   ```

3. Create a soft partition (d100) of a mirrored volume for Oracle.

   ```
   # metainit -s ora-ds d11 1 1 /dev/did/rdsk/dAs0
   # metainit -s ora-ds d12 1 1 /dev/did/rdsk/dBs0
   # metainit -s ora-ds d10 -m d11
   # metattach -s ora-ds d10 d12
   # metainit -s ora-ds d100 -p d10 3g
   ```

4. Create a file system:

   ```
   # newfs /dev/md/ora-ds/rdsk/d100
   ```

5. Create a mount point and an entry in `/etc/vfstab` on *all* nodes:

```
# mkdir /global/oracle
# vi /etc/vfstab
/dev/md/ora-ds/dsk/d100 /dev/md/ora-ds/rdsk/d100 /global/oracle ufs 2 yes global
```

6. On any cluster node, mount the file system by typing:

   ```
   # mount /global/oracle
   ```

7. From any cluster node, modify the owner and group associated with the newly mounted global file system by typing:

8. `# chown oracle:dba /global/oracle`

Task 5 – Preparing the `oracle` User Environment

To configure environment variables required for the ORACLE software, run the following commands from the cluster node on which the ORACLE software installation will be performed:

1. Switch the user to the `oracle` user by typing:

   ```
   # su - oracle
   ```

2. Edit the `.profile` file as shown. Be sure to substitute the proper value for your DISPLAY variable.

   ```
   $ vi .profile
   ORACLE_BASE=/global/oracle
   ORACLE_HOME=$ORACLE_BASE/product/10.1.0/db_1
   ORACLE_SID=MYORA
   PATH=$PATH:$ORACLE_HOME/bin
   DISPLAY=display-name-or-IP:#
   export ORACLE_BASE ORACLE_HOME ORACLE_SID PATH DISPLAY

   $ exit
   ```

Task 6 – Disabling Access Control of X Server on the Admin Workstation

Perform the following:

To allow client GUIs to be displayed, run the following command on the admin workstation:

```
(# or $) /usr/openwin/bin/xhost +
```

Task 7– Running the `runInstaller` Installation Script

To install the ORACLE software on the local file system, perform the following steps on the node that has been selected as the one to run the ORACLE installation program.

1. Unmount the `/global/oracle` directory as a global file system by typing:

   ```
   # umount /global/oracle
   ```

2. Mount the /global/oracle directory as a non-global file system—local to the node from which the ORACLE installation is to be performed (that is, without the global option so that the installation is expedited)—by typing:

```
# mount -o noglobal,logging /global/oracle
```

3. Switch the user to the oracle user by typing:

```
# su - oracle
```

4. Change directory to the location of the ORACLE software by typing:

```
$ cd ORACLE-software-location/Disk1
```

5. Run the runInstaller script by typing:

```
$ ./runInstaller -ignoreSysPrereqs
```

6. Respond to the dialogs using Table 11-1.

Table 11-1 The runInstaller Script Dialog Answers

Dialog	Action
Welcome	Click Next.
Specify Inventory Directory and Credentials	Verify and click OK.
Oracle Universal Installer orainstRoot.sh script	Open a terminal window as user root on the installing cluster node, change to the /global/oracle/oraInventory directory and run the ./orainstRoot.sh script. When the script finishes, click Continue.
Specify File Locations	Verify source and destination, and click Next.
Select Installation Type	Select the Custom radio button, and click Next.
Product Specific Prerequisite Checks	The security parameter check will not execute (because it is not "at one" with our Solaris 10 OS). Click Next.
Warning	Ignore the warning about the missing package. Click Continue.
Warning	Ignore the warning about kernel parameters (they are dynamically sized in Solaris 10). Click OK

Table 11-1 The `runInstaller` Script Dialog Answers (Continued)

Dialog	**Action**
Available Product Components	Deselect the following (so our install goes faster): ● Enterprise Edition Options ● Oracle Enterprise Manager 10.1.0.2 Database Control ● Oracle Development Kit 10.1.0.2 ● Oracle Transparent Gateways 10.1.0.2 Click Next.
Privileged Operating System Groups	Verify that both say `dba`, and click Next.
Create Database	Verify that the Yes radio button is selected, and click Next.
Summary	Verify, and click Install.

Script (setup privileges)	Open a terminal window as user root on the installing node and run the `root.sh` script (use default values when prompted).
	The script installs a new entry in `/etc/inittab` that launches a daemon called `cssd`. This entry is *not* appropriate for an installation of Oracle on the shared storage.
	On the same node you ran the script, edit `/etc/inittab` and remove the line it added (last line). Then type:
	`# init q`
	Verify that the daemon is no longer running.
	`# ps -ef|grep cssd`
	Click OK in the script prompt window.

Oracle Net Configuration Assistant Welcome	Verify that the Perform typical configuration check box is not selected, and click Next.
Listener Configuration, Listener Name	Verify the Listener name is `LISTENER`, and click Next.

Table 11-1 The `runInstaller` Script Dialog Answers (Continued)

Dialog	Action
Select Protocols	Verify that `TCP` is among the Selected Protocols, and click Next.
TCP/IP Protocol Configuration	Verify that the Use the standard port number of 1521 radio button is selected, and click Next.
More Listeners	Verify that the No radio button is selected, and click Next.
Listener Configuration Done	Click Next.
Naming Methods Configuration	Verify that the No, I do not want to configure additional naming methods configured radio button is selected, and click Next.
Done	Click Finish.
DBCA Step 1: Database Templates	Select the General Purpose radio button, and click Next.
Step 2: Database Identification	Type **`MYORA`** in the Global Database Name text field, observe that it is echoed in the SID text field, and click Next.
Step 3: Management Options	Verify this is not available (because you deselected the software option earlier) and Click Next.
Step 4: Database Credentials	Verify that the Use the Same Password for All Accounts radio button is selected. Enter `cangetin` as the password, and click Next.
Step 5: Storage Options	Verify that File System is selected, and click Next.
Step 6: Database File Locations	Verify that Use Common Location... is selected, and verify the path. Click Next.
Step 7: Recovery Configuration	Uncheck all the boxes, and click Next.
Step 8: Database Content	Uncheck Sample Schemas, and click Next.

Table 11-1 The `runInstaller` Script Dialog Answers (Continued)

Dialog	Action
Step 9: Initialization Parameters	On the Memory tab, select the Typical radio button. Change the Percentage to a ridiculously small number (1%). Click Next and accept the error telling you the minimum memory required. The percentage will automatically be changed on your form. Click Next.
Step 10: Database Storage	Click Next.
Step 11: Create Options	Verify that Create Database is selected, and click Finish.
Confirmation	Verify summary of database creation options, parameters, character sets, and data files, and click OK. **Note –** This might be a good time to take a break. The database configuration assistant will take 15 to 20 minutes to complete.
Database Configuration Assistant	Click Exit.
End of Installation	Click Exit and confirm.

Task 8– Preparing an ORACLE Instance for Cluster Integration

On the same node from which the ORACLE software was installed, perform the following steps as indicated to prepare the ORACLE instance for Sun Cluster 3.*x* software integration:

1. Configure an ORACLE user for the fault monitor by typing (as user `oracle`):

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> create user sc_fm identified by sc_fm;
SQL> grant create session, create table to sc_fm;
SQL> grant select on v_$sysstat to sc_fm;
SQL> alter user sc_fm default tablespace users quota 1m on users;
SQL> quit
```

2. Test the new ORACLE user account used by the fault monitor by typing (as user oracle):

   ```
   $ sqlplus sc_fm/sc_fm
   SQL> select * from sys.v_$sysstat;
   SQL> quit
   ```

3. Create a sample table by typing (as user oracle):

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> create table mytable (mykey VARCHAR2(10), myval NUMBER(10));
SQL> insert into mytable values ('off', 0);
SQL> insert into mytable values ('on', 1);
SQL> commit;
SQL> select * from mytable;
SQL> quit
```

Note – This sample table is created only for a *quick* verification that the database is running properly.

4. Shut down the ORACLE instance by typing:

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> shutdown
SQL> quit
```

5. Stop the ORACLE listener by typing:

```
$ lsnrctl stop
```

6. Configure the ORACLE listener by typing (as user `oracle`):

```
$ vi $ORACLE_HOME/network/admin/listener.ora
```

Modify the `HOST` variable to match logical host name `ora-lh`.

Add the following lines between the second-to-last and last right parentheses of the `SID_LIST_LISTENER` information:

```
(SID_DESC =
      (SID_NAME = MYORA)
      (ORACLE_HOME = /global/oracle/product/10.1.0/db_1)
      (GLOBALDBNAME = MYORA)
)
```

Your entire file should end up looking identical to the following, assuming your logical host name is literally `ora-lh`:

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /global/oracle/product/10.1.0/db_1)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (SID_NAME = MYORA)
      (ORACLE_HOME = /global/oracle/product/10.1.0/db_1)
      (GLOBALDBNAME = MYORA)
    )
  )

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = ora-lh)(PORT = 1521))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
      )
    )
  )
```

7. Configure the `tnsnames.ora` file by typing (as user `oracle`):

   ```
   $ vi $ORACLE_HOME/network/admin/tnsnames.ora
   ```

   Modify the `HOST` variables to match logical host name `ora-lh`.

8. Rename the parameter file by typing (as user `oracle`):

   ```
   $ cd /global/oracle/admin/MYORA/pfile
   $ mv init.ora.* initMYORA.ora
   ```

9. Unmount the `/global/oracle` file system and remount it using the global option (as user `root`) by typing:

   ```
   # umount /global/oracle
   # mount /global/oracle
   ```

Task 9 – Verifying ORACLE Software Runs on Other Nodes

Start the ORACLE software on the second node by performing the following steps:

1. Configure a virtual interface with the `ora-lh` IP address by typing:

```
# ifconfig pubnet-adapter addif ora-lh netmask + broadcast + up
```

2. Switch the user to `oracle` user by typing:

   ```
   # su - oracle
   ```

3. Start up the ORACLE instance by typing:

   ```
   $ sqlplus /nolog
   SQL> connect / as sysdba
   SQL> startup
   SQL> quit
   ```

4. Start the ORACLE listener by typing:

   ```
   $ lsnrctl start
   ```

5. Query the database and then shut it down by typing:

```
$ sqlplus /nolog
SQL> connect / as sysdba
SQL> select * from mytable;
SQL> shutdown
SQL> quit
```

6. Stop the ORACLE listener by typing:

```
$ lsnrctl stop
```

7. Remove the virtual interface for `ora-lh` (as user `root`) by typing:

```
# ifconfig pubnet-adapter removeif ora-lh
```

8. Repeat Steps 1–7 on all remaining cluster nodes.

Task 10 – Registering the `SUNW.HAStoragePlus` Type

Perform the following:

If necessary (if not done in a previous exercise), register the `SUNW.HAStoragePlus` type from any one node in the cluster:

```
# scrgadm -a -t SUNW.HAStoragePlus
```

Task 11 – Installing and Registering ORACLE Data Service

Perform the following steps as indicated to install and register the ORACLE data service:

1. Install the `SUNW.oracle_server` and `SUNW.oracle_listener` resource types on *all nodes* by typing:

```
# scinstall -i -d path-to-dataservices -s oracle
```

2. Register the `SUNW.oracle_server` and `SUNW.oracle_listener` resource types from *one cluster node* by typing:

```
# scrgadm -a -t SUNW.oracle_server
# scrgadm -a -t SUNW.oracle_listener
```

Task 12 – Creating Resources and Resource Groups for ORACLE

Perform the following steps from one cluster node to create the resource group and resources necessary for the ORACLE data service:

1. Create an empty resource group by typing:

```
# scrgadm -a -g ora-rg -h node1,node2[,node3]
```

2. Create a LogicalHostname resource by typing:

```
# scrgadm -a -L -g ora-rg -l ora-lh
```

3. Create an HAStoragePlus resource by typing:

```
# scrgadm -a -j ora-stor -g ora-rg -t HAStoragePlus \
-x FilesystemMountpoints=/global/oracle -x AffinityOn=true
```

4. Create an oracle_server resource by creating and running a script – (the command is too long to type interactively in some shells):

   a. Write a script containing the scrgadm command by typing:

```
# vi /var/tmp/setup-ora-res.sh
#!/bin/sh
/usr/cluster/bin/scrgadm -a -j ora-server-res -g ora-rg \
-t oracle_server \
-y Resource_dependencies=ora-stor \
-x Oracle_sid=MYORA -x Oracle_home=/global/oracle/product/10.1.0/db_1 \
-x Alert_log_file=/global/oracle/admin/MYORA/bdump/alert_MYORA.log \
-x Parameter_file=/global/oracle/admin/MYORA/pfile/initMYORA.ora \
-x Connect_string=sc_fm/sc_fm
```

   b. Run the script by typing:

```
# sh /var/tmp/setup-ora-res.sh
```

5. Create an oracle_listener resource by typing:

```
# scrgadm -a -j ora-listener-res -g ora-rg -t oracle_listener \
-x Oracle_home=/global/oracle/product/10.1.0/db_1 \
-x Listener_name=LISTENER -y Resource_dependencies=ora-stor
```

6. Bring the resource group online by typing:

```
# scswitch -Z -g ora-rg
```

Task 13 – Verifying ORACLE Runs Properly in the Cluster

Perform the following steps as indicated to verify the ORACLE software properly runs in the cluster:

1. Switch user to `oracle` user on the cluster node *not* currently running the `ora-rg` resource group by typing:

   ```
   # su - oracle
   ```

2. Query the database from a cluster node *not* currently running the `ora-rg` resource group by typing:

   ```
   $ sqlplus SYS/cangetin@MYORA as sysdba
   SQL> select * from mytable;
   SQL> quit
   ```

3. As user `root` (from any cluster node), switch the resource group to another node by typing:

   ```
   # scswitch -z -g ora-rg -h other-node
   ```

4. Query the database from a cluster node *not* currently running the `ora-rg` resource group (as user `oracle`) by typing:

   ```
   $ sqlplus SYS/cangetin@MYORA as sysdba
   SQL> select * from mytable;
   SQL> quit
   ```

5. Verify that the database fails over properly under various failure conditions:

   a. Complete failure of the node on which the resource group is running (bring it to the OK prompt).

   b. Complete public network failure on the node on which the resource group is running.

Exercise 3: Running ORACLE 10g RAC in Sun Cluster 3.1 Software

In this exercise you run ORACLE 10g RAC in Sun Cluster 3.1 software.

In this exercise, you complete the following tasks:

- Task 1 – Creating User and Group Accounts
- Task 2A (if using VxVM) – Installing RAC Framework Packages for ORACLE RAC With VxVM Cluster Volume Manager
- Task 2B (if using Solaris VM) – Installing RAC Framework Packages for ORACLE RAC With SVM Multiowner Disksets
- Task 3 – Installing ORACLE Distributed Lock Manager
- Task 4 – Creating and Enabling the RAC Framework Resource Group
- Task 5A – Creating Raw Volumes (VxVM)
- Task 5B – Creating Raw Volumes (Solaris Volume Manager)
- Task 6 – Configuring the `oracle` User Environment
- Task 7A – Creating the `dbca_raw_config` File (VxVM)
- Task 7B – Creating the `dbca_raw_config` File (Solaris VM)
- Task 8 – Disabling Access Control on X Server of the Admin Workstation
- Task 9 – Installing ORACLE CRS Software
- Task 10 – Upgrading ORACLE CRS Software
- Task 11 – Installing Oracle Database Software
- Task 12 – Upgrading ORACLE Database Software
- Task 13 – Configuring Oracle Virtual IPs
- Task 14 – Running the `vipca` Utility
- Task 15 – Configuring an ORACLE Listener
- Task 16 – Creating an ORACLE Database
- Task 17 – Verifying That ORACLE RAC Works Properly in a Cluster

Preparation

Before continuing with this exercise, read the background information in this section.

Background

ORACLE 10g RAC Software in the Sun Cluster environment encompasses several layers of software as follows:

- RAC Framework

  This layer sits just above the Sun Cluster framework. It encompasses the UNIX distributed lock manager (`udlm`) and a RAC-specific cluster membership monitor (`cmm`). In Solaris 10, it is *required* to create a resource group `rac-framework-rg` to control this layer (in Solaris 8 and 9, it is optional).

- ORACLE Cluster Ready Services (CRS)

  CRS is essentially ORACLE's own implementation of a resource group manager. That is, for ORACLE RAC Database instances and their associated listeners and related resources, CRS *takes the place* of our Sun Cluster resource group manager.

- ORACLE Database

  The actual ORACLE RAC database instances run on top of CRS. The database software must be installed separately (it is a different product) once CRS is installed and enabled. The database product has hooks that recognize that it is being installed in a CRS environment.

The various RAC layers are illustrated in figure Figure 11-1.

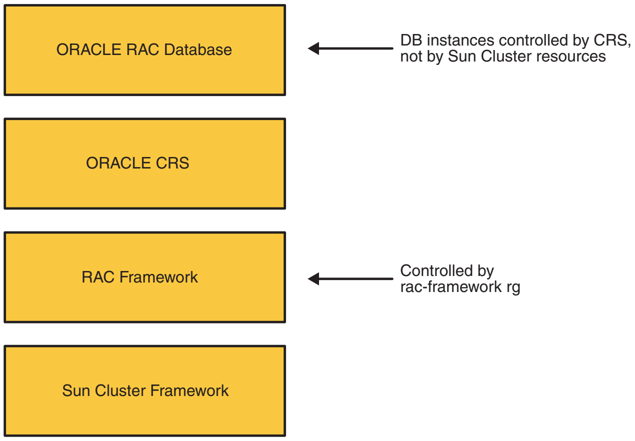

ORACLE RAC Database
DB instances controlled by CRS, not by Sun Cluster resources
ORACLE CRS
RAC Framework
Controlled by rac-framework rg
Sun Cluster Framework

Figure 11-1 ORACLE RAC Software Layers

RAC Database Storage

In the Sun Cluster environment, you have the following choices of where to store your actual data for RAC databases:

- Raw devices using VxVM Cluster Volume Manager (CVM) feature
- Raw devices using Solaris VM Multiowner diskset feature
- Raw devices using no volume manager (assumes hardware RAID)
- Shared QFS file system
- On a supported Network Attached Storage (NAS) device

  Starting in Sun Cluster 3.1 8/05, the only such supported device is a clustered Network Appliance Filer.

Note – Use of global devices (using normal device groups) or a global file system is *not supported*. The rationale is that, if you used global devices or global file system, your cluster transport would now be used for both the application-specific RAC traffic and for the underlying device traffic. The performance detriment this might cause might eliminate the advantages of using RAC in the first place.

In this lab you will be able to choose either of the first two choices. The variations in the lab are indicated by tasks that have an A or B suffix. You will always choose only one such task. Once you have built your storage devices, there are very few variations in the tasks that actually install and test the ORACLE software.

This lab does not actually address careful planning of your raw device storage. In the lab, you create all the volumes using the same pair of disks.

ORACLE 10.1.0.2 and 10.1.0.4

You will have to run a minimum of both Oracle CRS version 10.1.0.4 and Oracle Database version 10.1.0.4 on Sun Cluster 3.1 on the Solaris 10 OS. Unfortunately, version 10.1.0.4 is released only as a patch to version 10.1.0.2.

Therefore, in this lab, for *both* the Oracle CRS and Oracle Database products, you will have to initially install the 10.1.0.2 version and then use the 10.1.0.4 patch to upgrade.

The same patch release covers both the CRS and Database products, but you will have to apply the patch separately to each product.

Select the nodes on which the ORACLE 10g RAC software will be installed and configured. Note that the nodes you select must all be attached to shared storage.

Task 1 – Creating User and Group Accounts

Perform the following steps on *all* selected cluster nodes:

Note – If you have already done the HA-ORACLE lab, then delete the `oracle` user and `dba` groups from all nodes before proceeding.

1. Create the `dba` group account by typing:

```
# groupadd -g 7777 dba
```

2. Create the `oracle` user account by typing:

```
# useradd -g dba -u 7777 -d /oracle -m -s /bin/ksh oracle
```

3. Provide a password for the `oracle` user account by typing:

```
# passwd oracle
New Password: oracle
Re-enter new Password: oracle
```

Task 2A (if using VxVM) – Installing RAC Framework Packages for ORACLE RAC With VxVM Cluster Volume Manager

Perform the following steps on *all* selected cluster nodes

1. Install the appropriate packages from the data service agents CD:

```
# cd sc31-dataservices-location/components
# cd SunCluster_Oracle_RAC_FRAMEWORK_3.1/Solaris_10/Packages
# pkgadd -d . SUNWscucm SUNWudlm SUNWudlmr
# cd ../../../SunCluster_Oracle_RAC_CVM_3.1/Solaris_10/Packages
# pkgadd -d . SUNWcvm SUNWcvmr
```

2. Enter a license for the CVM feature of VxVM. Your instructor will tell you from where you can paste a CVM license.

```
# vxlicinst
```

3. Reboot the node. In production, you would always do this one node at a time so that existing clustered services remain highly available

```
# reboot
```

Task 2B (if using Solaris VM) – Installing RAC Framework Packages for ORACLE RAC With SVM Multiowner Disksets

Perform the following steps on *all* selected cluster nodes. Note that for Solaris VM installations, no reboot is required.

1. Install the appropriate packages from the data service agents CD:

```
# cd sc31-dataservices-location/components
# cd SunCluster_Oracle_RAC_FRAMEWORK_3.1/Solaris_10/Packages
# pkgadd -d . SUNWscucm SUNWudlm SUNWudlmr
# cd ../../../SunCluster_Oracle_RAC_SVM_3.1/Solaris_10/Packages
# pkgadd -d . SUNWscmd
```

2. List out local metadbs:

```
# metadb
```

3. Add metadbs on the root drive, only if they do not yet exist:

```
# metadb -a -f -c 3 c#t#d#s7
```

Task 3 – Installing ORACLE Distributed Lock Manager

Perform the following steps on *all* selected cluster nodes as user root:-

1. Install the ORCLudlm package by typing:

```
# cd course-sofwtare-location
# pkgadd -d . ORCLudlm
```

2. The pkgadd command will prompt you for the group that is to be DBA for the database. Respond by typing:

```
Please enter the group which should be able to act as
the DBA of the database (dba): [?] dba
```

Task 4 – Creating and Enabling the RAC Framework Resource Group

On *any one* of the selected nodes, perform the following steps to create and enable the RAC framework resource group:

1. Enter scsetup.

```
# scsetup
```

2. Choose Option #3 "Data Services."

3. Choose Option #1 “Sun Cluster support for Oracle RAC.”
4. Confirm you want to continue.
5. Choose Option #1 “Create the RAC framework resource group.”
6. If all of your cluster nodes are connected to the shared storage, confirm that you want to configure all nodes for the resource group.

   If all your nodes are *not* connected to shared storage, say “no” and choose specifically the RAC selected nodes.
7. Confirm that you want to create the RAC framework resource group.
8. Quit out of `scsetup`.
9. Enable the RAC Framework Resource Group:

   ```
   # scswitch -Z -g rac-framework-rg
   ```

Task 5A – Creating Raw Volumes (VxVM)

To create the raw volumes for the ORACLE 10g RAC Database, perform the following steps from the single node that is the *CVM Master*.

1. Determine which node is the CVM Master by running the following command on all nodes:

   ```
   # vxdctl -c mode
   ```
2. Select two disks from shared storage (one from one array and one from the other array) for a new disk group. Make sure you do not use any disks already in use in existing device groups. Note the logical device name (referred to as `cAtAdA` and `cBtBdB` in step 2). The following example checks against both VxVM and Solaris Volume Manager disks, just in case you are running both.

   ```
   # vxdisk -o alldgs list
   # metaset
   # scdidadm -L
   ```
3. Create a CVM shared disk group using these disks.

   ```
   # /etc/vx/bin/vxdisksetup -i cAtAdA format=sliced
   # /etc/vx/bin/vxdisksetup -i cBtBdB format=sliced
   # vxdg -s init ora-rac-dg cds=off ora1=cAtAdA \
   ora2=cBtBdB
   ```
4. Create the volumes for the database. You can use the provided `create_rac_vxvm_volumes` script. This script is reproduced here, if you want to type them in yourself rather than use the script:

```
vxassist -g ora-rac-dg make raw_system 450m layout=mirror
```

```
vxassist -g ora-rac-dg make raw_spfile 100m layout=mirror
vxassist -g ora-rac-dg make raw_users 120m layout=mirror
vxassist -g ora-rac-dg make raw_temp 100m layout=mirror
vxassist -g ora-rac-dg make raw_undotbs1 312m layout=mirror
vxassist -g ora-rac-dg make raw_undotbs2 312m layout=mirror
vxassist -g ora-rac-dg make raw_sysaux 300m layout=mirror
vxassist -g ora-rac-dg make raw_control1 110m layout=mirror
vxassist -g ora-rac-dg make raw_control2 110m layout=mirror
vxassist -g ora-rac-dg make raw_redo11 120m  layout=mirror
vxassist -g ora-rac-dg make raw_redo12 120m layout=mirror
vxassist -g ora-rac-dg make raw_redo21 120m layout=mirror
vxassist -g ora-rac-dg make raw_redo22 120m layout=mirror
vxassist -g ora-rac-dg make raw_ocr 100m layout=mirror
vxassist -g ora-rac-dg make raw_css_voting_disk 20m layout=mirror
```

5. Zero out the configuration and voting devices. This will eliminate problems that might arise from data left over from a previous class:

```
# dd if=/dev/zero of=/dev/vx/rdsk/ora-rac-dg/raw_ocr bs=16k
# dd if=/dev/zero of=/dev/vx/rdsk/ora-rac-dg/raw_css_voting_disk bs=16k
```

Note – Do not be concerned with error messages that appear at the end of running the above commands. You will probably get errors indicating that the end of the device does not align exactly on a 16k boundary

6. Change the owner and group associated with the newly-created volumes (assumes sh or ksh syntax here):

```
# cd /dev/vx/rdsk/ora-rac-dg
# for vol in *
> do
> vxedit -g ora-rac-dg set user=oracle group=dba $vol
> done
```

Task 5B – Creating Raw Volumes (Solaris Volume Manager)

To create the raw volumes for the ORACLE 10g RAC Database, perform steps 1-5 from *any one* of the selected cluster nodes. Please note that step 6 is the only one run on *all* selected nodes.

1. Select two disks from shared storage (one from one array and one from the other array) for a new disk set. Make sure you do not use any disks already in use in existing device groups. Note the device

identifier (DID) number of your two selected disks (referred to as d*A* and d*B* in step 2). The following example checks against both VxVM and Solaris Volume Manager disks, just in case you are running both.

```
# vxdisk -o alldgs list
# metaset
# scdidadm -L
```

2. Create a multiowner diskset using these disks.

```
# metaset -s ora-rac-ds -a -M -h node1 node2
# metaset -s ora-rac-ds -a /dev/did/rdsk/dA
# metaset -s ora-rac-ds -a /dev/did/rdsk/dB
```

3. Create a big mirror from which to partition all the necessary data volumes:

```
# metainit -s ora-rac-ds d11 1 1 /dev/did/rdsk/dAs0
# metainit -s ora-rac-ds d12 1 1 /dev/did/rdsk/dBs0
# metainit -s ora-rac-ds d10 -m d11
# metattach -s ora-rac-ds d10 d12
```

4. Create the volumes for the database. You can use the provided `create_rac_svm_volumes` script. This script is reproduced here, including the comments indicating which volumes you are creating. The commands are highlighted, if you want to type them in yourself rather than use the script.

```
# system volume d100 (450mb)
metainit -s ora-rac-ds d100 -p d10 450m

# spfile volume d101 (100mb)
metainit -s ora-rac-ds d101 -p d10 100m

# users volume d102 (120mb)
metainit -s ora-rac-ds d102 -p d10 120m

# temp volume d103 (100mb)
metainit -s ora-rac-ds d103 -p d10 100m

# undo volume #1 d104 (312mb)
metainit -s ora-rac-ds d104 -p d10 312m

# undo volume #2 d105 (312mb)
metainit -s ora-rac-ds d105 -p d10 312m

# sysaux volume d106 (300mb)
metainit -s ora-rac-ds d106 -p d10 300m
```

```
# control volume #1 d107 (100mb)
metainit -s ora-rac-ds d107 -p d10 110m

# control volume #2 d108 (100mb)
metainit -s ora-rac-ds d108 -p d10 110m

# redo volume #1.1 d109 (120m)
metainit -s ora-rac-ds d109 -p d10 120m

# redo volume #1.2 d110 (120m)
metainit -s ora-rac-ds d110 -p d10 120m

# redo volume #2.1 d111 (120m)
metainit -s ora-rac-ds d111 -p d10 120m

# redo volume #2.2 d112(120m)
metainit -s ora-rac-ds d112 -p d10 120m

# OCR Configuration volume d113 (100mb)
metainit -s ora-rac-ds d113 -p d10 100m

# CSS voting disk d114 (20mb)
metainit -s ora-rac-ds d114 -p d10 20m
```

5. From *either node*, zero out the configuration and voting devices. This will eliminate problems that might arise from data left over from a previous class:

```
# dd if=/dev/zero of=/dev/md/ora-rac-ds/rdsk/d113 bs=16k
# dd if=/dev/zero of=/dev/md/ora-rac-ds/rdsk/d114 bs=16k
```

6. On *all selected nodes,* change the owner and group associated with the newly-created volumes.

```
# chown oracle:dba /dev/md/ora-rac-ds/dsk/*
# chown oracle:dba /dev/md/ora-rac-ds/rdsk/*
```

Task 6 – Configuring the `oracle` User Environment

Perform the following steps on *all* selected cluster nodes:

1. switch user to `oracle` user by typing:

```
# su - oracle
```

2. Edit .profile (or in the profile.RAC provided) to include environment variables for ORACLE 10g RAC by typing:

```
$ vi .profile
ORACLE_BASE=/oracle
ORACLE_HOME=$ORACLE_BASE/product/10.1.0/db_1
CRS_HOME=$ORACLE_BASE/product/10.1.0/crs
TNS_ADMIN=$ORACLE_HOME/network/admin
DBCA_RAW_CONFIG=$ORACLE_BASE/dbca_raw_config
#SRVM_SHARED_CONFIG=/dev/md/ora-rac-ds/rdsk/d101
SRVM_SHARED_CONFIG=/dev/vx/rdsk/ora-rac-dg/raw_spfile
DISPLAY=display-station-name-or-IP:display#
if [ `/usr/sbin/clinfo -n` -eq 1 ]; then
        ORACLE_SID=sun1
fi
if [ `/usr/sbin/clinfo -n` = 2 ]; then
        ORACLE_SID=sun2
fi
PATH=/usr/ccs/bin:$ORACLE_HOME/bin:$CRS_HOME/bin:/usr/bin:/usr/sbin
export ORACLE_BASE ORACLE_HOME TNS_ADMIN DBCA_RAW_CONFIG CRS_HOME
export SRVM_SHARED_CONFIG ORACLE_SID PATH DISPLAY
```

3. Note the only modification in the file you may need to make depending on your choice of VxVM or Solaris VM. Make sure the line beginning SRVM_SHARED_CONFIG lists the correct option and that the other choice is deleted or commented out.

4. Make sure your actual X-Windows display is set correctly on the line that begins with DISPLAY=.

5. Read in the contents of your new .profile file and verify the environment.

```
$ . ./.profile
$ env
```

6. Enable rsh for the oracle user.

```
$ echo + >/oracle/.rhosts
```

Task 7A – Creating the dbca_raw_config File (VxVM)

Perform the following:

The dbca_raw_config file defines the locations of the raw devices of the ORACLE 10g RAC database. You must either copy the existing dbca_raw_config.vxvm file that your instructor has prepared for you or manually create the file as user oracle on *all* selected cluster nodes.

```
$ vi /oracle/dbca_raw_config
system=/dev/vx/rdsk/ora-rac-dg/raw_system
spfile=/dev/vx/rdsk/ora-rac-dg/raw_spfile
users=/dev/vx/rdsk/ora-rac-dg/raw_users
temp=/dev/vx/rdsk/ora-rac-dg/raw_temp
undotbs1=/dev/vx/rdsk/ora-rac-dg/raw_undotbs1
undotbs2=/dev/vx/rdsk/ora-rac-dg/raw_undotbs2
sysaux=/dev/vx/rdsk/ora-rac-dg/raw_sysaux
control1=/dev/vx/rdsk/ora-rac-dg/raw_control1
control2=/dev/vx/rdsk/ora-rac-dg/raw_control2
redo1_1=/dev/vx/rdsk/ora-rac-dg/raw_redo11
redo1_2=/dev/vx/rdsk/ora-rac-dg/raw_redo12
redo2_1=/dev/vx/rdsk/ora-rac-dg/raw_redo21
redo2_2=/dev/vx/rdsk/ora-rac-dg/raw_redo22
```

Task 7B – Creating the dbca_raw_config File (Solaris VM)

Perform the following:

The dbca_raw_config file defines the locations of the raw devices of the ORACLE 10g RAC database. You must either copy the existing dbca_raw_config.svm file that your instructor has prepared for you or manually create the file as user oracle on *all* selected cluster nodes.

```
$ vi /oracle/dbca_raw_config
system=/dev/md/ora-rac-ds/rdsk/d100
spfile=/dev/md/ora-rac-ds/rdsk/d101
users=/dev/md/ora-rac-ds/rdsk/d102
temp=/dev/md/ora-rac-ds/rdsk/d103
undotbs1=/dev/md/ora-rac-ds/rdsk/d104
undotbs2=/dev/md/ora-rac-ds/rdsk/d105
sysaux=/dev/md/ora-rac-ds/rdsk/d106
control1=/dev/md/ora-rac-ds/rdsk/d107
control2=/dev/md/ora-rac-ds/rdsk/d108
redo1_1=/dev/md/ora-rac-ds/rdsk/d109
redo1_2=/dev/md/ora-rac-ds/rdsk/d110
redo2_1=/dev/md/ora-rac-ds/rdsk/d111
redo2_2=/dev/md/ora-rac-ds/rdsk/d112
```

Task 8 – Disabling Access Control on X Server of the Admin Workstation

Perform the following:

To allow client GUIs to be displayed, run the following command on the admin workstation or display station:

```
(# or $) /usr/openwin/bin/xhost +
```

Task 9 – Installing ORACLE CRS Software

Install the ORACLE Cluster Ready Services (CRS) software from the first selected cluster node. The installer automatically copies over the binaries to the other nodes using `rsh` at the end of the installation.

Perform the following steps on the first selected cluster node as the `oracle` user:

1. Change directory to the ORACLE 10g CRS 10.1.0.2 software location by typing:

   ```
   $ cd ORACLE10g-CRS-10.1.0.2-software-location/Disk1
   ```

2. Run the `runInstaller` installation program by typing:

   ```
   $ ./runInstaller -ignoreSysPrereqs
   ```

3. Respond to the dialog boxes using Table 11-2.

Table 11-2 ORACLE CRS Installation Dialog Actions

Dialog	Action
Welcome	Click Next.
Specify Inventory directory and credentials	Verify, and click Next.
Script	Open a terminal window as user `root` on the node on which you are running the installer, and run the script `/oracle/oraInventory/orainstRoot.sh`. When it is finished, click Continue.
Specify File Locations	Verify, and click Next.
Language Selection	Choose any additional languages you want, and click Next.

Table 11-2 ORACLE CRS Installation Dialog Actions (Continued)

Dialog	**Action**
Cluster Configuration	Enter the name of the cluster. Enter the private node name for each node. The private node names are literally, `clusternode1-priv`, `clusternode2-priv`, and so on unless you have changed them. Verify that you are associating the correct private hostnames with the corresponding nodes by typing in any node terminal window: `scconf -pvv | grep 'private hostname'` Do to a Java graphical oddity, you must click in a different box after typing the last name for it to register. Click Next. If you get some error concerning a "public node not reachable," it is probably because you do not have an `/oracle/.rhosts` file. You must have one even on the node on which you are running the installer.
Private Interconnect Enforcement	Mark *only the first public adapter that you see* as `public.` Mark *only* the `clprivnet0` interface as `private.` Leave *all other adapters*, including actual private network adapters, as `Do Not Use.` Click Next.
Oracle Cluster Registry	VxVM: Enter `/dev/vx/rdsk/ora-rac-dg/raw_ocr` SVM: Enter `/dev/md/ora-rac-ds/rdsk/d113` Click Next.
Voting Disk	VxVM: Enter `/dev/vx/rdsk/ora-rac-dg/raw_css_voting_disk` SVM: Enter `/dev/md/ora-rac-ds/rdsk/d114` Click Next.
Script	On *all selected nodes* open a terminal window as user `root` and run the script `/oracle/oraInventory/orainstRoot.sh` When you have run it on all nodes, click Continue.

Table 11-2 ORACLE CRS Installation Dialog Actions (Continued)

Dialog	**Action**
Summary	Verify, and click Install. **Note –** The installation status bar remains at 98 percent while the CRS binaries are being copied from the installing node to the other nodes. When the installation is almost complete, you are prompted to run a script on the selected cluster nodes.
Script (Setup Privileges)	On all selected nodes, *one at a time, starting with the node on which you are running the installer,* open a terminal window as user `root` and run the script `/oracle/product/10.1.0/crs/root.sh` The script formats the voting device and enables the CRS daemons on each node. Entries are put in `/etc/inittab` so that the daemons run at boot time. When you have run it to completion on all nodes, one at a time, click OK.
Configuration Assistants	Let them run to completion and click Next.
End of Installation	Click Exit and confirm.

Task 10 – Upgrading ORACLE CRS Software

Upgrade the ORACLE Cluster Ready Services (CRS) software from the first selected cluster node. The installer automatically copies over the binaries to the other nodes using `rsh` at the end of the installation.

Perform the following steps on the first selected cluster node as the `oracle` user:

1. Change directory to the ORACLE10g 10.1.0.4 patch location by typing:

   ```
   $ cd ORACLE10g-10.1.0.4-patch-location/Disk1
   ```

2. Run the `runInstaller` installation program by typing:

   ```
   $ ORACLE_HOME=$CRS_HOME; export ORACLE_HOME
   $ ./runInstaller
   ```

3. Respond to the dialog boxes using Table 11-3.

Table 11-3 Upgrade ORACLE CRS Dialog Actions

Dialog	Action
Welcome	Click Next.
Specify File Locations	Verify, especially that the Destination Path is `/oracle/product/10.1.0/crs`. If it is not, you might have not set the `ORACLE_HOME` variable correctly, as described previously in step 2. Click Next.
Select Nodes	Verify that this includes your selected nodes, and click Next.
Summary	Verify, and click Install. Your 10.1.0.4 CRS upgrade is installed and copied to the other nodes. The meter remains at 95% as the software is copied.

Table 11-3 Upgrade ORACLE CRS Dialog Actions (Continued)

Dialog	Action

End of Installation	Log on as root on each of your cluster nodes, *one at a time starting with the one on which you are running the installer.* Follow the directions indicated in the installer window:
	`# /etc/init.d/init.crs stop`
	You will see INIT errors on the console (this is normal).
	`# /oracle/product/10.1.0/crs/install/root10104.sh`
	Click Exit when the procedure has run to completion on all of your selected nodes.

Task 11 – Installing Oracle Database Software

Upgrade the ORACLE database software from the first selected cluster node. The installer automatically copies over the binaries to the other nodes using `rsh` at the end of the installation.

Perform the following steps on the first selected cluster node as the `oracle` user:

1. Log out and back in as user `oracle` in order to reset the environment:

   ```
   $ exit
   # su - oracle
   ```

2. Change directory to the ORACLE10g 10.1.0.2 database location by typing:

   ```
   $ cd ORACLE10g-10.1.0.2-db-location/Disk1
   ```

3. Run the `runInstaller` installation program by typing:

   ```
   $ ./runInstaller -ignoreSysPrereqs
   ```

4. Respond to the dialog boxes using Table 11-4.

Table 11-4 Install ORACLE database software Dialog Actions

Dialog	Action
Welcome	Click Next.
Specify File Locations	Verify, especially that the Destination Path is `/oracle/product/10.1.0/db_1`. Click Next.
Specify Hardware Cluster Installation Mode	Verify that the Cluster Installation radio button is selected. Put check marks next to *all* of your selected cluster nodes. Click Next.
Select Installation Type	Select the Custom radio button, and click Next.
Product Specific Prerequisite Checks	You will get an error that Solaris 5.10 is unsupported. Ignore this error and click Next.
Warning	Ignore the warning about the missing package, and click Continue.
Warning	Ignore the warning about the kernel parameters (they are obsolete in Solaris 10), and click OK.
Available Product Components	Deselect the following components: ● Oracle Advanced Security 10.1.0.2.0 ● Oracle Enterprise Manager 10g Database Control 10.1.0.20 ● Oracle Development Kit 10.1.0.2.0 ● Oracle Transparent Gateways 10.1.0.2.0 Click Next.
Privileged Operating System Groups	Verify that `dba` is listed in both entries, and click Next.
Create Database	Select the No radio button, and click Next.

Table 11-4 Install ORACLE database software Dialog Actions (Continued)

Dialog	**Action**
Summary	Verify, and click Install. It will remain at 95% for a while (linking) and 99% for a while (copying files to other nodes).

Script (setup privileges)	On each node, *one at a time starting with the node on which you are running the installer*, log in as root. The script you will run expects the DISPLAY to be set:
	`# env | grep DISPLAY`
	If it is not set, set it:
	`# DISPLAY=`*`display-name-or-IP`*`:#`
	`# export DISPLAY`
	and run the script:
	`# /oracle/product/10.1.0/db_1/root.sh`
	Accept the default for the local bin directory.
	On each node the script wants to run the VIP Configuration Assistant. Click Cancel when you reach the VIP Configuration welcome screen (this will not work anyway until you upgrade to 10.1.0.4).
	When you have run the script on all nodes, click OK.

Configuration Assistants	Click Cancel when you reach the Net Configuration Assistant screen (this will not work until you successfully run VIPCA after the upgrade). Acknowledge the Error Screen (click OK). Click Next to leave the Configuration Assistants screen. Acknowledge the Warning (click OK).

Table 11-4 Install ORACLE database software Dialog Actions (Continued)

Dialog	**Action**
End of Installation	Click Exit.

Task 12 – Upgrading ORACLE Database Software

Upgrade the ORACLE Database software from the first selected cluster node. The installer automatically copies over the binaries to the other nodes using `rsh` at the end of the installation.

Perform the following steps on the first selected cluster node as the `oracle` user:

1. Change directory to the ORACLE10g 10.1.0.4 patch location by typing:
   ```
   $ cd ORACLE10g-10.1.0.4-patch-location/Disk1
   ```
2. Run the `runInstaller` installation program by typing:
   ```
   $ ./runInstaller
   ```
3. Respond to the dialog boxes using Table 11-5.

Table 11-5 Upgrade ORACLE Database Dialog Actions

Dialog	**Action**
Welcome	Click Next.
Specify File Locations	Verify that the Destination Path is `/oracle/product/10.1.0/db_1`. Click Next.
Select Nodes	Verify that this includes your selected nodes, and click Next.
Summary	Verify, and click Install. Your 10.1.0.4 Database upgrade is installed and copied to the other nodes) The meter remains at 95% as the software is linked and 99% as the software is copied.

Table 11-5 Upgrade ORACLE Database Dialog Actions (Continued)

Dialog	Action
Script (setup privileges)	On each node, *one at a time starting with the node on which you are running the installer*, log in as root and run the script: `# /oracle/product/10.1.0/db_1/root.sh` Accept the default for the local bin directory. Confirm that you want to overwrite the files that are already there. When you have run the script on all nodes, click OK.
End of Installation	Click Exit.

Task 13 – Configuring Oracle Virtual IPs

Configure virtual IPs for use by the Oracle RAC database. Oracle CRS will control these IPs, failing over both of them to a surviving node if one of the nodes crashes. When one of these IPs fails over, Oracle clients do *not* successfully contact the database using that IP. Instead, they get a "Connection Refused" indication, and have their client software set to automatically try the other IP.

Perform the following on *all* selected nodes as root (you can edit the `hosts` file on one node and copy it over or paste in your entries, if you like):

Edit the `/etc/hosts` file and create public network entries for new virtual IPs. You will have one IP per node that you are using with Oracle RAC. If you use consecutive IP addresses then the configuration assistant in the next task will automatically "guess" the second IP when you type in the first.

```
# vi /etc/hosts
x.y.z.w      vip-ora-nodename1
x.y.z.w+1    vip-ora-nodename2
```

Task 14 – Running the vipca Utility

Perform the following steps:

1. From *one node*, run the vipca utility as root.

```
# ORACLE_HOME=/oracle/product/10.1.0/db_1
# PATH=$PATH:$ORACLE_HOME/bin
# DISPLAY=display-name-or-IP:#
# export ORACLE_HOME PATH DISPLAY
# vipca
```

2. Respond to the dialog boxes using Table 11-6.

Table 11-6 VIPCA Dialog Actions

Dialog	Action
Welcome	Click Next.
Network Interfaces	Select *only your first public network interface.* Deselect all others. Click Next.
Virtual IP's for Cluster Nodes	Type the /etc/hosts name for the first node, vip-ora-*nodename1*. When you press TAB the form should automatically fill in IP addresses, and should fill in information for other nodes if the IP addresses are consecutive. If not, fill it in manually. Verify the netmasks are correct. Click Next.
Summary	Verify, and click Finish.
Configuration Assistant Progress Dialog	Confirm that the utility runs to 100%, and click OK.
Configuration Results	Verify, and click Exit.

Task 15 – Configuring an ORACLE Listener

On *one* cluster node, configure the ORACLE listener by performing the following steps as user `oracle`:

1. Run the Net Configuration Assistant by typing:

```
$ netca
```

2. Respond to the dialog boxes using Table 11-7.

Table 11-7 ORACLE Listener Dialog

Dialog	**Action**
Real Application Clusters Configuration	Verify that the Cluster configuration radio button is selected, and click Next.
Real Application Clusters, Active Nodes	Verify that all selected cluster nodes are highlighted, and click Next.
Welcome	Verify that the Listener configuration radio button is selected, and click Next.
Listener Configuration, Listener	Verify that the Add radio button is selected, and click Next.
Listener Configuration, Listener Name	Verify that the Listener name text field contains the word LISTENER, and click Next.
Listener Configuration, Select Protocols	Verify that TCP is in the Selected Protocols text area, and click Next.
Listener Configuration, TCP/IP Protocol	Verify that the Use the standard port number of 1521 radio button is selected, and click Next.
Listener Configuration, More Listeners?	Verify that the No radio button is selected, and click Next.
Listener Configuration Done	Click Next.
Welcome	Click Finish.

Task 16 – Creating an ORACLE Database

Perform the following steps on *one* of the selected cluster nodes:

1. Run the Database Configuration Assistant as user `oracle` by typing:

```
$ dbca
```

2. Use Table 11-8 to respond to the dialog boxes.

Table 11-8 ORACLE Database Creation Dialog Answers

Dialog	**Action**
Welcome	Verify that the Oracle Real Application Clusters database radio button is selected, and click Next.
Step 1: Operations	Verify that the Create a database radio button is selected, and click Next.
Step 2: Node Selection	Select all of your Oracle RAC nodes and click Next.
Step 3: Database Templates	Select General Purpose radio button, and click Next.
Step 4: Database Identification	Type **`sun`** in the Global Database Name text field (notice that your keystrokes are echoed in the SID Prefix text field), and click Next.
Step 5: Management Options	Verify that Enterprise Manager is not available (you eliminated it when you installed the database software) and click Next.
Step 6: Database Credentials	Verify that the Use the Same Password for All Accounts radio button is selected. Enter `cangetin` as the password, and click Next.
Step 7: Storage Options	Select the Raw Devices radio button. Verify that the checkbox for Raw Devices Mapping File is selected, and that the value is `/oracle/dbca_raw_config`.
Step 8: Recovery Configuration	Leave the boxes unchecked, and click Next.
Step 9: Database Content	Click Next.
Step 10: Database Services	Click Next.

Table 11-8 ORACLE Database Creation Dialog Answers (Continued)

Dialog	**Action**
Step 11: Initialization Parameters	On the Memory tab, verify that the Typical radio button is selected. Change the Percentage to a ridiculously small number (1%). Click Next and accept the error telling you the minimum memory required. The percentage will automatically be changed on your form. Click Next.
Step 12: Database Storage	Verify that the database storage locations are correct by clicking on leaves in the file tree in the left pane and examining the values shown in the right pane. Note that these locations are determined by the contents of the `/oracle/dbca_raw_config` file that you prepared in a previous task. Click Next.
Step 13: Creation Options	Verify that the Create Database check box is selected, and click Finish.
Summary	Verify and click OK. **Note –** The database creation takes from 20 to 30 minutes. If you see a warning pop up about failure to set the time zone, this can be safely ignored.
Final Screen (Database Configuration Assistant)	Confirm and click Exit. You will see a popup telling you that the database instances are being started. The popup will disappear when the database is online on all nodes.

Task 17 – Verifying That ORACLE RAC Works Properly in a Cluster

Run the following commands as indicated to verify the ORACLE software properly runs in the cluster:

1. Switch user to the `oracle` user by typing (all selected cluster nodes):

   ```
   # su - oracle
   ```

2. On one node, connect to database sub-instance `sun1` and create a tab.

```
$ sqlplus SYS@sun1 as sysdba
Enter password: cangetin

SQL> create table mytable (name VARCHAR2(10), age NUMBER(10));
SQL> insert into mytable values ('yourname', age);
SQL> commit;
SQL> select * from mytable;
SQL> quit
```

3. From the other node, query the other database sub-instance and verify that the data is there:

   ```
   $ sqlplus SYS@sun2 as sysdba
   Enter password: cangetin

   SQL> select * from mytable;
   SQL> quit
   ```

4. Send the break signal to the console of node 1 to simulate a crash of that node.

   ```
   # Control-]
   telnet> send break
   ```

5. On the surviving node, you should see (after 45 seconds or so), the CRS-controlled virtual IP for crashed node migrate to the surviving node:

   ```
   $ crs_stat -t|grep vip

   ora.node2.vip  application    ONLINE    ONLINE node2
   ora.node1.vip  application    ONLINE    ONLINE node2
   $
   ```

6. While this virtual IP has failed over, verify that there is actually no failover listener controlled by Oracle CRS. This virtual IP fails over merely so a client quickly gets a TCP disconnect without having to wait for a long time-out. Client software then has a client-side option to fail over to the other instance.

```
$ sqlplus SYS@sun1 as sysdba

SQL*Plus: Release 10.1.0.4.0 - Production on Tue May 24
10:56:18 2005

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

Enter password:
ERROR:
ORA-12541: TNS:no listener


Enter user-name: ^D
```

7. Boot the node that you had halted, by typing `boot` or `go` at the `OK` prompt in the console. If you choose the latter, the node will panic and reboot.

8. After the node boots, monitor the automatic recovery of the virtual IP, the listener, and the database instance by typing, as user `oracle`, on the surviving node:

```
$ crs_stat -t
```

It can take several minutes for the full recovery.

If you look at the console on the booting node, you may see the Oracle CRS seem to stop, and then automatically restart after the `rac-framework-rg` starts properly.

Repeat the preceding command until all Oracle resources are in the `ONLINE` state on their proper nodes.

9. Verify the proper operation of the Oracle database by contacting the various sub-instances as the user `oracle` on the various nodes:

```
$ sqlplus SYS@sun1 as sysdba
Enter password: cangetin

SQL> select * from mytable;
SQL> quit
```

Exercise Summary

Discussion – Take a few minutes to discuss what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications

Appendix A

Terminal Concentrator

This appendix describes the configuration of the Sun Terminal Concentrator (TC) as a remote connection mechanism to the serial consoles of the nodes in the Sun Cluster software environment.

Viewing the Terminal Concentrator

The Sun Terminal Concentrator (Annex NTS) has its own internal operating system and resident administration programs.

Note – If any other TC is substituted, it *must not* send an abort signal to the attached host systems when it is powered on.

Figure A-1 shows the TC is a self-contained unit with its own operating system.

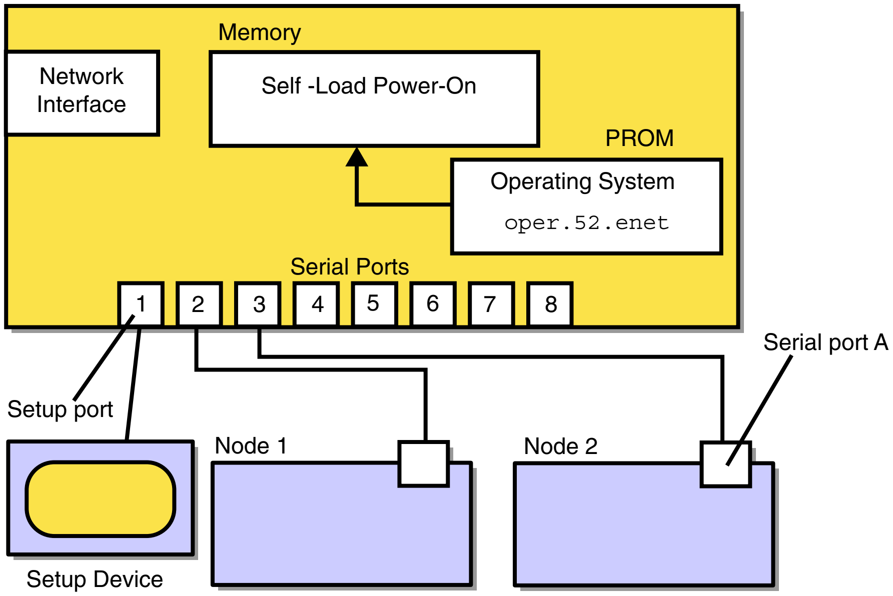


Figure A-1 Terminal Concentrator Functional Diagram

Note – If the programmable read-only memory (PROM) operating system is older than version 52, you must upgrade it.

Operating System Load

You can set up the TC to load its operating system either internally from the resident PROM or externally from a server. In the cluster application, it is always set to load internally. Placing the operating system on an external server can decrease the reliability of the terminal server.

When power is first applied to the TC, it performs the following steps:

1. It runs a PROM-based self-test and displays error codes.
2. It loads a resident PROM-based operating system into the TC memory.

Setup Port

Serial port 1 on the TC is a special purpose port that is used only during initial setup. It is used primarily to set up the IP address and load sequence of the TC. You can access port 1 from either a `tip` connection or from a locally connected terminal.

Terminal Concentrator Setup Programs

You must configure the TC nonvolatile random access memory (NVRAM) with the appropriate IP address, boot path, and serial port information. Use the following resident programs to specify this information:

- `addr`
- `seq`
- `image`
- `admin`

Setting Up the Terminal Concentrator

The TC must be configured for proper operation. Although the TC setup menus seem simple, they can be confusing and it is easy to make a mistake. You can use the default values for many of the prompts.

Connecting to Port 1

To perform basic TC setup, you must connect to the TC setup port. Figure A-2 shows a tip hardwire connection from the administrative console. You can also connect an American Standard Code for Information Interchange (ASCII) terminal to the setup port.

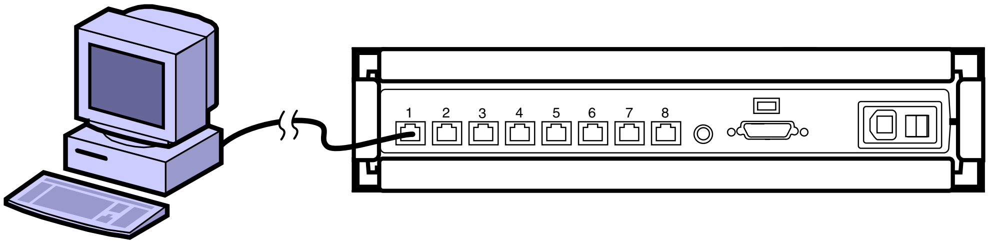


Figure A-2 Setup Connection to Port 1

Enabling Setup Mode

To enable Setup mode, press the TC Test button shown in Figure A-3 until the TC power indicator begins to blink rapidly, then release the Test button and press it again briefly.

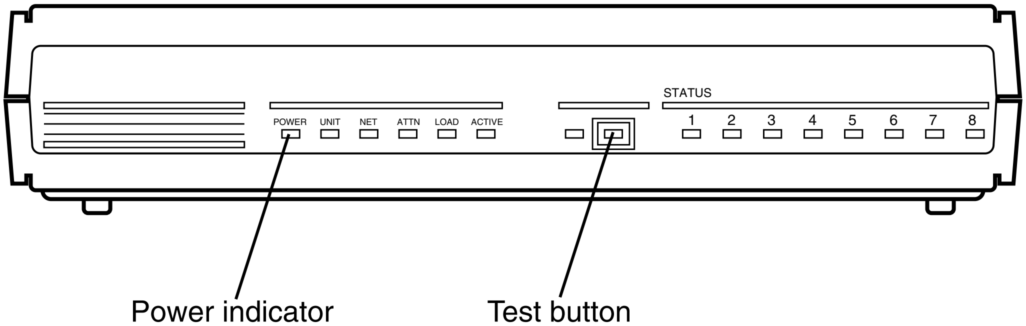


Figure A-3 Terminal Concentrator Test Button

After you have enabled Setup mode, a `monitor::` prompt should appear on the setup device. Use the `addr`, `seq`, and `image` commands to complete the configuration.

Setting the Terminal Concentrator IP Address

The following example shows how to use the addr program to set the IP address of the TC. Usually this is set correctly when your cluster arrives, but you should always verify that it is correct.

```
monitor:: addr
Enter Internet address [192.9.22.98]:: 129.150.182.100
Enter Subnet mask [255.255.255.0]::
Enter Preferred load host Internet address
[192.9.22.98]:: 129.150.182.100
Enter Broadcast address [0.0.0.0]:: 129.150.182.255
Enter Preferred dump address [192.9.22.98]::
129.150.182.100
Select type of IP packet encapsulation
(ieee802/ethernet) [<ethernet>]::
    Type of IP packet encapsulation: <ethernet>

Load Broadcast Y/N [Y]:: y
```

Setting the Terminal Concentrator Load Source

The following example shows how to use the seq program to specify the type of loading mechanism to be used:

```
monitor:: seq
Enter a list of 1 to 4 interfaces to attempt to use for
downloading code or upline dumping. Enter them in the
order they should be tried, separated by commas or
spaces. Possible interfaces are:

    Ethernet: net
    SELF:  self

Enter interface sequence [self]::
```

The self response configures the TC to load its operating system internally from the PROM when you turn on the power. The PROM image is currently called oper.52.enet.

Enabling the self-load feature negates other setup parameters that refer to an external load host and dump host, but you must still define these parameters during the initial setup sequence.

Note – Although you can load the TC's operating system from an external server, this introduces an additional layer of complexity that is prone to failure.

Specifying the Operating System Image

Even though the self-load mode of operation negates the use of an external load and dump device, you should still verify the operating system image name as shown by the following:

```
monitor:: image

    Enter Image name ["oper.52.enet"]::
    Enter TFTP Load Directory ["9.2.7/"]::
    Enter TFTP Dump path/filename
["dump.129.150.182.100"]::

monitor::
```

Note – Do not define a dump or load address that is on another network because you receive additional questions about a gateway address. If you make a mistake, you can press `Control-C` to abort the setup and start again.

Setting the Serial Port Variables

The TC port settings must be correct for proper cluster operation. This includes the `type` and `mode` port settings. Port 1 requires different `type` and `mode` settings. You should verify the port settings before installing the cluster host software. The following is an example of the entire procedure:

```
admin-ws# telnet terminal_concentrator_name
Trying terminal concentrator IP address ...
Connected to terminal concentrator IP address.
Escape character is '^]'.
Rotaries Defined:
    cli                               -
Enter Annex port name or number: cli
Annex Command Line Interpreter  *  Copyright 1991
Xylogics, Inc.
annex: su
```

```
Password: type the password
annex# admin
Annex administration MICRO-XL-UX R7.0.1, 8 ports
admin: show port=1 type mode
Port 1:
type: hardwired     mode: cli
admin:set port=1 type hardwired mode cli
admin:set port=2-8 type dial_in mode slave
admin:set port=1-8 imask_7bits Y
admin: quit
annex# boot
bootfile: <CR>
warning: <CR>
```

Note – Do not perform this procedure through the special setup port; use public network access.

Setting the Port Password

An optional and recommended security feature is to set per-port passwords. These provides an extra password challenge as you access a serial port in slave mode remotely through the `telnet` command.

You can set different (or the same) port passwords on each port. You must set the `enable_security` parameter to `Y` to enable all the passwords.

If you ever forget a port password but know the TC root password, you can just reset the passwords to whatever you want.

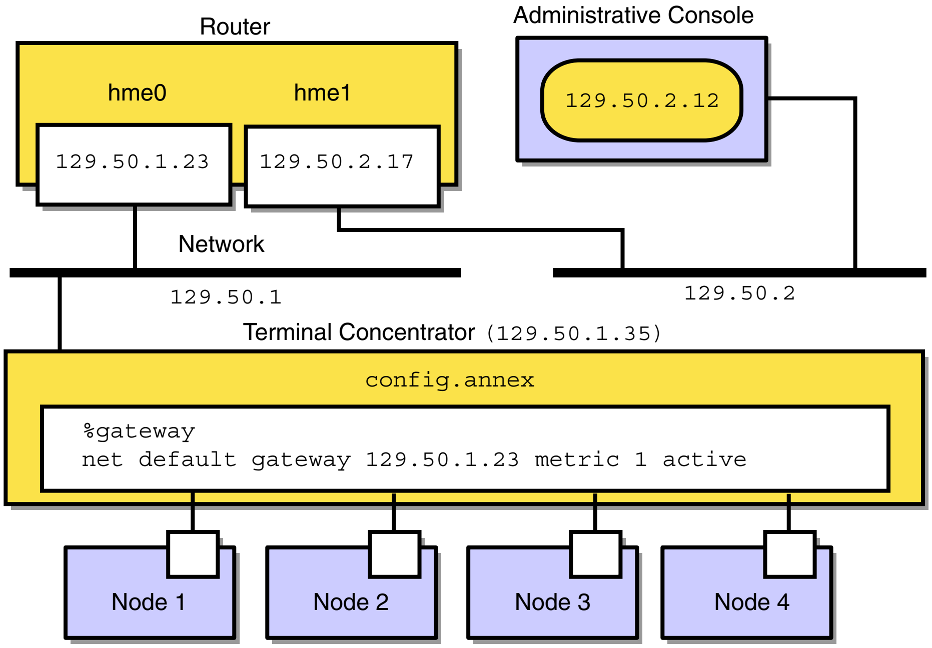
```
admin-ws# telnet terminal_concentrator_name
Trying terminal concentrator IP address . . .
Connected to terminal concentrator IP address.
Escape character is '^]'.
Rotaries Defined:
    cli                              -
Enter Annex port name or number: cli
Annex Command Line Interpreter  *  Copyright 1991
Xylogics, Inc.
annex: su
Password: type the password
annex# admin
Annex administration MICRO-XL-UX R7.0.1, 8 ports
```

```
admin: set port=2 port_password homer
admin: set port=3 port_password marge
admin: reset 2-3
admin: set annex enable_security Y
admin: reset annex security
```

Setting a Terminal Concentrator Default Route

If you access a TC from a host that is on a different network, the TC's internal routing table can overflow. If the TC routing table overflows, network connections can be intermittent or lost completely.

Figure A-4 shows that you can correct this problem by setting a default route within the TC `config.annex` configuration file.



```
%gateway
net default gateway 129.50.1.23 metric 1 active
```

Figure A-4 Terminal Concentrator Routing

Creating a Terminal Concentrator Default Route

To create a default route for the TC, you must edit an electrically erasable programmable read-only memory (EEPROM) file in the TC named `config.annex`. You must also disable the TC routing function. The following is a summary of the general process:

```
admin-ws# telnet tc1.central
Trying 129.50.1.35 ...
Connected to 129.50.1.35.
Escape character is '^]'.
[Return] [Return]
Enter Annex port name or number: cli
...
annex: su
Password: root_password
annex# edit config.annex
(Editor starts)
Ctrl-W:save and exit Ctrl-X:exit Ctrl-F:page down
Ctrl-B:page up
%gateway
net default gateway 129.50.1.23 metric 1 active ^W
annex# admin set annex routed n
You may need to reset the appropriate port, Annex
subsystem or reboot the Annex for changes to take
effect.
annex# boot
```

Note – You must enter an IP routing address appropriate for your site. While the TC is rebooting, the node console connections are not available.

Using Multiple Terminal Concentrators

A single TC can provide serial port service to a maximum of eight nodes. If it is necessary to reconfigure the TC, the node connection to port 1 must be switched with a serial connection to the configuration device, and the TC placed into setup mode. After configuration is complete, the normal node connection to port 1 is replaced and the TC rebooted.

The maximum length for a TC serial port cable is approximately 348 feet. As shown in Figure A-5, it might be necessary to have cluster host systems separated by more than the serial port cable limit. You might need a dedicated TC for each node in a cluster.

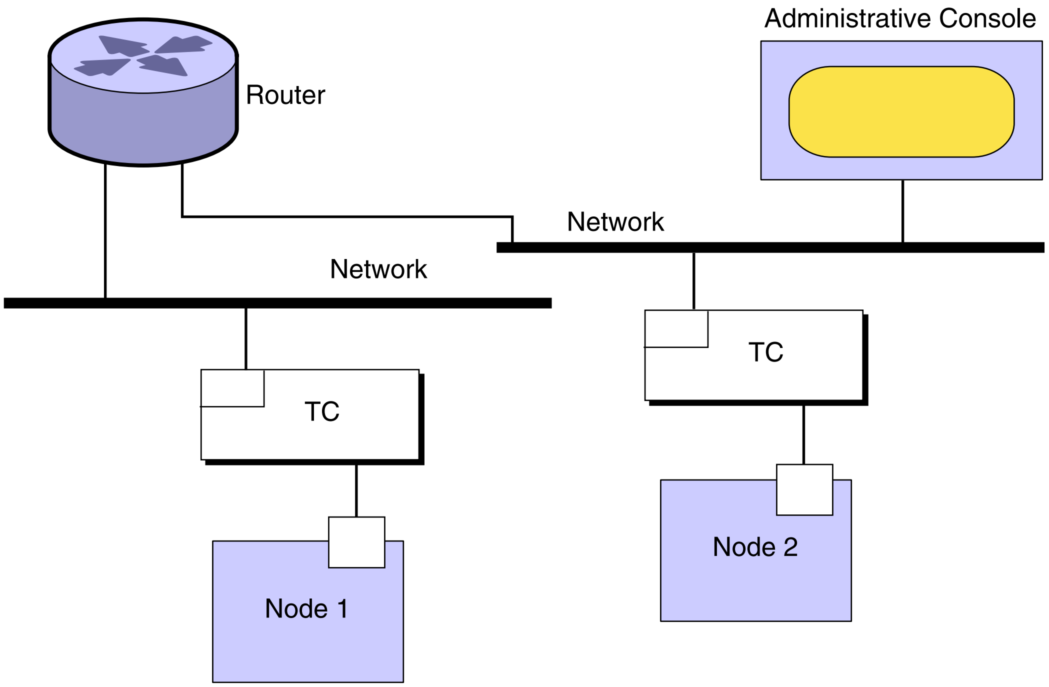


Figure A-5 Multiple Terminal Concentrators

Troubleshooting Terminal Concentrators

Occasionally, it is useful to be able to manually manipulate the TC. The commands to do this are not well documented in the cluster manuals.

Using the `telnet` Command to Manually Connect to a Node

If the `cconsole` tool is not using the TC serial ports, you can use the `telnet` command to connect to a specific serial port as follows:

```
# telnet tc_name 5002
```

You can then log in to the node attached to port 5002. After you have finished and logged out of the node, you must break the `telnet` connection with the Control-] keyboard sequence and then type **`quit`**. If you do not, the serial port remains locked and cannot be used by other applications, such as the `cconsole` tool.

Using the `telnet` Command to Abort a Node

If you have to abort a cluster node, you can either use the `telnet` command to connect directly to the node and use the Control-] keyboard sequence, or you can use the Control-] keyboard sequence in a cluster console window. When you have the `telnet` prompt, you can abort the node with the following command:

```
telnet > send brk
ok
```

Note – You might have to repeat the command multiple times.

Connecting to the Terminal Concentrator Command-Line Interpreter

You can use the telnet command to connect directly to the TC, and then use the resident command-line interface (CLI) to perform status and administration procedures.

```
# telnet IPaddress
Trying 129.146.241.135...
Connected to 129.146.241.135
Escape character is '^]'.

Enter Annex port name or number: cli
Annex Command Line Interpreter * Copyright 1991
Xylogics, Inc.
annex:
```

Using the Terminal Concentrator help Command

After you connect directly into a terminal concentrator, you can get online help as follows:

```
annex: help
annex: help hangup
```

Identifying and Resetting a Locked Port

If a node crashes, it can leave a telnet session active that effectively locks the port from further use. You can use the who command to identify which port is locked, and then use the admin program to reset the locked port. The command sequence is as follows:

```
annex: who
annex: su
Password:
annex# admin
Annex administration MICRO-XL-UX R7.0.1, 8 ports
admin : reset 6
admin : quit
annex# hangup
```

Erasing Terminal Concentrator Settings

Using the TC erase command can be dangerous. Use it only when you have forgotten the superuser password. It returns all settings to their default values. When the addr command is then run to give the TC its IP address, the password will be set to this IP. For security reasons, the erase command is available only through the port 1 interface. A typical procedure is as follows:

```
monitor :: erase

        1) EEPROM(i.e. Configuration information)
        2) FLASH(i.e. Self boot image)

Enter 1 or 2 :: 1
```

Caution – Do not use option 2 of the erase command; it destroys the TC boot PROM-resident operating system.

Appendix B

Configuring Multi-Initiator SCSI

This appendix contains information that can be used to configure multi-initiator Small Computer System Interface (SCSI) storage devices including the Sun StorEdge MultiPack desktop array and the Sun StorEdge D1000 array.

Multi-Initiator Overview

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

In a standalone server, the server node controls the SCSI bus activities using 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. 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 setting the `scsi-initiator-id` OpenBoot PROM property. 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 3.1 Hardware Guide*.

Installing a Sun StorEdge Multipack Device

This section provides the procedure for an initial installation of a Sun StorEdge MultiPack device.

Use this procedure to install a Sun StorEdge MultiPack device in a cluster prior to installing the Solaris OS and Sun Cluster software. Perform this procedure with the procedures in the *Sun Cluster 3.1 Installation Guide* and your server hardware manual.

1. Ensure that each device in the SCSI chain has a unique SCSI address.

   The default SCSI address for host adapters is 7. Reserve SCSI address 7 for one host adapter in the SCSI chain. This procedure refers to the host adapter you choose for SCSI address 7 as the host adapter on the `second` node. To avoid conflicts, in Step 7 you change the `scsi-initiator-id` of the remaining host adapter in the SCSI chain to an available SCSI address. This procedure refers to the host adapter with an available SCSI address as the host adapter on the `first` node. Depending on the device and configuration settings of the device, either SCSI address 6 or 8 is usually available.

Caution – Even though a slot in the device might not be in use, you should avoid setting `scsi-initiator-id` for the first node to the SCSI address for that disk slot. This precaution minimizes future complications if you install additional disk drives.

For more information, refer to the *OpenBoot™* 3.x *Command Reference Manual* and the labels inside the storage device.

2. Install the host adapters in the nodes that will be connected to the device.

For the procedure on installing host adapters, refer to the documentation that shipped with your nodes.

3. Connect the cables to the device, as shown in Figure B-1 on page B-4.

Make sure that the *entire* SCSI bus length to each device is less than 6 meters. This measurement includes the cables to both nodes, as well as the bus length internal to each device, node, and host adapter. Refer to the documentation which shipped with the device for other restrictions regarding SCSI operation.

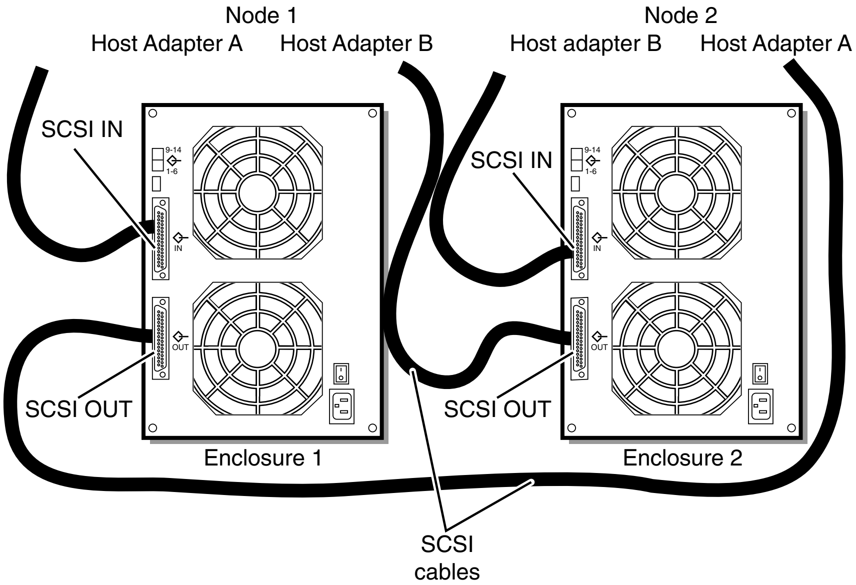


Figure B-1 Example of a Sun StorEdge MultiPack Desktop Array Enclosure Mirrored Pair

4. Connect the AC power cord for each device of the pair to a different power source.
5. Without allowing the node to boot, power on the first node. If necessary, abort the system to continue with OpenBoot PROM Monitor tasks.
6. Find the paths to the host adapters.

   ```
   ok show-disks
   ```

   Identify and record the two controllers that will be connected to the storage devices, and record these paths. Use this information to change the SCSI addresses of these controllers in the `nvramrc` script. Do not include the `/sd` directories in the device paths.
7. Edit the `nvramrc` script to set the `scsi-initiator-id` for the host adapter on the first node.

   For a list of `nvramrc` editor and `nvedit` keystroke commands, see the "The `nvramrc` Editor and `nvedit` Keystroke Commands" on page B-11.

The following example sets the scsi-initiator-id to 6. The OpenBoot PROM Monitor prints the line numbers (0:, 1:, and so on).

```
nvedit
0: probe-all
1: cd /sbus@1f,0/
2: 6 encode-int " scsi-initiator-id" property
3: device-end
4: cd /sbus@1f,0/SUNW,fas@2,8800000
5: 6 encode-int " scsi-initiator-id" property
6: device-end
7: install-console
8: banner <Control-C>
ok
```

Note – Insert exactly one space after the first double quotation mark and before scsi-initiator-id.

8. Store the changes.

   The changes you make through the nvedit command are done on a temporary copy of the nvramrc script. You can continue to edit this copy without risk. After you complete your edits, save the changes. If you are not sure about the changes, discard them.

   - To discard the changes, type:

     ```
     ok nvquit
     ok
     ```

   - To store the changes, type:

     ```
     ok nvstore
     ok
     ```

9. Verify the contents of the nvramrc script you created in Step 7.

```
ok printenv nvramrc
nvramrc = probe-all
cd /sbus@1f,0/
6 encode-int " scsi-initiator-id" property
device-end
cd /sbus@1f,0/SUNW,fas@2,8800000
6 encode-int " scsi-initiator-id" property
device-end
install-console
banner
ok
```

10. Instruct the OpenBoot PROM Monitor to use the `nvramrc` script.

    ```
    ok setenv use-nvramrc? true
    use-nvramrc? = true
    ok
    ```

11. Without allowing the node to boot, power on the second node. If necessary, abort the system to continue with OpenBoot PROM Monitor tasks.

12. Verify that the `scsi-initiator-id` for the host adapter on the second node is set to 7.

    ```
    ok cd /sbus@1f,0/SUNW,fas@2,8800000
    ok .properties
    scsi-initiator-id        00000007
    ...
    ```

13. Continue with the Solaris OS, Sun Cluster software, and volume management software installation tasks.

For software installation procedures, refer to the *Sun™ Cluster 3.1 Installation Guide.*

Installing a Sun StorEdge D1000 Array

This section provides the procedure for an initial installation of a Sun StorEdge D1000 array.

Use this procedure to install a Sun StorEdge D1000 array in a cluster prior to installing the Solaris OS and Sun Cluster software. Perform this procedure with the procedures in the *Sun™ Cluster 3.1 Installation Guide* and your server hardware manual.

1. Ensure that each device in the SCSI chain has a unique SCSI address.

   The default SCSI address for host adapters is 7. Reserve SCSI address 7 for one host adapter in the SCSI chain. This procedure refers to the host adapter you choose for SCSI address 7 as the host adapter on the `second` node. To avoid conflicts, in Step 7 you change the `scsi-initiator-id` of the remaining host adapter in the SCSI chain to an available SCSI address. This procedure refers to the host adapter with an available SCSI address as the host adapter on the `first` node. SCSI address 6 is usually available.

Note – Even though a slot in the device might not be in use, you should avoid setting the `scsi-initiator-id` for the first node to the SCSI address for that disk slot. This precaution minimizes future complications if you install additional disk drives.

For more information, refer to the *OpenBoot™ 3.*x *Command Reference Manual* and the labels inside the storage device.

2. Install the host adapters in the node which will be connected to the array.

   For the procedure on installing host adapters, refer to the documentation that shipped with your nodes.

3. Connect the cables to the arrays, as shown in Figure B-2.

Make sure that the *entire* bus length connected to each array is less than 25 meters. This measurement includes the cables to both nodes, as well as the bus length internal to each array, node, and the host adapter.

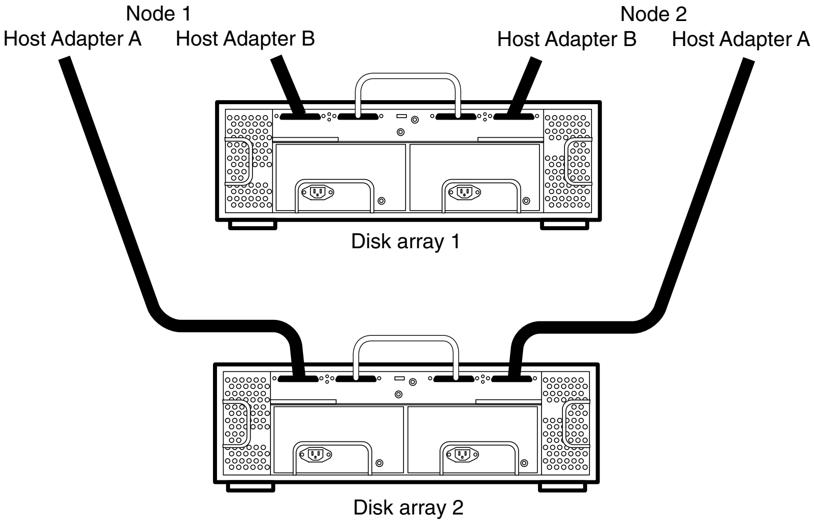


Figure B-2 Example of a Sun StorEdge D1000 Array Mirrored Pair

4. Connect the AC power cord for each array of the pair to a different power source.
5. Power on the first node and the arrays.
6. Find the paths to the host adapters.

   ```
   ok show-disks
   ```

   Identify and record the two controllers that will be connected to the storage devices and record these paths. Use this information to change the SCSI addresses of these controllers in the `nvramrc` script. Do not include the `/sd` directories in the device paths.
7. Edit the `nvramrc` script to change the `scsi-initiator-id` for the host adapter on the first node.

   For a list of `nvramrc` editor and `nvedit` keystroke commands, see "The `nvramrc` Editor and `nvedit` Keystroke Commands" on page B-11.

The following example sets the `scsi-initiator-id` to 6. The OpenBoot PROM Monitor prints the line numbers (`0:`, `1:`, and so on).

```
nvedit
0: probe-all
1: cd /sbus@1f,0/QLGC,isp@3,10000
2: 6 encode-int " scsi-initiator-id" property
3: device-end
4: cd /sbus@1f,0/
5: 6 encode-int " scsi-initiator-id" property
6: device-end
7: install-console
8: banner <Control-C>
ok
```

Note – Insert exactly one space after the first double quotation mark and before `scsi-initiator-id`.

8. Store or discard the changes.

   The edits are done on a temporary copy of the `nvramrc` script. You can continue to edit this copy without risk. After you complete your edits, save the changes. If you are not sure about the changes, discard them.

   - To store the changes, type:

     ```
     ok nvstore
     ok
     ```

   - To discard the changes, type:

     ```
     ok nvquit
     ok
     ```

9. Verify the contents of the `nvramrc` script you created in Step 7.

   ```
   ok printenv nvramrc
   nvramrc = probe-all
   cd /sbus@1f,0/QLGC,isp@3,10000
   6 encode-int " scsi-initiator-id" property
   device-end
   cd /sbus@1f,0/
   6 encode-int " scsi-initiator-id" property
   device-end
   install-console
   banner
   ok
   ```

10. Instruct the OpenBoot PROM Monitor to use the `nvramrc` script.

```
ok setenv use-nvramrc? true
use-nvramrc? = true
ok
```

11. Without allowing the node to boot, power on the second node. If necessary, abort the system to continue with OpenBoot PROM Monitor tasks.

12. Verify that the `scsi-initiator-id` for each host adapter on the second node is set to 7.

```
ok cd /sbus@1f,0/QLGC,isp@3,10000
ok .properties
scsi-initiator-id         00000007
differential
isp-fcode                 1.21 95/05/18
device_type               scsi
...
```

13. Continue with the Solaris OS, Sun Cluster software, and volume management software installation tasks.

For software installation procedures, refer to the *Sun™ Cluster 3.1 Installation Guide*.

The nvramrc Editor and nvedit Keystroke Commands

The OpenBoot PROM Monitor builds its own device tree based on the devices attached to the system when the boot sequence is invoked. The OpenBoot PROM Monitor has a set of default aliases for the commonly occurring devices in the system.

An nvramrc script contains a series of OpenBoot PROM commands that are executed during the boot sequence. The procedures in this guide assume that this script is empty. If your nvramrc script contains data, add the entries to the end of the script. To edit an nvramrc script or merge new lines in an nvramrc script, you must use nvedit editor and nvedit keystroke commands.

Table B-1 and Table B-2 on page B-12 list useful nvramrc editor and nvedit keystroke commands, respectively. For an entire list of nvedit editor and nvedit keystroke commands, refer to the *OpenBoot™ 3.x Command Reference Manual*.

Table B-1 The nvramrc Editor Commands

Command	Description
nvedit	Enters the nvramc editor. If the data remains in the temporary buffer from a previous nvedit session, resume editing previous contents. Otherwise, read the contents of nvramrc into the temporary buffer and begin editing it. This command works on a buffer, and you can save the contents of this buffer by using the nvstore command.
nvstore	Copies the contents of the temporary buffer to nvramrc, and discard the contents of the temporary buffer.
nvquit	Discards the contents of the temporary buffer, without writing it to nvramrc. Prompt for confirmation.
nvrecover	Attempts to recover the content of the nvramrc if the content was lost as a result of the execution of set-defaults, then enters the nvramrc editors as with nvedit. This command fails if nvedit is executed between the time the content of nvramrc was lost and the time the content of the nvramrc was executed.
nvrun	Executes the contents of the temporary buffer.

Table B-2 lists more useful nvedit commands.

Table B-2 The nvedit Keystroke Commands

Keystroke	Description
^A	Moves to the beginning of the line
^B	Moves backward one character
^C	Exits the script editor
^F	Moves forward one character
^K	Deletes until end of line
^L	Lists all lines
^N	Moves to the next line of the nvramrc editing buffer
^O	Inserts a new line at the cursor position and stay on the current line
^P	Moves to the previous line of the nvramrc editing buffer
^R	Replaces the current line
Delete	Deletes previous character
Return	Inserts a new line at the cursor position and advances to the next line

Appendix C

Role-Based Access Control Authorizations

This document describes the Sun Cluster 3.1 software RBAC authorizations required for the Sun Cluster 3.1 software status and maintenance commands.

RBAC Cluster Authorizations

SunPlex Manager and selected Sun Cluster commands and options that you issue on the command line use RBAC for authentication. Several RBAC rights profiles are included in Sun Cluster. You can assign these rights profiles to users or to roles to give them different levels of access to Sun Cluster. Sun provides the following rights profiles with Sun Cluster software.

Rights Profile	**Includes Authorizations**	**This Authorization Permits the Role Identity to**

Sun Cluster Commands	None, but includes a list of Sun Cluster commands that run with euid=0	Execute selected Sun Cluster commands that you use to configure and manage a cluster, including:
		scgdevs(1M)
		scswitch(1M) (selected options)
		scha_control(1HA)
		scha_resource_get(1HA)
		scha_resource_setstatus(1HA)
		scha_resourcegroup_get(1HA)
		scha_resourcetype_get(1HA)

Basic Solaris OS User	This existing Solaris OS rights profile contains Solaris OS authorizations, as well as:	Perform the same operations that the Basic Solaris OS User role identity can perform, as well as:
	solaris.cluster.device.read	Read information about device groups
	solaris.cluster.gui	Access SunPlex Manager

Rights Profile	Includes Authorizations	This Authorization Permits the Role Identity to
	`solaris.cluster .network.read`	Read information about IP Network Multipathing **Note –** This authorization does not apply to SunPlex Manager
	`solaris.cluster .node.read`	Read information about attributes of nodes
	`solaris.cluster .quorum.read`	Read information about quorum devices and the quorum state
	`solaris.cluster .resource.read`	Read information about resources and resource groups
	`solaris.cluster .system.read`	Read the status of the cluster
	`solaris.cluster .transport.read`	Read information about transports
Cluster Operation	`solaris.cluster .appinstall`	Install clustered applications
	`solaris.cluster .device.admin`	Perform administrative tasks on device group attributes
	`solaris.cluster .device.read`	Read information about device groups
	`solaris.cluster .gui`	Access SunPlex Manager
	`solaris.cluster .install`	Install clustering software **Note –** This authorization does not apply to SunPlex Manager

Rights Profile	Includes Authorizations	This Authorization Permits the Role Identity to
	`solaris.cluster .network.admin`	Perform administrative tasks on IP Network Multipathing attributes **Note –** This authorization does not apply to SunPlex Manager
	`solaris.cluster .network.read`	Read information about IP Network Multipathing **Note –** This authorization does not apply to SunPlex Manager
	`solaris.cluster .node.admin`	Perform administrative tasks on node attributes
	`solaris.cluster .node.read`	Read information about attributes of nodes
	`solaris.cluster .quorum.admin`	Perform administrative tasks on quorum devices and quorum state attributes
	`solaris.cluster .quorum.read`	Read information about quorum devices and the quorum state
	`solaris.cluster .resource.admin`	Perform administrative tasks on resource attributes and resource group attributes
	`solaris.cluster .resource.read`	Read information about resources and resource groups
	`solaris.cluster .system.admin`	Administer the system **Note –** This authorization does not apply to SunPlex Manager
	`solaris.cluster .system.read`	Read the status of the cluster

Rights Profile	Includes Authorizations	This Authorization Permits the Role Identity to
	`solaris.cluster .transport.admin`	Perform administrative tasks on transport attributes
	`solaris.cluster .transport.read`	Read information about transports
System Administrator	This existing Solaris OS rights profile contains the same authorizations that the Cluster Management profile contains.	Perform the same operations that the Cluster Management role identity can perform, in addition to other system administration operations
Cluster Management	This rights profile contains the same authorizations that the Cluster Operation profile contains, as well as the following authorizations:	Perform the same operations that the Cluster Operation role identity can perform, as well as:
	`solaris.cluster .device.modify`	Modify device group attributes
	`solaris.cluster .gui`	Access SunPlex Manager
	`solaris.cluster .network.modify`	Modify IP Network Multipathing attributes **Note –** This authorization does not apply to SunPlex Manager
	`solaris.cluster .node.modify`	Modify node attributes **Note –** This authorization does not apply to SunPlex Manager
	`solaris.cluster .quorum.modify`	Modify quorum devices and quorum state attributes
	`solaris.cluster .resource.modify`	Modify resource attributes and resource group attributes

Rights Profile	Includes Authorizations	This Authorization Permits the Role Identity to
	`solaris.cluster .system.modify`	Modify system attributes **Note –** This authorization does not apply to SunPlex Manager
	`solaris.cluster .transport.modify`	Modify transport attributes