



# Netra™ High Availability Suite 3.0 1/08 Foundation Services Overview

---

Sun Microsystems, Inc.  
[www.sun.com](http://www.sun.com)

Part No. 819-5240-13  
March 2008, Revision A

Submit comments about this document at: <http://www.sun.com/hwdocs/feedback>

Copyright 2008 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents>, and one or more additional patents or pending patent applications in the U.S. and in other countries.

This document and the product to which it pertains are distributed under licenses restricting their use, copying, distribution, and decompilation. No part of the product or of this document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Java, docs.sun.com, Netra, Solstice DiskSuite, Java Management Extensions, JMX, and Solaris are trademarks, registered trademarks, or service marks of Sun Microsystems, Inc. in the U.S. and other countries.

All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights—Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

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.

---

Copyright 2008 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, Californie 95054, Etats-Unis. Tous droits réservés.

Sun Microsystems, Inc. a les droits de propriété intellectuels relatants à la technologie qui est décrit dans ce document. En particulier, et sans la limitation, ces droits de propriété intellectuels peuvent inclure un ou plus des brevets américains énumérés à <http://www.sun.com/patents> et un ou les brevets plus supplémentaires ou les applications de brevet en attente dans les Etats-Unis et dans les autres pays.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a.

Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, Java, docs.sun.com, Netra, Solstice DiskSuite, Java Management Extensions, JMX, et Solaris sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays.

Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciées de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

LA DOCUMENTATION EST FOURNIE "EN L'ÉTAT" 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.



Adobe PostScript

# Contents

---

## **Preface xi**

### **1. Introduction to the Foundation Services 1**

What Are the Foundation Services? 1

High-Level View of the Foundation Services 2

### **2. Foundation Services Concepts 5**

Cluster Model 5

Peer Nodes and Nonpeer Nodes 7

Server (Master-Eligible) Nodes 7

Client (Master-Ineligible) Nodes 8

Reliability, Serviceability, Redundancy, and Availability 8

Reliability 8

Serviceability 9

Redundancy 9

Availability 9

Failover and Switchover 9

Data Management Policy 10

Service Models 11

Fault Management Models 12

	Fault Types	12
	Fault Detection	13
	Fault Reporting	13
	Fault Isolation	14
	Fault Recovery	14
<b>3.</b>	<b>Reliable Network Communication Service (CGTP)</b>	<b>15</b>
	Introduction to CGTP	15
	Data Transfer Using CGTP	16
	Sharing Physical Interfaces Between CGTP and IPMP Using VLAN	18
<b>4.</b>	<b>Cluster Network Addressing</b>	<b>19</b>
	Introduction to Cluster Network Addressing	19
	Cluster Network Addressing Scheme	20
	Node Address Triplets	21
	Floating Address Triplet	22
<b>5.</b>	<b>Cluster Membership Manager</b>	<b>27</b>
	Introduction to the Cluster Membership Manager	27
	Configuring the Cluster Membership	28
	Monitoring the Presence of Peer Nodes	29
	Interaction Between the nhprobed Daemon and the nhcmmmd Daemon	29
	Using the Direct Link to Prevent Split Brain Errors	30
	Multicast Transmission of Heartbeats	30
	Masterless Cluster	31
<b>6.</b>	<b>Reliable File Service</b>	<b>33</b>
	Reliable NFS	34
	Master Node IP Address Failover	34
	IP Replication	35

Replication During Normal Operations	35
Replication During Failover and Switchover	36
Data Area Used to Track Modifications to Data Blocks	38
Scorecard Bitmaps on the Solaris OS	39
DRBD Metadata on Linux	39
Synchronization Options	40
Delayed Synchronization	40
Reduced Duration of Disk Synchronization (Solaris OS Only)	40
Serialized Slice Synchronization	41
Sanity Check of Replicated Slices	41
Disk Partitioning	42
Solaris OS Standard Disk Partitioning	42
Solaris OS Virtual Disk Partitioning	43
Linux Standard Disk Partitioning	44
Linux Virtual Disk Partitioning	45
Logical Mirroring	46
Shared Disks	47
Standard Shared Disk Partitioning	49
<b>7. Reliable Boot Service</b>	<b>51</b>
Introduction to the Reliable Boot Service	51
Booting Diskless Nodes	52
Boot Policies for Diskless Nodes	53
DHCP Static	53
DHCP Client ID	53
<b>8. External Addressing</b>	<b>55</b>
Introduction to External Addressing	55
External Addressing Scheme	56

Floating External Addresses	56
Connecting Nonpeer Nodes Directly to a Cluster Network	57
Addressing a Shared Cluster Network and External Network	58
Connecting Nonpeer Nodes to the Cluster Through Additional Physical Interfaces	59
Addressing Physical Interfaces That Are Connected to an External Network	61
Connecting Nonpeer Nodes to the Cluster Network Through a Router	61
<b>9. Node State Manager</b>	<b>63</b>
Introduction to the Node State Manager	63
Writing Scripts	64
Output From User-Defined Scripts	64
<b>10. Daemon Monitor</b>	<b>65</b>
The nhpmd Daemon	65
Using the Node Management Agent With the Daemon Monitor	66
<b>11. Node Management Agent</b>	<b>67</b>
Introduction to the Node Management Agent	67
Monitoring Statistics With the NMA	68
Manipulating the Cluster With the NMA	71
Receiving Notifications With the NMA	71
<b>Index</b>	<b>73</b>

# Figures

---

FIGURE 1-1	Basic Foundation Services Cluster	2
FIGURE 1-2	Basic Foundation Services Cluster	3
FIGURE 2-1	Example of a Cluster Configuration With a Non-Peer Node Connected to the Cluster	6
FIGURE 3-1	CGTP Transfer of Data Packets From a Source Node to a Destination Node	17
FIGURE 3-2	CGTP Link Failure	18
FIGURE 4-1	Structure of an IPv4 Address on a Peer Node	20
FIGURE 4-2	Node Address Triplets	22
FIGURE 4-3	Example of the Floating Address Triplet of a Master Node and Vice-Master Node	24
FIGURE 4-4	Example of the Floating Address Triplet After Failover	25
FIGURE 6-1	Data Replication	36
FIGURE 6-2	Reliable File Service During Failover or Switchover	37
FIGURE 6-3	Restoration of the Synchronized State	38
FIGURE 6-4	One Partition of a Physical Disk Configured as a Virtual Disk	44
FIGURE 6-5	Defining Logical Volumes on Linux Using LVM	46
FIGURE 6-6	Data Replication Using Shared Disk	48
FIGURE 6-7	Reliable File Service During Failover or Switchover for Shared Disk	48
FIGURE 6-8	Restoration of the Synchronized State for Shared Disk	49
FIGURE 7-1	Request for Boot Broadcast From a Diskless Node	52
FIGURE 8-1	Example of a Nonpeer Node Connected Directly to a Cluster Network Using a Public IP Address Space	58

<b>FIGURE 8-2</b>	Example of a Nonpeer Node Connected to the Cluster Network Through Additional Physical Interfaces on Peer Nodes	60
<b>FIGURE 8-3</b>	Example of Nonpeer Nodes Connected to the Cluster Network Through a Router	62
<b>FIGURE 11-1</b>	Remote Access to the Cluster	68
<b>FIGURE 11-2</b>	Cascading Data From Peer Nodes to the Master Node	70



# Tables

---

TABLE 4-1	Example of Node Address Triplets for a Four-Node Cluster	21
TABLE 4-2	Example of Master Node Address Triplets	23
TABLE 6-1	Example Disk Partition for a Cluster of Server Nodes and Client (Diskless) Nodes	43
TABLE 6-2	Example Disk Partition for a Linux NHAS Cluster	45
TABLE 6-3	Example Disk Partition for a Cluster of Server Nodes and Client Nodes	50
TABLE 8-1	Example IP Addresses for a Master Node With a Logical Interface Configured for External Access	58
TABLE 8-2	Example IP Addresses for a Master Node With Three Physical Interfaces	61
TABLE 11-1	Statistics Collected by the NMA	69



# Preface

---

The *Netra High Availability Suite 3.0 1/08 Foundation Services Overview* describes the concepts and architecture around which the Netra™ High Availability (HA) Suite Foundation Services are built.

---

## Who Should Use This Book

This book is for system administrators who are maintaining a cluster running the Netra HA Suite software, or for system developers who are developing applications for a cluster running the Netra HA Suite software.

---

## How This Book Is Organized

- [Chapter 1](#) provides a brief introduction to the Netra HA Suite Foundation Services.
- [Chapter 2](#) describes the concepts on which the Foundation Services are built. You must be familiar with the concepts presented in this chapter before installing the Netra HA Suite software.
- [Chapter 3](#) provides information about the reliable network communication service provided by the Carrier Grade Transport Protocol (CGTP).
- [Chapter 4](#) provides an overview of cluster addressing.
- [Chapter 5](#) provides information about managing and configuring the Cluster Membership Manager (CMM).
- [Chapter 6](#) provides information about the Reliable File Service (RFS).

- [Chapter 7](#) provides information about the Reliable Boot Service (RBS).
- [Chapter 8](#) explains options for connecting nonpeer nodes to the cluster network using external addressing.
- [Chapter 9](#) describes how to use user-provided scripts to affect the reaction of the Node State Manager (NSM) to notifications it receives from the Cluster Membership Manager (CMM).
- [Chapter 10](#) describes how the Daemon Monitor is used to survey other process daemons.
- [Chapter 11](#) describes how the Node Management Agent (NMA) can be used to monitor and manipulate a cluster.

---

**Note** – This chapter refers to Internet Engineering Task Force Request for Comments (RFC) standards. For more information, see the complete text of RFC papers at: <http://www.ietf.org/>

---

## Shell Prompts

Shell	Prompt
C shell	<i>machine-name%</i>
C shell superuser	<i>machine-name#</i>
Bourne shell and Korn shell	\$
Bourne shell and Korn shell superuser	#

---

# Typographic Conventions

Typeface*	Meaning	Examples
AaBbCc123	The names of commands, files, and directories; onscreen computer output	Edit your <code>.login</code> file. Use <code>ls -a</code> to list all files. % You have mail.
<b>AaBbCc123</b>	What you type, when contrasted with onscreen computer output	% <b>su</b> Password:
<i>AaBbCc123</i>	Book titles, new words or terms, and words to be emphasized. Replace command-line variables with real names or values.	Read Chapter 6 in the <i>User's Guide</i> . These are called <i>class</i> options. To delete a file, type <code>rm filename</code> .

\* The settings on your browser might differ from these settings.

---

## Related Documentation

The following table lists the documentation for this product. The online documentation is available at:

<http://docs.sun.com/app/docs/prod/netra.ha30>

Application	Title	Part Number
Late-breaking news	<i>Netra High Availability Suite 3.0 1/08 Release Notes</i>	819-5249-14
Introduction to concepts	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Overview</i>	819-5240-13
Basic setup, supported hardware, and configurations	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Getting Started Guide</i>	819-5241-13
Automated installation methods	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Installation Guide</i>	819-5242-13
Detailed installation methods	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Manual Installation Guide for the Solaris OS</i>	819-5237-13
Cluster administration	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide</i>	819-5235-13

<b>Application</b>	<b>Title</b>	<b>Part Number</b>
Using the Cluster Membership Manager	<i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>	819-5236-13
Using the SAF CMM API	<i>Netra High Availability Suite 3.0 1/08 Foundation Services SA Forum Programming Guide</i>	819-5246-13
Using the Node Management Agent	<i>Netra High Availability Suite 3.0 1/08 Foundation Services NMA Programming Guide</i>	819-5239-13
Configuring outside the cluster using CGTP	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Standalone CGTP Guide</i>	819-5247-13
Man pages for Foundation Services features and APIs using the Solaris OS	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Solaris Reference Manual</i>	819-5244-13
Man pages for Foundation Services features and APIs using Linux	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Linux Reference Manual</i>	819-5245-12
Definitions and acronyms	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Glossary</i>	819-5238-13
Common problems	<i>Netra High Availability Suite 3.0 1/08 Foundation Services Troubleshooting Guide</i>	819-5248-13

## Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (<http://www.sun.com/documentation>)
- Support (<http://www.sun.com/support>)
- Training (<http://www.sun.com/training>)

## Third-Party Web Sites

Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

---

# Sun Welcomes Your Comments

Sun is interested in improving its documentation, and welcomes your comments and suggestions. You can submit your comments by going to:

<http://www.sun.com/hwdocs/feedback>

Please include the title and part number of your document with your feedback:

*Netra™ High Availability Suite 3.0 1/08 Foundation Services Overview*, part number 819-5240-13.





# Introduction to the Foundation Services

---

This chapter introduces you to the Netra High Availability (HA) Suite software Foundation Services, and includes the following sections:

- [“What Are the Foundation Services?” on page 1](#)
- [“High-Level View of the Foundation Services” on page 2](#)

---

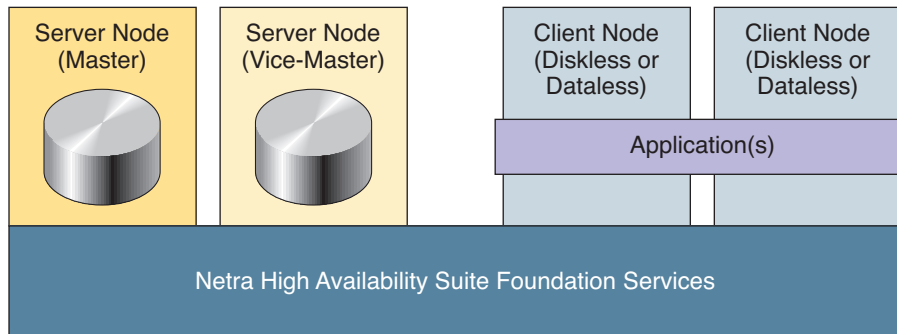
## What Are the Foundation Services?

The Netra HA Suite Foundation Services are a suite of highly available and distributed software services that run either on the Solaris™ Operating System (Solaris OS) or Carrier Grade Linux distributions. Refer to the *Netra High Availability Suite 3.0 1/08 Release Notes* for information about which Solaris OS releases and which Linux distributions are supported for use with the Netra HA Suite software.

The Foundation Services enable you to deploy applications in a continuous-availability environment. You can use the Netra HA Suite Foundation Services to create a highly available, dynamically scalable cluster of distributed nodes, and to augment existing highly available frameworks.

The following figure illustrates a basic Foundation Services cluster.

**FIGURE 1-1** Basic Foundation Services Cluster



The concepts of clusters, server nodes (master and vice-master nodes), and client nodes (diskless and dataless nodes) are described in [“Cluster Model” on page 5](#).

The Foundation Services are designed to do the following:

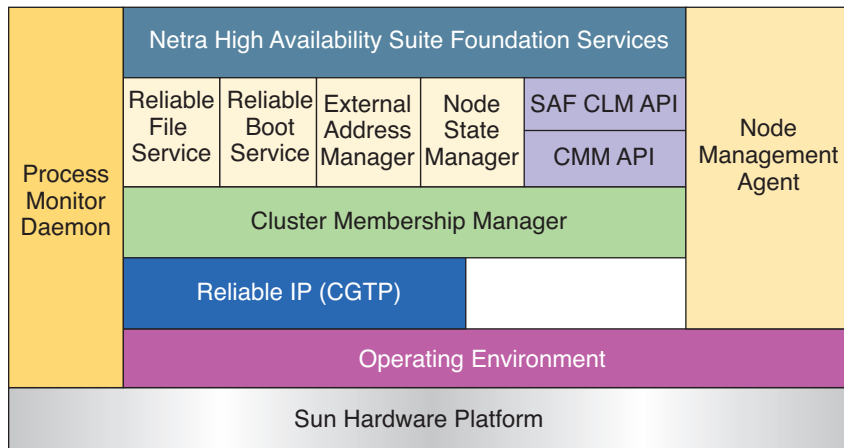
- Allow for hardware replacement, upgrade, and diagnostics without incurring system outage
- Provide highly available services (running on server nodes) to applications (in general, running on client nodes)
- Give HA-unaware applications a simple means for becoming HA aware

---

## High-Level View of the Foundation Services

The following figure shows a high-level view of the services and application programming interfaces (APIs) provided by the Netra HA Suite Foundation Services product.

**FIGURE 1-2** Basic Foundation Services Cluster



The Foundation Services offer the following services:

- A reliable network communication service provided by the Carrier Grade Transport Protocol (CGTP). CGTP limits the consequences of single network failure by duplicating the communication on a minimum of two links. For more information, see [Chapter 3](#).
- A Cluster Membership Manager (CMM) to provide a global view of the cluster. The Cluster Membership Manager determines which nodes are members of the cluster. It assigns the roles and attributes of nodes, detects the failure of nodes, and notifies clients of changes to the cluster. A heartbeat mechanism detects node failure. For more information, see [Chapter 5](#).
- A Reliable File Service (RFS) to ensure that data is accessible to applications, even in the event of hardware or software failure. Reliable File Service uses mounted file systems, IP replication of disk-based data or dual-hosted disks, and IP address failover of the master role. For more information, see [Chapter 6](#).
- A Reliable Boot Service (RBS) to ensure the boot of diskless nodes regardless of software or hardware failures. For more information, see [Chapter 7](#).

---

**Note** – Diskless nodes are not supported on the Linux OS, therefore, Reliable Boot Service is not supported on clusters running Linux.

---

- An External Address Manager (EAM) to allow external access to the cluster on highly available IPv4 or IPv6 addresses. For more information, see [Chapter 8](#).

---

**Note** – IPv6 addresses are not supported on Linux.

---

- A Node State Manager (NSM) with user-provided scripts executed to react to role changes, when the node becomes master or vice-master. For more information, see [Chapter 9](#).
- A Process Monitor Daemon (PMD) to survey Netra HA Suite daemons, some operating system daemons (on the Solaris OS only), and some companion product daemons. If any of the monitored daemons fail, the PMD initiates a recovery response. For more information, see [Chapter 10](#).
- A Node Management Agent (NMA) to monitor cluster statistics. The Node Management Agent can initiate a switch to the backup node, change some error recovery responses, and listen for notifications of some cluster events.

The Node Management Agent is compliant with the Java™ Management Extensions (JMX™) and based on the Java Dynamic Management Kit. For more information, see [Chapter 11](#).

---

**Note** – The Node Management Agent is supported only on clusters running the Solaris Operating System.

---

# Foundation Services Concepts

---

This chapter describes the concepts around which the Foundation Services are built. Ensure that you are familiar with the concepts described in this chapter before installing and using the Netra HA Suite software.

This chapter includes the following sections:

- [“Cluster Model” on page 5](#)
- [“Reliability, Serviceability, Redundancy, and Availability” on page 8](#)
- [“Service Models” on page 11](#)
- [“Fault Management Models” on page 12](#)

---

## Cluster Model

This section describes the cluster environment and the types of nodes in a cluster.

A cluster is a set of interconnected nodes (peer nodes) that collaborate through distributed services to provide highly available services. A cluster is made of 2 to 64 peer nodes, as follows:

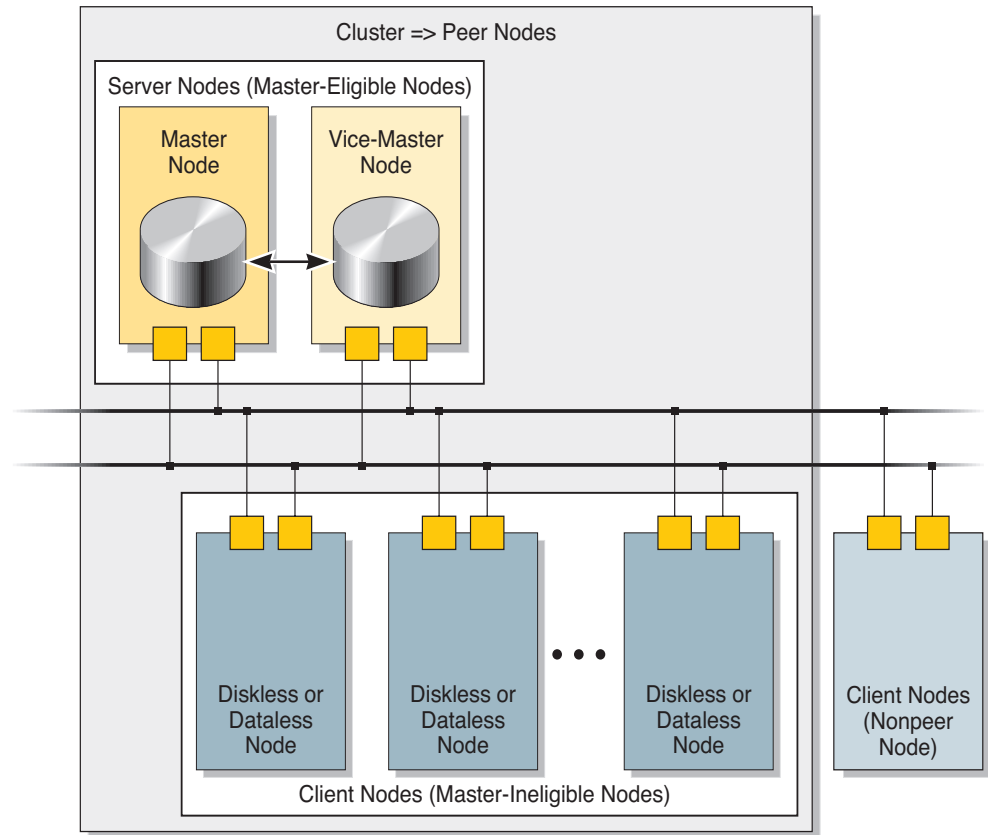
- Two mandatory peer nodes are server nodes, also called master-eligible nodes:
- A master node to coordinate the cluster membership information and provide highly available services to applications
- A vice-master node that backs up the master node
- Redundant data is shared between the two server nodes (master and vice-master nodes).
- Other peer nodes (if any) are client nodes, also called master-ineligible nodes. Client nodes can be diskless or dataless nodes. There can be a maximum of 62 client nodes.

- All peer nodes are connected to a redundant network between all nodes for reliable inter-node communication.

Non-peer nodes outside of the cluster can connect to the server nodes of a cluster.

The following figure is an example of a cluster configuration with non-peer nodes connected to the cluster.

**FIGURE 2-1** Example of a Cluster Configuration With a Non-Peer Node Connected to the Cluster



All types of nodes are described in the following sections.

## Peer Nodes and Nonpeer Nodes

Nodes that are configured as members of a cluster are called peer nodes. Peer nodes can run Netra HA Suite software and communicate with each other on the same network. Peer nodes can be server nodes or client nodes.

Nodes that are not configured as members of a cluster are called *nonpeer nodes*. A nonpeer node communicates with one or more peer nodes to access resources or services provided by the cluster. In [FIGURE 2-1](#), the nonpeer node is connected to both of the redundant network links. For information about the options for connecting nonpeer nodes to a cluster, see [Chapter 8](#).

## Server (Master-Eligible) Nodes

A cluster must contain two *server nodes* (master-eligible nodes). A server node is a peer node that can be elected as the master node or the vice-master node.

The *master node* is the node that coordinates the cluster membership information. The master node generates its view of the cluster configuration and communicates this view to the other peer nodes. Highly available services (for example, Reliable File Service and Reliable Boot Service ) run on only the master node. If an application is a client/server type of application, services provided by the application must run on a master node and must use the CMM API to get notifications about the cluster state. The application must also manage its own availability (whether or not NSM is being used).

The *vice-master node* backs up the master node. The vice-master node has a copy of all of the cluster management information that is on the master node. It can transparently take control of the cluster if required.

You must take care that any tasks you run on the vice-master node either have a very low load level, or are designed so that the tasks can be interrupted if the vice-master node needs to become the master node. The vice-master node must always be available to take over the master node's load if the current master node is no longer able to continue in the master role.

Each server node must be a *diskfull node*. A diskfull node has at least one disk on which information can be permanently stored. A server node must be configured as master-eligible at the time of installation and configuration. The master node and vice-master node are the only nodes that are configured as diskfull in a Netra HA Suite cluster.

## Client (Master-Ineligible) Nodes

A cluster contains two server (master-eligible) nodes, the master node and the vice-master node. All other peer nodes are *client nodes*. It is highly recommended, but not mandatory, that you run applications on only client nodes except when the application is a client/server type of application. In this case, the server part of the application must run on the master node in order to be made easily highly available.

In a Netra HA Suite cluster, client nodes are either *diskless nodes* or *dataless nodes*.

A diskless node either does not have a local disk or is configured not to use its local disk. Diskless nodes boot through the cluster network, using the master node as a boot server.

A dataless node has a local disk from which it boots, but it cannot store data that has to be redundant on its disk; this type of data must be stored on server (master-eligible) nodes. An application running on a dataless node accesses redundant data from server (master-eligible) nodes through the cluster network.

For examples of supported cluster configurations, see the *Netra High Availability Suite 3.0 1/08 Foundation Services Getting Started Guide*.

---

## Reliability, Serviceability, Redundancy, and Availability

This section defines the concepts of reliability, serviceability, redundancy, and availability. These concepts use the mechanisms of failover and switchover, as described in [“Failover and Switchover” on page 9](#).

### Reliability

*Reliability* is a measure of continuous system uptime. Netra HA Suite software provides distributed services and highly available services to increase the reliability of your system.



## Serviceability

*Serviceability* is the probability that a service can be restored within a specified period of time following a service failure. The Foundation Services increase the serviceability of applications through the highly available services they provide and the fast failover (or switchover) time used to transfer highly available services from one server node to another (in general, such transfers take place in less than five seconds).

## Redundancy

*Redundancy* increases the availability of a service by providing a backup to take over in the event of failure.

The Foundation Services provide an active/stand-by model for services running on server nodes (including services from the Foundation Services or those provided by an application, specifically, the server part of a client/server type of application). Because no application framework is currently provided by the Netra HA Suite software, applications must manage the redundancy model they want to apply themselves. The Cluster Membership Manager (CMM) APIs (proprietary or SA Forum-compliant) provide applications with information to help them implement their own redundancy models. The master node is backed up by the vice-master node. If the master node fails, there is a transparent transfer of services to the vice-master node. In the Foundation Services, the instance of the service running on the master node is the primary instance. The instance of the service running on the vice-master node is the secondary instance.

## Availability

*Availability* is the probability that a service is available for use at any given time. Availability is a function of system reliability and serviceability, supported by redundancy.

---

## Failover and Switchover

Failover and switchover are the mechanisms that ensure the high availability of a cluster.

*Failover* occurs if the master node fails, or if a vital service running on the master node fails. The services on the master node fail over to the vice-master node. The vice-master node has all of the necessary state information to take over from the master node. The vice-master node expects no cooperation or coordination from the failed master node.

*Switchover* is the planned transfer of services from the master node to the vice-master node. Switchover is orchestrated by the system or by an operator so that a node can be maintained without affecting system performance. Switchover is not linked to node failure. As in the case of a failover, the backup must have all of the necessary state information to take over at the moment of the switchover. Unlike failover, in switchover the master node can help the vice-master node by, for example, flushing caches for shared files.

Only the master node and vice-master node take part in failover and switchover. If a diskless node or dataless node fails, there is no failover, but all nodes that are part of the cluster (peer nodes) are made aware of the node failure through the Cluster Membership Manager (CMM) APIs (proprietary or SA Forum-compliant). If a diskless node or dataless node is the only node running an application, the application fails. If other diskless nodes or dataless nodes are running the application, the application will continue to run on these other nodes.

---

## Data Management Policy

Three data management policies are available in the Foundation Services. These policies determine how the cluster behaves when a failed vice-master node reboots in a cluster that has no master node. The policy you choose depends on the availability and data-integrity requirements of your cluster.

The data management policies are as follows:

---

Integrity	Ensures that the cluster uses the most up-to-date data. The vice-master node does not take the master role, but waits for the old master node to return to the cluster. This is the default data management policy.
Availability	Prioritizes the availability of services running on the cluster over data integrity. The vice-master node takes the master role when there is no master node in the cluster. This policy triggers a full synchronization when the old master node joins the cluster as the new vice-master node. This synchronization might result in the loss of any data written to the old master node while the vice-master node was down.
Adaptability	Prioritizes availability only if the master and vice-master disks are synchronized. Such disk synchronization would increase the level of data integrity. The vice-master checks the disk synchronization state. If the state indicates that the master and vice-master nodes are not synchronized, the vice-master node does not take the master role, but waits until the old master node returns to the cluster. If the state indicates that the master and vice-master disks are synchronized, the vice-master takes on the master role without waiting for the old master to rejoin the cluster.

---

Choose the data management policy by setting the value of the `Cluster.DataManagementPolicy` parameter in the `nhfs.conf` file. If you are installing the cluster with `nhinstall`, choose the data management policy by setting the value of `DATA_MGT_POLICY` in the `cluster_definition.conf` file. For more information, see the `nhfs.conf4` and `cluster_definition.conf4` man pages.

---

## Service Models

The Foundation Services provide two categories of service: highly available services and distributed services.

Highly available services run on the master node and vice-master node only. The Reliable Boot Service, Reliable File Service, External Address Manager (EAM), and Node State Manager (NSM) are highly available services. If the master node or one of these services on the master node fails, a failover occurs.

*Distributed services* are services that run on all peer nodes. The distributed services include the Cluster Membership Manager (CMM), Node Management Agent (NMA), and the Process Monitor Daemon (PMD). If a distributed service fails and cannot be restarted, the node running the service is removed from the cluster. If the node is the master node, a failover occurs.

---

## Fault Management Models

This section describes some of the faults that can occur in a cluster, and how those faults are managed.

### Fault Types

When one critical fault occurs, it is called a *single fault*. A single fault can be the failure of one server node, the failure of a service, or the failure of one of the redundant networks. After a single fault, the cluster continues to operate correctly but it is not highly available until the fault is repaired.

Two critical faults occurring that affect both parts of the redundant system is referred to as a *double fault*. A double fault can be the simultaneous failure of both server nodes, or the simultaneous failure of both redundant network links. Although many double faults can be detected, it might not be possible to recover from all double faults. Although rare, double faults can result in cluster failure.

Some faults can result in the election of two master nodes. This error scenario is called *split brain*. Split brain is usually caused by communication failure between server nodes. When the communication between the server nodes is restored, the last elected master node remains the master node. The other server node is elected as the vice-master node. Data shared between the two server (master-eligible) nodes must then be resynchronized. There is a risk of loss of data.

After a cluster is down (when both master-eligible nodes are down), if the cluster is restarted and the newly elected master node was previously the vice-master node, there is a risk that the system will experience amnesia. This happens if the newly elected master node thinks its data is synchronized with the data on the node that

was previously the master node, but it actually is not. When the node that was previously the master node comes back as the new vice-master node, the most recently updated data on this node will be lost.

## Fault Detection

Fault detection is critical for a cluster running highly available applications. The Foundation Services have the following fault detection mechanisms:

- The Cluster Membership Manager (CMM) detects the failure of peer nodes. It notifies the other peer nodes of the failure. For information about CMM, see [Chapter 5](#).
- The Process Monitor Daemon (PMD) surveys Foundation Services daemons, some operating system daemons (on the Solaris OS only), and some companion products daemons. When a critical service or a descendent of a critical service fails, the PMD detects the failure and triggers a recovery response. For information about the PMD, see [Chapter 10](#).
- External Address Manager (EAM) supervises the external address links to the master node. When EAM detects that links are broken, it triggers a switchover to the vice-master node (if allowed).

## Fault Reporting

Errors that indicate potential failure are reported so that you can understand the sequence of events that have led to the problem. The Foundation Services have the following fault-reporting mechanisms:

- All error messages are sent to system log files. For information about how to configure log files, see the *Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide*.
- The Cluster Membership Manager on the master node notifies clients when a node fails, a failover occurs, or the cluster membership changes. Clients can be subscribed system services or applications.
- The Node Management Agent can be used to develop applications that retrieve statistics about services provided by the Foundation Services. These applications can be used to detect faults or diminished levels of service on your system. For further information on how to collect and manage node and cluster statistics, see the *Netra High Availability Suite 3.0 1/08 Foundation Services NMA Programming Guide*.

## Fault Isolation

When a fault occurs in the cluster, the node on which the fault occurred is isolated. The Cluster Membership Manager ensures that the failed node cannot communicate with the other peer nodes.

## Fault Recovery

Typical fault recovery scenarios include the following behaviors:

- If a process monitored by the Process Monitor Daemon fails, the recovery depends on the policy of recovery that is attached to a particular process. A recovery can involve a process restart (with a maximum number of trials), a node reboot (implying a failover if the node on which the process failed is the master node), or a simple message logging. In all cases, the cluster is still up and running.
- If a highly available or distributed service fails, this triggers a reboot of the node on which the service failed. If the node is the master, the node fails over; if the node is the vice-master or a client node, action is taken by the applications running on the node. The cluster remains up and running.
- If a link of the cluster network fails, an error message is logged, and an event is sent to the NMA. The cluster remains up and running.
- If an external link to the master node is down, IPMP (on the Solaris OS) or bonding (on Linux), if they are configured, can move all of the external traffic to another link, assuming there are at least two external links giving access to the master node and no single point of failure in a cluster. There will be no impact on the cluster. If all external links to the master node from the same IPMP group (on the Solaris OS) or from the Linux bonding driver (on Linux) are down, the EAM will trigger a switchover. The cluster remains up and running.
- If there is a client node failure, the node is isolated and rebooted. It is up to the applications running on this client node to implement the recovery procedure, as no application management framework is provided by the Foundation Services.

Failed nodes are often repaired by rebooting the system. Overload errors are often repaired by waiting for an acceptable delay and then rebooting or restarting the failed service. The Foundation Services are designed so that individual nodes can be shut down and restarted independently, reducing the impact of errors. After failover, the master node and vice-master node are synchronized so that the repaired vice-master node can rejoin the cluster in its current state.

## Reliable Network Communication Service (CGTP)

---

The reliable network communication service is provided with the implementation of a reliable IP transport mechanism in each node. For information about the reliable IP transport mechanism provided by the Carrier Grade Transport Protocol (CGTP), see the following sections:

- [“Introduction to CGTP” on page 15](#)
- [“Data Transfer Using CGTP” on page 16](#)

---

### Introduction to CGTP

The Netra High Availability (HA) Suite software Foundation Services use a reliable IP transport mechanism provided by CGTP. CGTP is based on transparent multirouting using redundant routes.

In the Foundation Services, each peer node is connected by two high-speed Ethernet networks. When data is sent from a source node to a destination node, it is sent along both Ethernet networks. If data on one network fails to reach the destination node, the data on the other network is still able to reach the destination node. To ensure symmetry in the redundant routes, the Ethernet networks must have equal bandwidth and latency. To prevent single points of failure, the Ethernet networks must not share switching equipment or communication links.

CGTP is configured by the Cluster Membership Manager on peer nodes. CGTP installed on nonpeer nodes is called *standalone CGTP*. Standalone CGTP must be configured manually. For information about installing and configuring CGTP on nonpeer nodes, see the *Netra High Availability Suite 3.0 1/08 Foundation Services Standalone CGTP Guide*.

You can configure your cluster without CGTP. When CGTP is not used, each peer node is connected by one Ethernet network only. This configuration introduces a single point of failure. If the Ethernet network fails, there is no backup network.

---

## Data Transfer Using CGTP

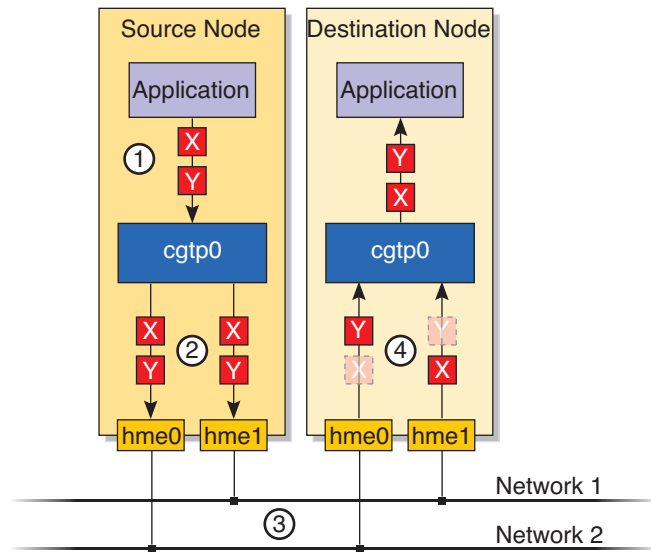
This section describes how CGTP transfers data from a source node to a destination node.

CGTP adds a CGTP source address and CGTP destination address to the header of a data packet on a source node, creating an *IP data packet*. The header contains all the information necessary to uniquely identify an IP data packet.

The Cluster Membership Manager defines routing tables on the source node and destination node. CGTP duplicates the IP data packet, and, using the routing tables, sends one copy of each IP data packet on each of the Ethernet networks. [FIGURE 3-1](#) illustrates the transfer of data packets from a source node to a destination node, using CGTP.



**FIGURE 3-1** CGTP Transfer of Data Packets From a Source Node to a Destination Node

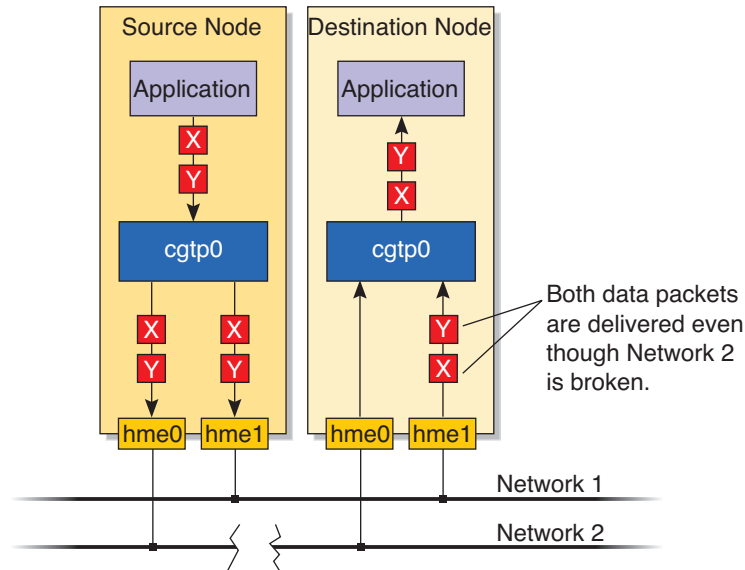


- ① Data packets are generated.
- ② Data packets are duplicated.
- ③ Data packets are sent on redundant paths.
- ④ First packet is transmitted, the second packet is discarded.

When the first IP data packet reaches its destination address, CGTP consults the filtering table on the destination node. CGTP verifies that the destination node has not already received the IP data packet. If it has not, CGTP sends the IP data packet to the higher protocols for processing. When the second incoming packet is detected, CGTP identifies it as a duplicate and filters it out. CGTP supports packet filtering on IPv4 and IPv6.

FIGURE 3-2 illustrates how CGTP is able to deliver the data packets when one of the redundant networks fails.

**FIGURE 3-2** CGTP Link Failure



If one link fails, the data packet sent on the other network path can still reach the destination address. However, until Network 2 is repaired, the system is not HA.

## Sharing Physical Interfaces Between CGTP and IPMP Using VLAN

On the Solaris OS, CGTP and IPMP can share physical links only when CGTP is used over a virtual local area network (VLAN). This requires a specific topology. For more information, refer to the *Netra High Availability Suite 3.0 1/08 Foundation Services Manual Installation Guide for the Solaris OS*.

On the Linux OS, CGTP cannot share interfaces with a Linux bonding driver, therefore, the available solutions are as follows:

- Two interfaces for CGTP with two or more interfaces for bonding
- Two interfaces for CGTP and the floating addresses on them without bonding

---

**Note** – This functionality is specific to the Solaris OS and is not supported on Linux.

---

## Cluster Network Addressing

---

For a description of the type of addressing that can be used within the private network of a Netra HA Suite Foundation services cluster, see the following sections:

- [“Introduction to Cluster Network Addressing” on page 19](#)
- [“Cluster Network Addressing Scheme” on page 20](#)
- [“Node Address Triplets” on page 21](#)
- [“Floating Address Triplet” on page 22](#)

---

### Introduction to Cluster Network Addressing

Peer nodes communicate with each other over a private network called the *cluster network*.

The addressing scheme used by the cluster network is classless. The IP addresses of peer nodes can be in a private network or a public network. It is advantageous to configure a cluster in a private network for the following reasons:

- There is no need to allocate addresses from a public address space for each cluster.
- External data can be added into or taken out of the cluster network in a controlled way, making the cluster network more secure.
- The cluster network cannot be overloaded by external traffic.
- The redundant network symmetry is protected.

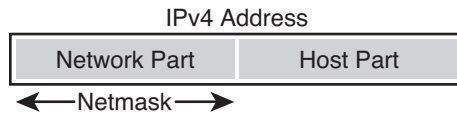
---

# Cluster Network Addressing Scheme

All peer nodes are assigned IPv4 addresses. The address is split into a *network part* and a *host part*. The dividing point between the network part and the host part is defined by the *netmask*.

The following figure illustrates the structure of an IPv4 address on a peer node.

**FIGURE 4-1** Structure of an IPv4 Address on a Peer Node



The network part of an IP address represents the identity of the network to which a peer node is connected. The host part of an IP address represents the `nodeid` of the peer node. The `nodeid` is the decimal equivalent of the host part of the IP address.

In a class B addressing scheme, the value of the netmask is `ffff0000`. The network part of an IP address is 16 bits long, and the host part of an IP address is 16 bits long.

In the class C addressing scheme, the value of the netmask is `ffffff00`. The network part of an IP address is 24 bits long, and the host part of an IP address is 8 bits long. By default, a class C address on a Netra HA Suite cluster has the following format:

```
10.domainid.interfaceid.nodeid
```

Netra HA Suite software can use a classless addressing scheme. The value of the netmask is not restricted to any address class.

The following values of the host part of the IP address are reserved:

- Value 0 is reserved for the identification of the network part of the IP address.
- Value 1 is reserved for the floating address triplet. For information about the floating address triplet, see [“Floating Address Triplet” on page 22](#).
- Value  $2^n - 1$  is reserved for the broadcast address. The parameter  $n$  is the number of bits in the host part of the IP address.

# Node Address Triplets

A *node address triplet* is assigned to each peer node in a cluster. The node address triplet consists of three IP addresses:

- An IP address for each of the two physical interfaces, *NIC0* and *NIC1*
- An IP address for the virtual interface, called the *CGTP address*

The CGTP address is used for IP routing. It hides the multirouting capability of CGTP. The CGTP address presents the redundant network as a single network interface. Applications can use the CGTP address to communicate with the master node. The CGTP address supports unicast, broadcast, and multicast transmissions. For more information about CGTP, see [Chapter 3](#).

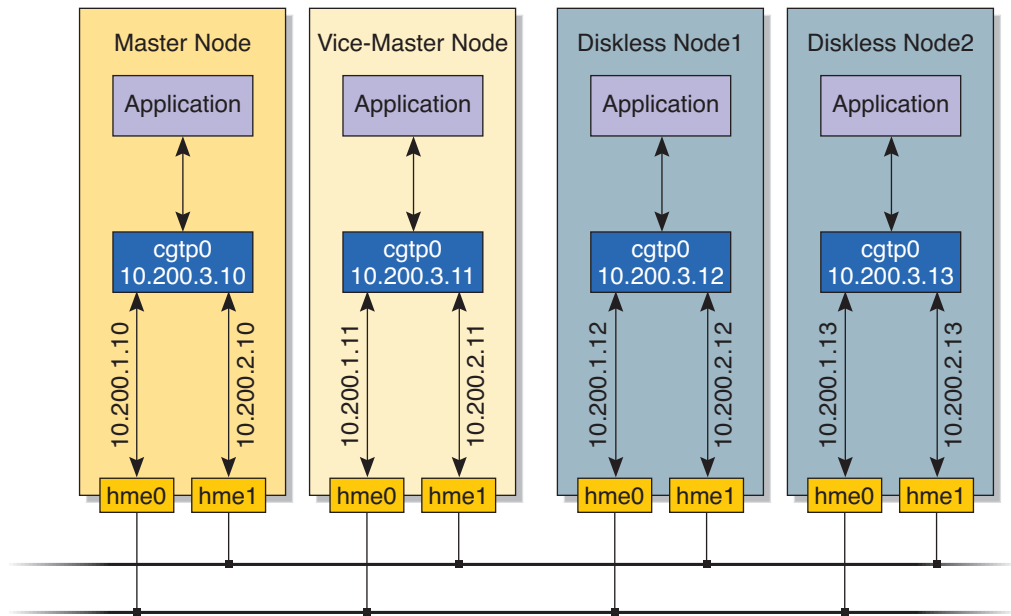
The following table shows an example of node address triplets for a cluster containing two server (master and vice-master) nodes and two client (diskless) nodes. The cluster uses a default class C addressing scheme.

**TABLE 4-1** Example of Node Address Triplets for a Four-Node Cluster

Node Name	Address for Physical Interface <i>hme0</i>	Address for Physical Interface <i>hme1</i>	Address for Virtual Interface <i>cgtp0</i>
Master Node	10.200.1.10	10.200.2.10	10.200.3.10
Vice-Master Node	10.200.1.11	10.200.2.11	10.200.3.11
Diskless Node 1	10.200.1.12	10.200.2.12	10.200.3.12
Diskless Node 2	10.200.1.13	10.200.2.13	10.200.3.13

The following figure shows the address triplets of the preceding cluster.

**FIGURE 4-2** Node Address Triplets



## Floating Address Triplet

In addition to a node address triplet, the master node and vice-master node have an address triplet called the *floating address triplet*. The floating address triplet is activated on the master node only. If the master node fails over or is switched over, the floating address triplet is activated on the new master node. It is deactivated on the old master node.

Diskless nodes and dataless nodes access services and data on the master node, through the floating address triplet. Because the floating address triplet is always up on the master node, the diskless nodes and dataless nodes can access the master node even after a failover or switchover.

The floating address triplet has a *logical address*. A logical address is an address that is assigned to a physical interface or virtual interface. A logical address for an hme0 or cgtp0 interface has the format hme0:x or cgtp0:x.

The following table shows an example of the node address triplet and floating address triplet of a master node. The cluster is using a default class C addressing scheme.

**TABLE 4-2** Example of Master Node Address Triplets

Triplet Type	Address Type	Interface	Address Example
Node Address Triplet	Physical	hme0	10.200.1.10
	Physical	hme1	10.200.2.10
	Virtual Physical	cgtp0	10.200.3.10
Floating Address Triplet	Logical	hme0:1	10.200.1.1
	Logical	hme1:1	10.200.2.1
	Virtual Logical	cgtp0:1	10.200.3.1

FIGURE 4-3 shows the node address triplet and floating address triplet of the master node and vice-master node. The diskless nodes are doing a Network File System (NFS) mount onto the master node through the floating address triplet. The floating address triplet of the vice-master node is crossed out because it is down.

**FIGURE 4-3** Example of the Floating Address Triplet of a Master Node and Vice-Master Node

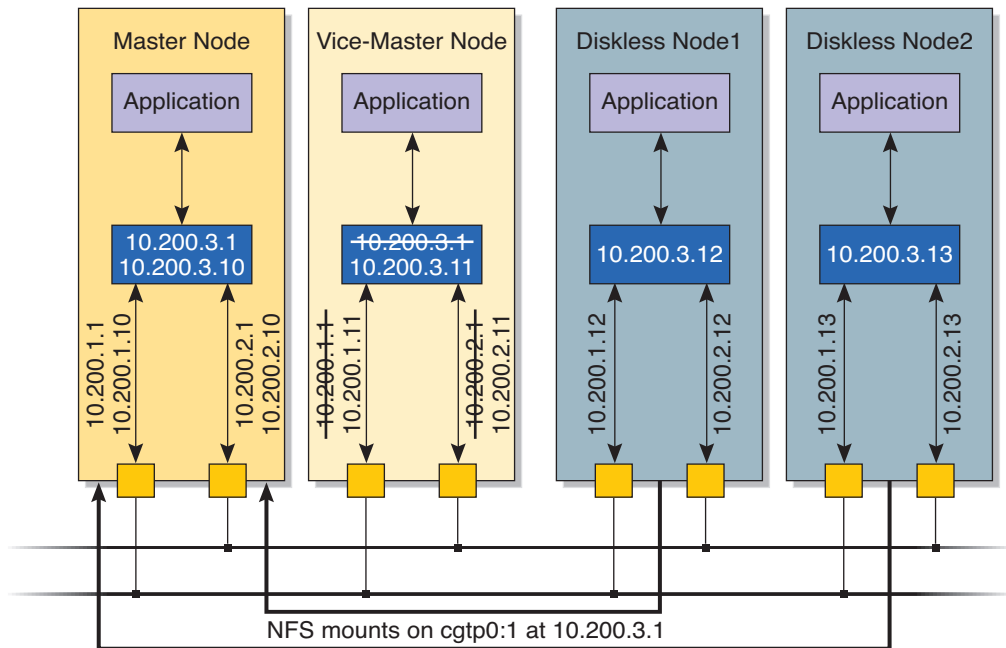
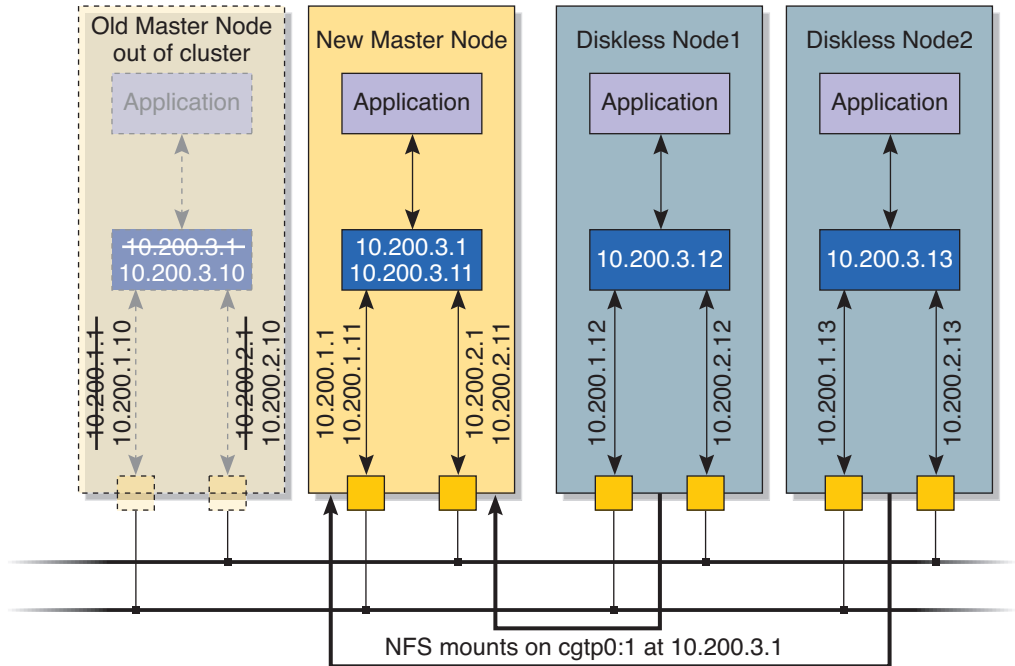


FIGURE 4-4 shows the cluster in FIGURE 4-3 after a failover. The diskless nodes are still accessing their mounted partitions through the floating address triplet and on the new master node.

**FIGURE 4-4** Example of the Floating Address Triplet After Failover









# Cluster Membership Manager

---

For information about how the cluster membership is managed and configured, and how the presence of peer nodes is monitored, see the following sections:

- [“Introduction to the Cluster Membership Manager” on page 27](#)
- [“Configuring the Cluster Membership” on page 28](#)
- [“Monitoring the Presence of Peer Nodes” on page 29](#)
- [“Masterless Cluster” on page 31](#)

---

## Introduction to the Cluster Membership Manager

The Cluster Membership Manager (CMM) is implemented by the `nhcmmmd` daemon. There is a `nhcmmmd` daemon on each peer node.

The `nhcmmmd` daemon on the master node has the current view of the cluster configuration. It communicates its view to the `nhcmmmd` daemons on the other peer nodes. The `nhcmmmd` daemon on the master node determines which nodes are members of the cluster, and assigns roles and attributes to the nodes. It detects the failure of nodes and configures routes for reliable transport.

The `nhcmmmd` daemon on the vice-master node monitors the status of the master node. If the master node fails, the vice-master node is able to take over as the master node.

The `nhcmmmd` daemons on the client nodes do not communicate with one another. Each `nhcmmmd` daemon exports two APIs to do the following:

- Notify clients of changes to the cluster
- Notify services and applications when the cluster membership or master changes

Notification messages describe the change and the *nodeid* of the affected node. Clients can use notifications to maintain an accurate view of the peer nodes in the cluster.

For further information about the `nhcmmd` daemon, see the `nhcmmd1M` man page.

You can use the CMM API to write applications that manage peer nodes or that register clients to receive notifications. For further information about writing applications that use the CMM API, see the *Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide*.

The standard SA Forum CLM API can only be used to retrieve membership information about the cluster nodes, and to receive notifications about membership changes. For more information, see the *Netra High Availability Suite 3.0 1/08 Foundation Services SA Forum Programming Guide*.

---

## Configuring the Cluster Membership

Cluster membership information is stored in the configuration files `cluster_nodes_table` and `nhfs.conf`.

At cluster startup, the cluster membership is configured as follows:

1. Both of the server nodes retrieve the list of peer nodes and their attributes from the `cluster_nodes_table`, and configuration information from `nhfs.conf`. All other peer nodes retrieve configuration information from `nhfs.conf`.
2. The `nhcmmd` daemon on the master node uses the list of nodes and their attributes to generate its view of the cluster configuration. It communicates this view to the `nhcmmd` daemons on the other peer nodes, including the vice-master node.
3. Using the master node view of the cluster, the `nhcmmd` daemon on the vice-master node updates its local `cluster_nodes_table`.

The `nhcmmd` daemon on the master node updates its `cluster_nodes_table` and its view of the cluster configuration when a peer node is added, removed, or disqualified. The `nhcmmd` daemon on the master node communicates the updated view to the `nhcmmd` daemons on the other peer nodes. The vice-master node uses this view to update its local `cluster_nodes_table`. In this way, the master node and vice-master node always have an up-to-date view of the cluster.

---

# Monitoring the Presence of Peer Nodes

Each peer node runs a daemon called `nhprobed` that periodically sends a heartbeat in the form of an IP packet. Heartbeats are sent through each of the two physical interfaces of each peer node. When a heartbeat is detected through a physical interface, it indicates that the node is reachable and that the physical interface is alive. If a heartbeat is not detected for a period of time exceeding the detection delay, the physical interface is considered to have failed. If both of the node's physical interfaces fail, the node itself is considered to have failed.

Heartbeats from each peer node are sent to a multicast group on the cluster network. Only the master node listens to heartbeats coming from the vice-master node and client nodes. Only the vice-master node listens to heartbeats coming from the master node. For more information, see [“Interaction Between the `nhprobed` Daemon and the `nhcmmnd` Daemon” on page 29](#). For more information about the `nhprobed` daemon, see the `nhprobed1M` man page.

## Interaction Between the `nhprobed` Daemon and the `nhcmmnd` Daemon

On the server nodes, the `nhprobed` daemon receives a list of nodes from the `nhcmmnd` daemon. The `nhprobed` daemon monitors the heartbeats of the nodes on the list. On the master node, the list contains all of the client nodes and the vice-master node. On the vice-master node, the list contains the master node only.

On the server nodes, the `nhprobed` daemon notifies the `nhcmmnd` daemon when, for any node on its list, any of the following events occur:

- One link becomes available, indicating that the node is accessible through the link.
- One link becomes unavailable, indicating that the node is not accessible through the link.
- The node becomes available, indicating that the first link to the node becomes available.
- The node becomes unavailable, indicating that the last available link to the node becomes unavailable.

When a node other than the master node becomes unavailable, the master node eliminates the node from the cluster. The master node uses the TCP abort facility to close communication to the node. When the master node becomes unavailable, a failover is provoked.

# Using the Direct Link to Prevent Split Brain Errors

Split brain is an error scenario in which the cluster has two master nodes. A direct communication link between the server nodes prevents the occurrence of split brain when the communication over the cluster network between the master node and vice-master node fails.

As described in “[Monitoring the Presence of Peer Nodes](#)” on page 29, the `nhprobed` daemon on the vice-master node monitors the presence of the master node. If the `nhprobed` daemon on the vice-master node fails to detect the master node, the master node itself or the communication to the master node has failed. If this happens, the vice-master node uses the direct link to try to contact the master node.

- If the vice-master node does not receive a reply from the master node by using the direct link, it is assumed that the master node has failed. The vice-master node becomes the master node.
- If the vice-master node receives a reply from the master node by using the direct link, it is assumed that the communication to the master node has failed but the master node is alive. The vice-master node is rebooted.

The Node Management Agent can monitor the following statistics on the direct link:

- The number of times that the vice-master node has requested to become the master node.
- The state of the direct communication link. The state can be *up* or *down*.

For information about how to connect the direct link between the server nodes, see the *Netra High Availability Suite 3.0 1/08 Foundation Services Manual Installation Guide for the Solaris OS*.

## Multicast Transmission of Heartbeats

Probe heartbeats are multicast. Each cluster on a local area network (LAN) is assigned to a different multicast group, and each network interface card (NIC) on a node is assigned to a different multicast group. For example, NICs connected to an `hme0` Ethernet network are assigned to one multicast group, and NICs connected to an `hme1` Ethernet network are assigned to another multicast group.

A heartbeat sent from one multicast group cannot be detected by another multicast group. Therefore, heartbeats sent from one cluster cannot be detected by another cluster on the same LAN. Similarly, for a cross-switched topology, heartbeats sent from one Ethernet network cannot be detected on another Ethernet network.

Multicast addresses are 32-bit. The lower 28 bits of the multicast address represent the multicast group. The multicast address is broken into the following parts:

- Bits 28 to 31 are fixed.

- Bits 23 to 27 identify Netra HA Suite. For the Foundation Services, bits 23 to 27 are always set to 10100.
- Bits 8 to 22 identify the cluster. The value for a given cluster is specified in the `nhfs.conf` file by the `Node.DomainId` parameter.
- Bits 0 to 7 identify the NIC. The value for a given NIC is specified in the `nhfs.conf` file by the `Node.NIC0` and `Node.NIC1` parameters.

When you are defining multicast groups for applications, follow these recommendations:

- Bits 23 to 27 of the multicast address must not have the value 10100. This value is reserved for Netra HA Suite.
- Bits 0 to 22 of the multicast address should not have the same value as any of your Netra HA Suite clusters.

When a message is sent, the IP stack uses the lower 23 bits of the multicast address to define the destination media access control (MAC) address. If several multicast addresses have the same value for the lower 23 bits, even if they have different values for the upper five bits, the addresses must be filtered at the IP level. The IP filtering would create a corresponding reduction in performance.

- Because of the way that `hme` and `le` interfaces filter multicast packets, one in four clusters share the same multicast filter. To reduce the need to filter at the IP level, clusters in the same LAN should have sequential cluster identities.

---

## Masterless Cluster

In normal usage, a cluster should contain a master node and a vice-master node. It can also contain diskless and dataless nodes. If the cluster does not have a master node, there is a risk of data loss because it is the master node that holds the most up-to-date view of the cluster. However, in some cluster usage, you might want to reduce the possible downtime for services running on client nodes. If this is the case, you can permit the diskless and dataless nodes to stay up even when there is no master node in the cluster by enabling the Masterless Cluster feature. By default, this feature is disabled, and diskless and dataless nodes reboot if there is no master node in the cluster for more than a few minutes.

If you enable this feature, you must ensure that the diskless and dataless nodes can handle the situation where they would no longer be able to access the files exported by the master node.

Activate this feature by setting the `CMM.Masterloss.Detection` parameter in the `nhfs.conf` file. If you are installing the cluster with `nhinstall`, enable this feature by setting `MASTER_LOSS_DETECTION` in the `cluster_definition.conf` file. For more information, see the `nhfs.conf(4)` and `cluster_definition.conf(4)` man pages.

You can also configure the amount of time the vice-master node will wait before taking over when it detects a stale cluster situation. To do this, define the `CMM.Masterloss.Timeout` parameter in the `nhfs.conf` file. If you are installing the cluster using `nhinstall`, define `MASTER_LOSS_TIMEOUT` in the `cluster_definition.conf` file. For more information, refer to the `nhfs.conf(4)` and `cluster_definition.conf(4)` man pages.



## Reliable File Service

---

The Reliable File Service provides a method by which customer applications can access highly available data on the master node disks. This service is one of the highly available services that the Netra HA Suite Foundation Services provides.

The Reliable File Service includes the following features:

- A reliable file system that customer applications access through NFS mount points (Reliable NFS). When using Reliable NFS, customer applications running on peer or non-peer nodes can access highly available data on the master node disks.
- IP replication of disk-based data from the master node to the vice-master node.
- Data sharing between the master node and vice-master node through shared disks. This functionality is not yet available on Linux.

IP replication and shared disks are two different methods for sharing data between a master node and a vice-master node and, as a consequence, for making data highly available. Only one of these two mechanisms may be installed at a time for use on a given cluster.

This chapter contains the following sections:

- [“Reliable NFS” on page 34](#)
- [“IP Replication” on page 35](#)
- [“Shared Disks” on page 47](#)

---

# Reliable NFS

The Reliable NFS is implemented by the `nhcrfsd` daemon. The `nhcrfsd` daemon runs on the master node and vice-master node. It implements the failover or switchover from the master node to the vice-master node. If the master node fails, the vice-master node becomes master and the Reliable NFS server on the new master node becomes active.

The `nhcrfsd` daemon responds to changes in the cluster state as it receives notifications from the Cluster Membership Manager. For more information about the Cluster Membership Manager, see [Chapter 5](#). The Reliable NFS daemon is monitored by the Daemon Monitor, `nhpmd`. For more information about the Daemon Monitor, see [Chapter 10](#).

If the impact on performance is acceptable, do not use data and attribute caches when writing to shared file systems. If it is necessary to use data caches to improve performance, ensure that your applications minimize the risk of using inconsistent data. For guidelines on how to use data and attribute caches when writing to shared file systems, see “Using Data Caches in Shared File Systems” in the *Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide*.

For reference information about network tunable parameters and the Solaris kernel, see the *Solaris Tunable Parameters Reference Manual* for your version of the Solaris Operating System. Information about tunable parameters in the Linux kernel can be obtained from the provider of the actual Linux distribution.

## Master Node IP Address Failover

For a failover to be transparent to customer applications running on peer or non-peer nodes, the following must be true:

- NFS mount points on the master node disks, which customer applications will use to access to highly available data, must be mounted on the floating address of the master node.
- The floating address must always be configured and active on the node that holds the master role, even after failover or switchover. The Reliable NFS daemon takes care of configuring and unconfiguring this address depending on the role of the node (master or vice-master) on which it runs.

For more information about the floating address of the master node, see [“Floating Address Triplet” on page 22](#).

---

# IP Replication

IP replication is one of two mechanisms provided by the Reliable File Service for sharing data between a master node and a vice-master node. It is also called a “shared nothing” mechanism.

## Replication During Normal Operations

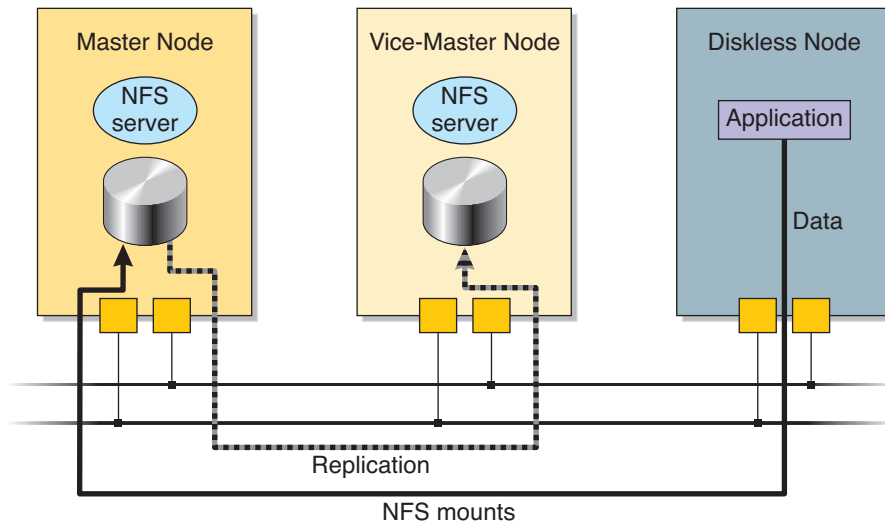
*Replication* is the act of copying data from the master node to the vice-master node. Through replication, the vice-master node has an up-to-date copy of the data on the master node. Replication enables the vice-master node to take over the master role at any time, transparently. After replication, the master node disk and vice-master node disk are *synchronized*, that is, the replicated partitions contain exactly the same data.

Replication occurs at the following times:

- When the master node and vice-master node are running, and data on the master node disk is changed
- After startup, to replicate the software installed on the replicated partitions
- After a failover or switchover

The following figure illustrates a client node (diskless or dataless) writing data to the master node, and that data being replicated to the vice-master node.

**FIGURE 6-1** Data Replication



## Replication During Failover and Switchover

During failover or switchover, the master node goes out of service for a time before being re-established as the vice-master node. During this time, changes that are made to the new master node disk cannot be replicated to the vice-master node. Consequently, the cluster becomes unsynchronized.

While the vice-master node is out of service, data continues to be updated on the master node disk, and the modified data blocks are identified in a specific data area.

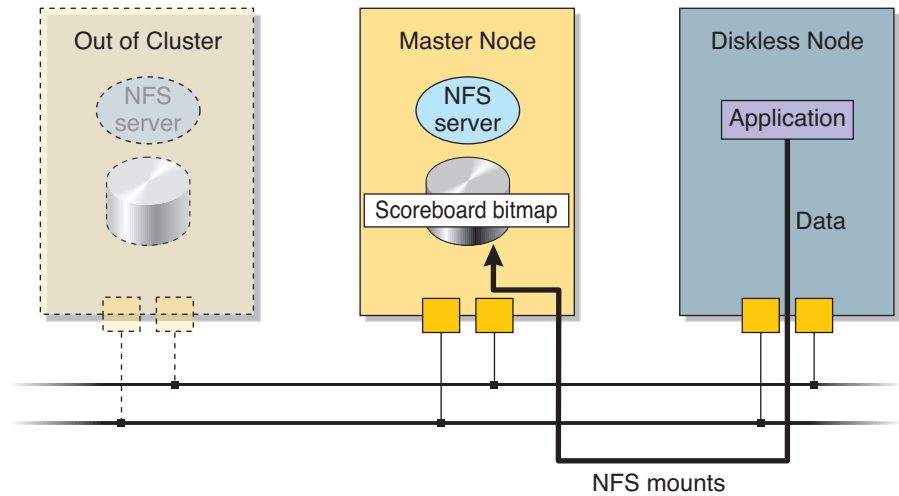
[FIGURE 6-2](#) illustrates Reliable File Service during failover or switchover.

---

**Note** – In the Solaris OS, the data area used to keep track of modifications to data blocks is referred to as a “scoreboard bitmap.” On the Linux OS, this data area is referred to as “Distributed Replicated Block Device (DRBD) metadata.”

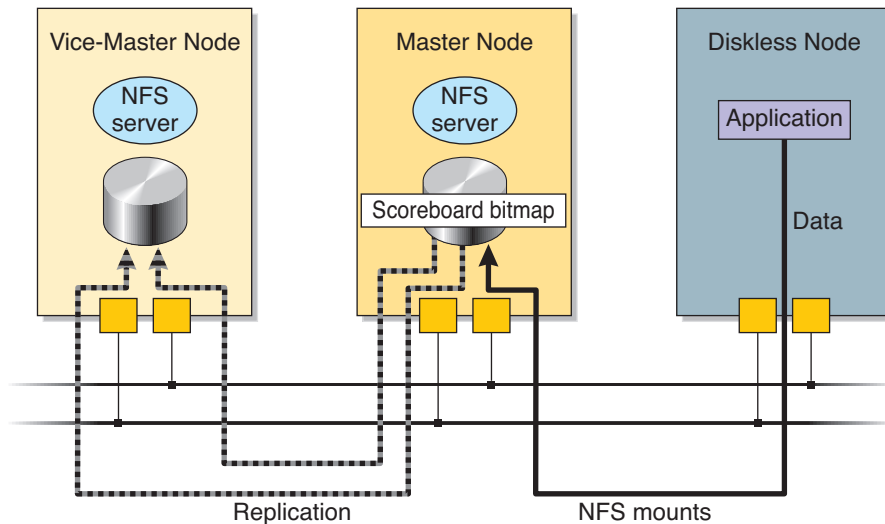
---

**FIGURE 6-2** Reliable File Service During Failover or Switchover



When the vice-master node is re-established, replication resumes. Any data written to the master node is replicated to the vice-master node. In addition, the data area used to keep track of modifications to data blocks is examined to determine which data blocks have been changed while the vice-master node was out of service. Any changed data blocks are also replicated to the vice-master node. In this way, the cluster becomes synchronized again. The following figure illustrates the restoration of the synchronized state.

**FIGURE 6-3** Restoration of the Synchronized State



You can verify whether a cluster is synchronized, as described in “To Verify That the Master Node and Vice-Master Node Are Synchronized” in the *Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide*.

You can collect replication statistics by using the Node Management Agent (NMA), as described in the *Netra High Availability Suite 3.0 1/08 Foundation Services NMA Programming Guide*.

---

**Note** – NMA is not available for use on Linux.

---

## Data Area Used to Track Modifications to Data Blocks

On the Solaris OS, the data area that is used to keep track of modifications that are made to data blocks is called the scorecard bitmap. On Linux, these modifications are tracked by DRBD metadata. The following sections describe how this works in each environment.

## Scorecard Bitmaps on the Solaris OS

When data is written to a replicated partition on the master node disk, the corresponding scoreboard bitmap is updated.

The scoreboard bitmap maps one bit to a block of data on a replicated partition. When a block of data is changed, the corresponding bit in the scoreboard bitmap is set to 1. When the data has been replicated to the vice-master node, the corresponding bit in the scoreboard bitmap is set to zero.

The scoreboard bitmap can reside on a partition on the master node disk or in memory. There are advantages and disadvantages to storing the scoreboard bitmap on the master node disk or in memory:

- Storing the scoreboard bitmap in memory is a problem if the master node and vice-master node fail simultaneously. In this case, the scoreboard bitmap is lost and a full resynchronization is required when the nodes are rebooted.
- Storing the scoreboard bitmap on a disk partition is slower during normal operation because writing to disk is slower than writing to memory. However, if the master node and vice-master node fail simultaneously, the scoreboard bitmap can be used to resynchronize the nodes, without the need for a full resynchronization.
- Storing the scoreboard bitmap in memory is encouraged when data is continuously and frequently updated, or when data is expected to be written to a replicated partition during a switchover. This ensures that writes to replicated partitions are performed as quickly as possible and that the time required to synchronize the partitions after a switchover is reduced.

Storing the scoreboard bitmap in memory means better performance because writing to memory is quicker than writing to disk.

Each replicated partition on a disk must have a corresponding partition for a scoreboard bitmap, even if the scoreboard bitmap is stored in memory.

For information about how to configure the scoreboard bitmap in memory or on disk, see “Changing the Location of the Scoreboard Bitmap” in the *Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide*.

## DRBD Metadata on Linux

On Linux, DRBD is used for disk replication. DRBD metadata keeps track of writes to replicated partitions on the master node disk and works like scoreboard bitmaps on the Solaris OS as described in preceding section, except that DRBD metadata cannot be kept in memory, it has to be stored on a disk partition.

# Synchronization Options

When a cluster is unsynchronized, the data on the master node disk is not fully backed up. You must not schedule major tasks when a cluster is unsynchronized. If this constraint does not suit the use you want to make of your cluster, you can choose to:

- Delay the start of synchronization
- Reduce the duration of synchronization

If you need to reduce the disk and network load on the cluster and so want to reduce the CPU used during synchronization, you can serialize slice synchronization.

## Delayed Synchronization

This feature enables you to delay the start of synchronization between the master and vice-master disks. By default, this feature is disabled. Delay synchronization if you do not want the synchronization task to conflict with other CPU-intensive activities. If you enable this feature, you can trigger synchronization at a later time of your choosing using the `nhenablesync` command.

Until synchronization is triggered and completed, the vice-master is not eligible to become master, and the cluster is open to a single point of failure. By delaying synchronization you are prolonging this vulnerability.

Activate this feature by setting the `RNFS.EnableSync` parameter in the `nhfs.conf` file. If you are installing the cluster with `nhinstall`, specify the synchronization method by setting `SYNC_FLAG` in the `cluster_definition.conf` file. For more information, see the `nhfs.conf4`, `cluster_definition.conf4`, and `nhenablesync1M` man pages.

## Reduced Duration of Disk Synchronization (Solaris OS Only)

Choose how the master and vice-master disks are synchronized if you want to reduce the time it takes to synchronize the master and vice-master disks. You can synchronize either an entire master disk or only those blocks that contain data. By choosing the latter option, you reduce the time it takes to synchronize the disks. You can choose to synchronize only the blocks that contain data if you have a UNIX® File System (UFS). If you have a file system other than UFS, the entire master disk is synchronized.



Set this feature to have the same value on both server nodes. If you change the value of the `RNFS.SyncType` property defining this feature, reboot the cluster as described in “Shutting Down and Restarting a Cluster” in the *Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide*.

Choose the synchronization method by setting the `RNFS.SyncType` parameter in the `nhfs.conf` file. If you are installing the cluster with `nhinstall`, choose the synchronization method by setting `SLICE_SYNC_TYPE` in the `cluster_definition.conf` file. For more information, see the `nhfs.conf4` and `cluster_definition.conf4` man pages.

## Serialized Slice Synchronization

This feature enables you to synchronize the master and vice-master disks one slice at a time rather than all at once. By default, this feature is disabled. Enabling this feature reduces the disk and network load. However, it limits the availability of the cluster during a certain limited time period because the vice-master cannot become master before all slices have been synchronized. During this time period, the cluster is vulnerable to a single point of failure.

Activate this feature by setting the value of the `RNFS.SerializeSync` parameter in the `nhfs.conf` file. If you are installing the cluster with `nhinstall`, specify the synchronization method by setting `SERIALIZE_SYNC` in the `cluster_definition.conf` file. For more information, see the `nhfs.conf4`, and `cluster_definition.conf4` man pages.

## Sanity Check of Replicated Slices

This feature enables you to continuously scan the state of replicated slices. By default, this feature is disabled and it can only be used if the master and vice-master disks are synchronized. If you do not monitor the state of the replicated slices, it is possible that the vice-master disk is corrupted and partly or completely inaccessible.

Activate this feature by setting the `RNFS.CheckReplicatedSlices` parameter in the `nhfs.conf` file. If you are installing the cluster with `nhinstall`, enable this feature by setting `CHECK_REPLICATED_SLICES` in the `cluster_definition.conf` file. For more information, see the `nhfs.conf4` and `cluster_definition.conf4` man pages.

# Disk Partitioning

This section describes how the master node disk and vice-master node disk are partitioned.

The master node, vice-master node, and dataless nodes access their local disks. The vice-master node and dataless nodes also access some disk partitions of the master node. Diskless nodes do not have, or are not configured to use, local disks. Diskless nodes rely entirely on the master node to boot and access services and data.

You can partition your disks as described in [“Solaris OS Standard Disk Partitioning” on page 42](#), or as described in [“Solaris OS Virtual Disk Partitioning” on page 43](#).

## Solaris OS Standard Disk Partitioning

To use standard disk partitioning, you must specify your disk partitions in the cluster configuration files. During installation, the `nhinstall` tool partitions the disks according to the specifications in the cluster configuration files. If you manually install the Netra HA Suite software, you must partition the system disk and create the required file systems manually.

The master node disk and vice-master node disk can be split identically into a maximum of eight partitions. For a cluster containing diskless nodes, you can arrange the partitions as follows:

- Three partitions for the system configuration
- Two partitions for data
- Two partitions for scoreboard bitmaps
- One free partition

Partitions that contain data are called *data partitions*. One data partition might typically contain the exported file system for diskless nodes. The other data partition might contain configuration and status files for the Foundation Services. Data partitions are replicated from the master node to the vice-master node.

To be replicated, a data partition must have a corresponding *scoreboard bitmap partition*. If a data partition does not have a corresponding scoreboard bitmap partition, it cannot be replicated. For information about the scoreboard bitmap, see [“Replication During Normal Operations” on page 35](#).

TABLE 6-1 shows an example disk partition for a cluster containing server nodes and client (diskless) nodes. This example indicates which partitions are replicated.

**TABLE 6-1** Example Disk Partition for a Cluster of Server Nodes and Client (Diskless) Nodes

Partition	Use	Replicated
s0	Solaris boot	Not replicated
s1	Swap	Not replicated
s2	Whole disk	Not applicable
s3	Data partition for diskless Solaris images	Replicated read/write for the diskless nodes
s4	Data partition for middleware data and binaries	Replicated read/write for applications
s5	Scoreboard bitmap partition	Used to replicate partition s3
s6	Scoreboard bitmap partition	Used to replicate partition s4
s7	Free	

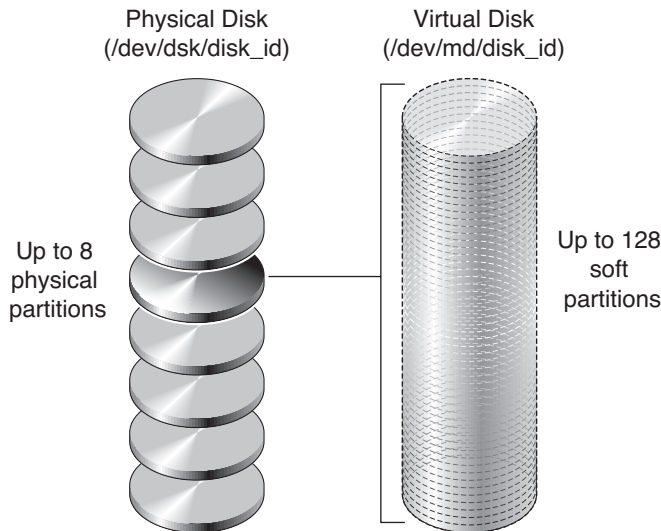
**Note** – Server nodes in a cluster that does not contain diskless nodes do not require partitions s3 and s5.

## Solaris OS Virtual Disk Partitioning

Virtual disk partitioning is integrated into the Solaris OS (since the Solaris 9 OS) in the Solaris Volume Manager software.

One of the partitions of a physical disk can be configured as a *virtual disk* using Solaris Volume Manager. A virtual disk can be partitioned into a maximum of 128 *soft partitions*. To an application, a virtual disk is functionally identical to a physical disk. The following figure shows one partition of a physical disk configured as a virtual disk with soft partitions.

**FIGURE 6-4** One Partition of a Physical Disk Configured as a Virtual Disk



In Solaris Volume Manager, a virtual disk is called a *volume*.

To use virtual disk partitioning, you must manually install and configure the Solaris Operating System and virtual disk partitioning on your cluster. You can then configure the `nhinstall` tool to install the Netra HA Suite software only, or you can install the Netra HA Suite software manually.

For more information about virtual disk partitioning, see the Solaris documentation.

## Linux Standard Disk Partitioning

To use standard disk partitioning on Linux, you must specify your disk partitions in the cluster configuration files. During installation, the `nhinstall` tool partitions the disks according to the specifications in the cluster configuration files. If you manually install the Foundation Services, you must partition the system disk and create the required file systems manually.

For a cluster containing diskless nodes, you can arrange the partitions on master and vice-master nodes as follows:

- Three partitions for the system configuration
- One partition for data
- One partition for DRBD metadata
- One free partition

Partitions that contain data are called data partitions. The data partition might contain configuration and status files for the Foundation Services. Data partitions are replicated from the master node to the vice-master node. If needed, more data partitions can be added.

To be replicated, a data partition must have corresponding DRBD metadata. Metadata for all replicated data partitions can be kept in a single metadata partition (in separate 128-MB slots).

TABLE 6-2 shows an example disk partition for a Netra HA Suite cluster. This example indicates which partitions are replicated.

**TABLE 6-2** Example Disk Partition for a Linux NHAS Cluster

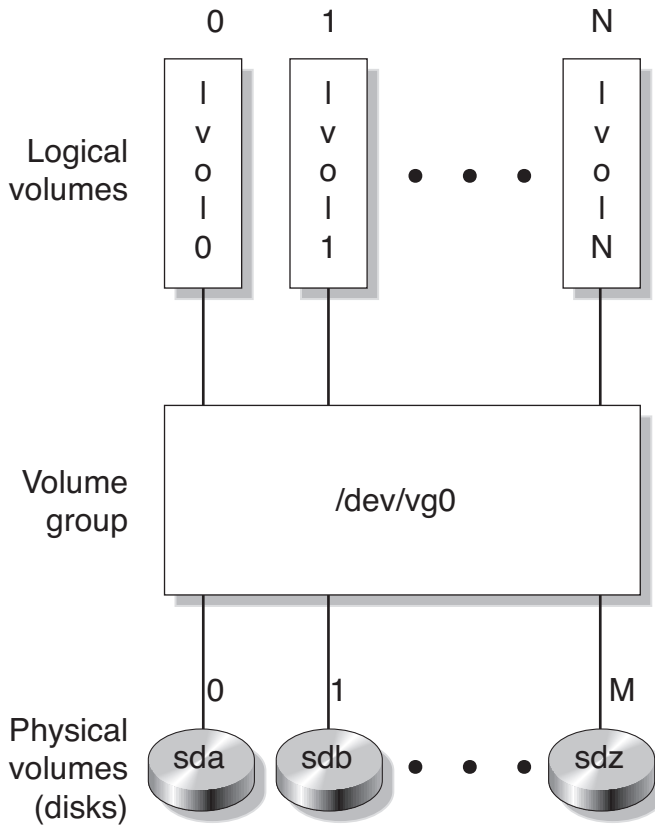
Partition	Use	Replicated
sda1	Linux root partition, boot	Not replicated
sda2	Swap	Not replicated
sda3	Extended partition	Not replicated
sda5	Data partition for middleware data and binaries	Replicated read/write for applications
sda7	DRBD metadata partition	Used to replicate sda5
sda8	free	

## Linux Virtual Disk Partitioning

Virtual disk partitioning on Linux can be done using the Logical Volume Manager (LVM). To do this, begin by declaring one or more physical disks as LVM physical volumes. Next, you need to create a volume group as a set of physical volumes. Finally, create the logical volumes from space that is available in a volume group.

To an application, a logical volume is functionally identical to a physical disk.

**FIGURE 6-5** Defining Logical Volumes on Linux Using LVM



To use virtual disk partitioning, you must manually install and configure the Linux operating system and virtual disk partitioning on your cluster. You can then configure the `nhinstall` tool to install only the Foundation Services, or you can install the Foundation Services manually. For more information about virtual disk partitioning, see the Linux and LVM documentation, for example: <http://www.tldp.org/HOWTO/LVM-HOWTO/index.html>

## Logical Mirroring

On the Solaris OS, logical mirroring is provided by Solaris Volume Manager (Solaris VM) for the Solaris 9 and Solaris 10 OS. In the Solaris environment, logical mirroring can be used with IP replication to strengthen system reliability and availability.

Logical mirroring can be used on server nodes with two or more disks. The disks are mirrored locally on the server nodes. They always contain identical information. If a disk on the master node is replaced or crashes, the second local disk takes over without a failover.

For more information about logical mirroring on the Solaris OS, see the Solaris documentation. For more information about logical mirroring on Linux, see the Linux Software RAID documentation, for example:  
<http://www.tldp.org/HOWTO/Software-RAID-HOWTO.html>

---

## Shared Disks

The shared disk capability is one of two mechanisms provided by the Reliable File Service for sharing data between a master node and a vice-master node. It requires at least one disk bay, which can be physically attached to the master node and to the vice-master node. Usually, the disk bay contains two disks that are mirrored using the Solaris VM. This section describes how the shared disk is partitioned.

---

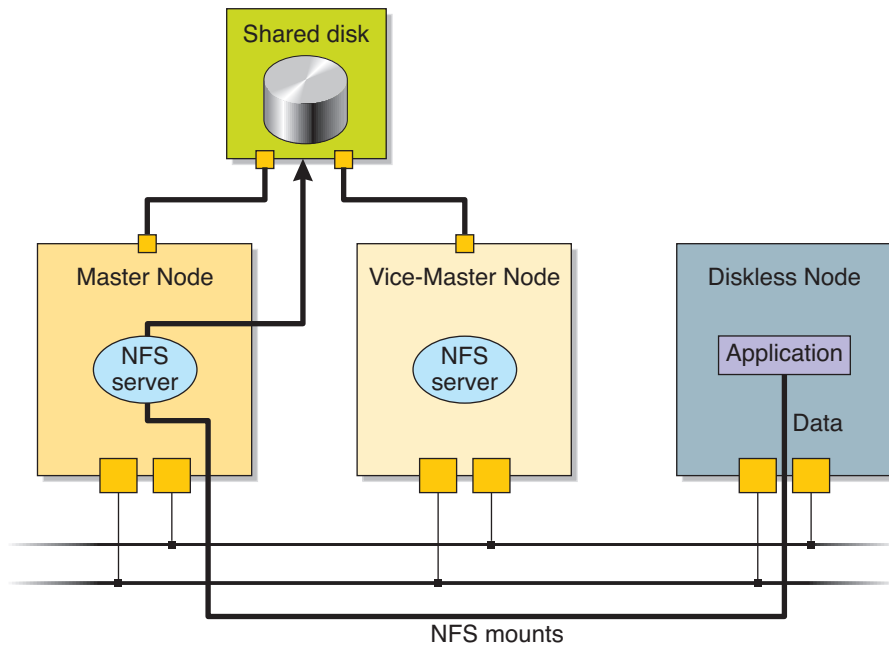
**Note** – The shared disk functionality is not supported for use with Carrier Grade Linux.

---

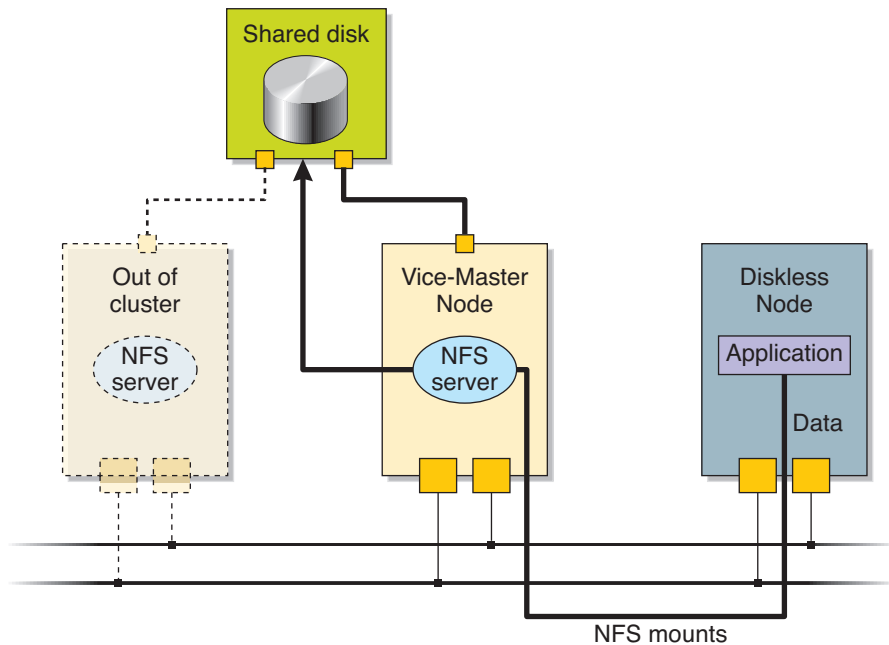
The master node, vice-master node, and dataless nodes access their local disks. The vice-master node and dataless nodes also access some disk partitions mounted on the master node. Client nodes do not have, or are not configured to use, local disks. Client nodes rely entirely on the master node to boot and access services and data.

The following figures show how data is replicated on a shared disk, how a shared disk fails or switches over, and how the shared disk is returned to a synchronized state.

**FIGURE 6-6** Data Replication Using Shared Disk

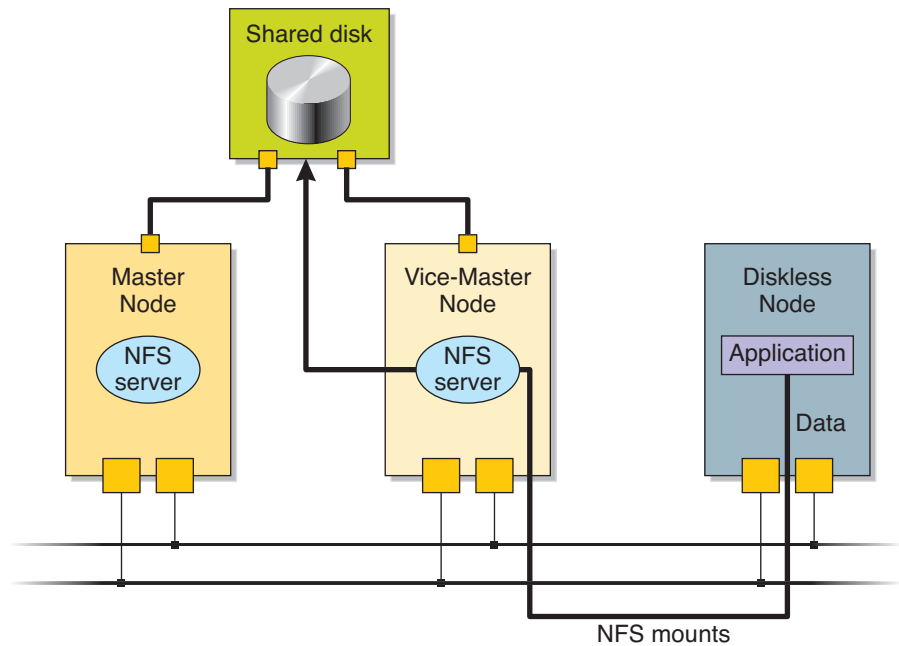


**FIGURE 6-7** Reliable File Service During Failover or Switchover for Shared Disk





**FIGURE 6-8** Restoration of the Synchronized State for Shared Disk



## Standard Shared Disk Partitioning

To use standard disk partitioning, you must specify your disk partitions in the cluster configuration files. During installation, the `nhinstall` tool partitions the disks according to the specifications in the cluster configuration files. If you manually install the Netra HA Suite software, you must partition the system disk and create the required file systems manually.

The shared disk can be split into a maximum of eight partitions. For a cluster containing client nodes, the partitions can be arranged as follows:

- Two partitions for data
- One partition used as a replica (Solaris VM entity)

Partitions that contain data are called *data partitions*. One data partition might typically contain the exported file system for client nodes. The other data partition might contain configuration and status files for the Foundation Services.

TABLE 6-3 shows an example disk partition for a cluster containing server nodes and client nodes. This example indicates which partitions are shared

**TABLE 6-3** Example Disk Partition for a Cluster of Server Nodes and Client Nodes

Partition	Use
s0	Data partition for diskless Solaris images
s1	Data partition for middleware data and binaries
s2	Whole disk
s7	Solaris VM replica

Server nodes in a cluster that does not contain client (diskless) nodes do not require partition s0.



## Reliable Boot Service

---

The Reliable Boot Service uses the Dynamic Host Configuration Protocol (DHCP) service and the other services provided by the Foundation Services to ensure the boot of diskless nodes regardless of software or hardware failure. For information about how diskless nodes are booted and how they are allocated IP addresses, see the following sections:

- [“Introduction to the Reliable Boot Service” on page 51](#)
- [“Boot Policies for Diskless Nodes” on page 52](#)
- [“Boot Policies for Diskless Nodes” on page 53](#)

---

## Introduction to the Reliable Boot Service

In a Netra HA Suite cluster, diskless nodes rely on network services to boot their operating system and run their software. The Reliable Boot Service provides a standard DHCP server on each of the server nodes. The Reliable Boot Service keeps the DHCP service operational even after failover or switchover.

The DHCP servers use a public DHCP module that is configured by the DHCP administration utilities `dhcpconfig`, `pntadm`, and `dhtadm`. For information about these utilities, see their man pages.

By default, the DHCP configuration files are stored on the master node. They are mounted on a mirrored file system and replicated on the vice-master node by the Reliable File Service. For information about the Reliable File Service, see [Chapter 6](#).

You can also put the DHCP configuration files locally on the master node and copy them to the vice-master node. For information on putting DHCP configuration files locally, see the `nhfs.conf4` man page.

The DHCP service provides several methods of allocating IP addresses to diskless nodes at boot.

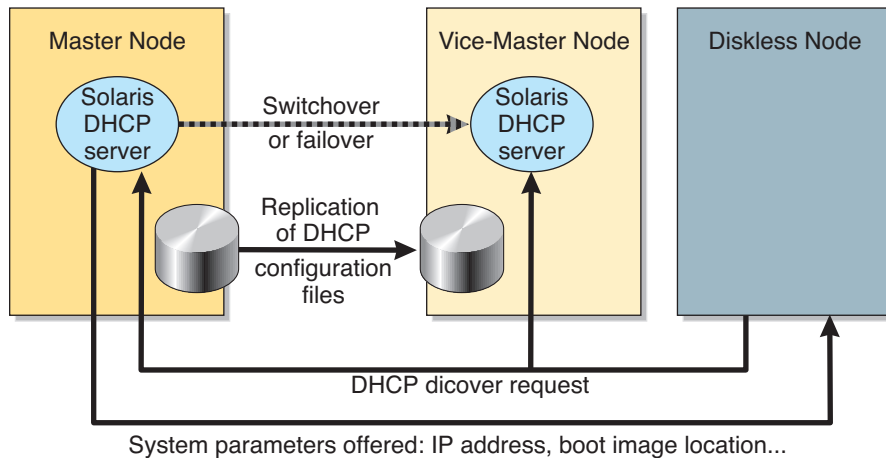
The DHCP daemon is started under the control of the Daemon Monitor. If the Reliable Boot Service fails, the Daemon Monitor takes the recovery action described in the `nhpmd1M` man page.

For further information about DHCP in the Solaris Operating System, see the *Solaris DHCP Service Developer's Guide*.

## Booting Diskless Nodes

FIGURE 7-1 shows the Reliable Boot Service on the master node and vice-master node, and the request for boot from a diskless node. The diskless node broadcasts a *DHCP discover request* to the DHCP servers on the server nodes. Only the master node responds to the DHCP discover request. After failover or switchover, the boot server on the new master node responds to the DHCP requests from diskless nodes.

**FIGURE 7-1** Request for Boot Broadcast From a Diskless Node



The Reliable Boot Service is notified by the Cluster Membership Manager when a new master node is elected, or when a node joins or leaves the cluster.

When a diskless node boots, the Reliable Boot Service assigns an IP address to it. If the node does not boot successfully within a specific time period, the Reliable Boot Service frees the allocated resources for the node. When a node leaves the cluster, the Reliable Boot Service retrieves the IP address that was being used by the node and clears the associated entry in the DHCP configuration files.

---

# Boot Policies for Diskless Nodes

The method of booting diskless nodes at cluster startup is called the *boot policy*. There are advantages and disadvantages to each of the boot policies. Your choice depends on the hardware configuration of your cluster. For information about how to configure a boot policy, see the *Netra High Availability Suite 3.0 1/08 Foundation Services Manual Installation Guide for the Solaris OS*. This section describes the boot policies used by the Foundation Services.

## DHCP Static

This boot policy maps the Ethernet address of a diskless node to a fixed IP address.

This system has the advantage of statically dedicating IP addresses to specific machines, making it possible to attribute groups of software to specific machines. However, this system does not support hardware hot-swap. Furthermore, when a node fails, the Ethernet address to IP address mapping remains assigned and cannot be reused.

## DHCP Client ID

This boot policy associates a `CLIENT_ID` string with a diskless node. When the diskless node is replaced, the `CLIENT_ID` string must be associated with the new node.



## External Addressing

---

For descriptions of the options available for connecting nonpeer nodes to the cluster network, see the following sections:

- [“Introduction to External Addressing” on page 55](#)
- [“External Addressing Scheme” on page 56](#)
- [“Connecting Nonpeer Nodes Directly to a Cluster Network” on page 57](#)
- [“Connecting Nonpeer Nodes to the Cluster Through Additional Physical Interfaces” on page 59](#)
- [“Connecting Nonpeer Nodes to the Cluster Network Through a Router” on page 61](#)

---

## Introduction to External Addressing

A non-peer node communicates through an external network with peer nodes of a cluster running the Netra High Availability (HA) Suite software to perform one or more of the following tasks:

- Access one or more of the Foundation Services, such as the Reliable File Service
- Access services provided by user applications
- Retrieve cluster information and statistics from the Node Management Agent
- Debug and maintain the cluster peer nodes
- Install software from a build server or an installation server
- Install software from a development host

The external network can be an Ethernet, ATM, or any other network type supported by the operating system.

An external network can be connected to a cluster peer nodes in the following ways:



- Directly to the cluster network (sharing a physical interface used by the cluster network)

If the cluster IP addresses are in a private network, logical interfaces must be created to connect the external network to the cluster network.

If the cluster IP addresses are in the same subnet as the external network, it is not necessary to create logical interfaces.

- Through additional physical interfaces on the peer nodes
- Through a router

---

## External Addressing Scheme

External addresses have no inherent relationship to internal cluster addresses. External addresses are flexible. They can be statically assigned to an interface, or they can fail over from a failed interface to a working one using IP multipathing on the Solaris OS or the Linux bonding driver on Linux. They can be IPv4 or IPv6.

## Floating External Addresses

A *floating external address* is a logical address assigned to an interface that is used to connect the master node to an external network. The External Address Manager (EAM) uses the Cluster Membership Manager (CMM) notifications to determine when a node takes on or loses the master role. When notified that a node has become the master node, the EAM configures the floating external addresses on one of the node's external interfaces. When notified that a node has lost the master role, the EAM unconfigures the floating external addresses. Because the floating external address is always configured on the master node, non-peer nodes on an external network can always access the master node, even after failover and switchover.

You can install the EAM when you first install the software on the cluster, or after you have completed the installation process and have a running cluster.

At the same time, floating external addresses can be managed by IPMP on the Solaris OS or by the Linux bonding driver on Linux. When a node has two or more NICs connected to the external network, IPMP (Solaris OS) or the Linux bonding driver (Linux) will failover the floating external addresses from one NIC to the other if the interface on which they are configured fails. Additionally, you can configure EAM to monitor the status of those NICs and trigger a switch-over when all NICs in a monitored group have failed.

For more IPMP information, see the Solaris *System Administration Guide: IP Services*.

---

# Connecting Nonpeer Nodes Directly to a Cluster Network

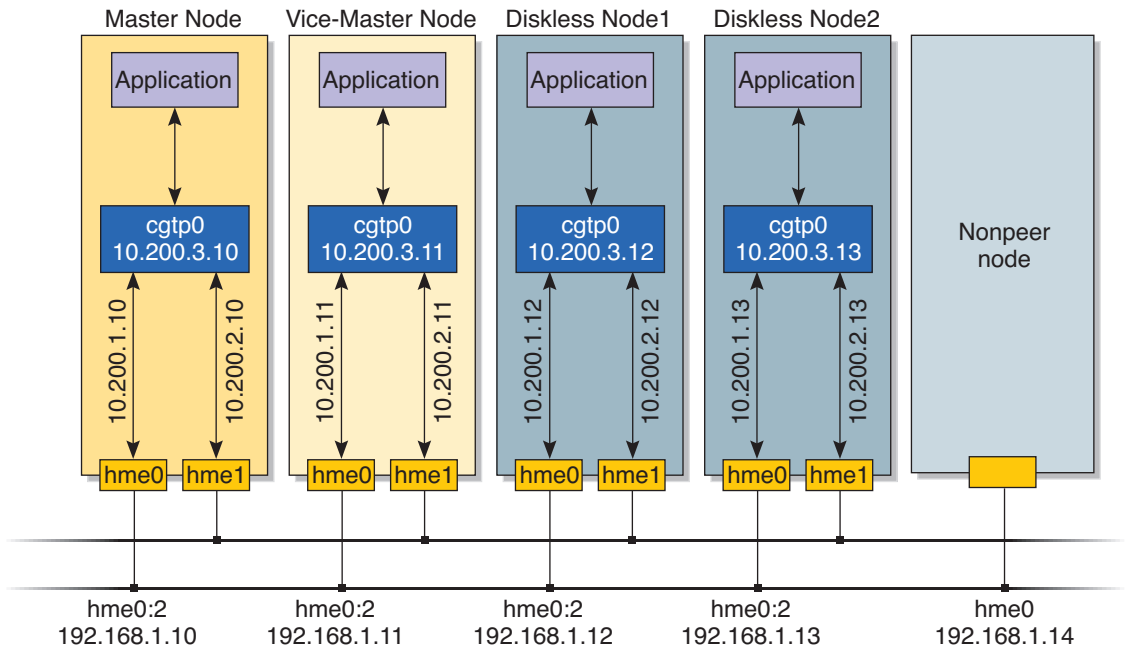
This section describes how a nonpeer node can be connected directly to a cluster network. Connecting a nonpeer node directly to the cluster network is disadvantageous for the following reasons:

- Internal traffic can leave the cluster network, compromising security.
- External traffic can enter the cluster network, reducing network performance.
- Traffic on the two cluster network paths can become asymmetric if the external network is connected to only one of the cluster networks. This could affect the performance of the redundant transport mechanism provided by CGTP.

[FIGURE 8-1](#) shows how you can connect directly to a cluster network with a public IP address.

In [FIGURE 8-1](#), the nonpeer node is connected to the `hme0` interface of each peer node. Each `hme0` interface has a logical interface called `hme0:2`, configured with an address in the public IP address space. The nonpeer node can access the cluster network through these logical interfaces.

**FIGURE 8-1** Example of a Nonpeer Node Connected Directly to a Cluster Network Using a Public IP Address Space



## Addressing a Shared Cluster Network and External Network

TABLE 8-1 shows the IP addresses of the master node in FIGURE 8-1. In addition to the addresses shown in FIGURE 8-1, the master node has a floating address for each interface. The External Address Manager configures the floating external address, hme0:3.

**TABLE 8-1** Example IP Addresses for a Master Node With a Logical Interface Configured for External Access

Address Type	Interface	IP Address
Master Node Addresses	hme0	10.200.1.10
	hme1	10.200.2.10
	cgtp0	10.200.3.10
External Address	hme0:2	192.168.1.10

**TABLE 8-1** Example IP Addresses for a Master Node With a Logical Interface Configured for External Access (*Continued*)

Address Type	Interface	IP Address
Floating Addresses	hme0:1	10.200.1.1
	hme1:1	10.200.2.1
	cgt0:1	10.200.3.1
Floating External Address	hme0:3	192.168.1.1

## Connecting Nonpeer Nodes to the Cluster Through Additional Physical Interfaces

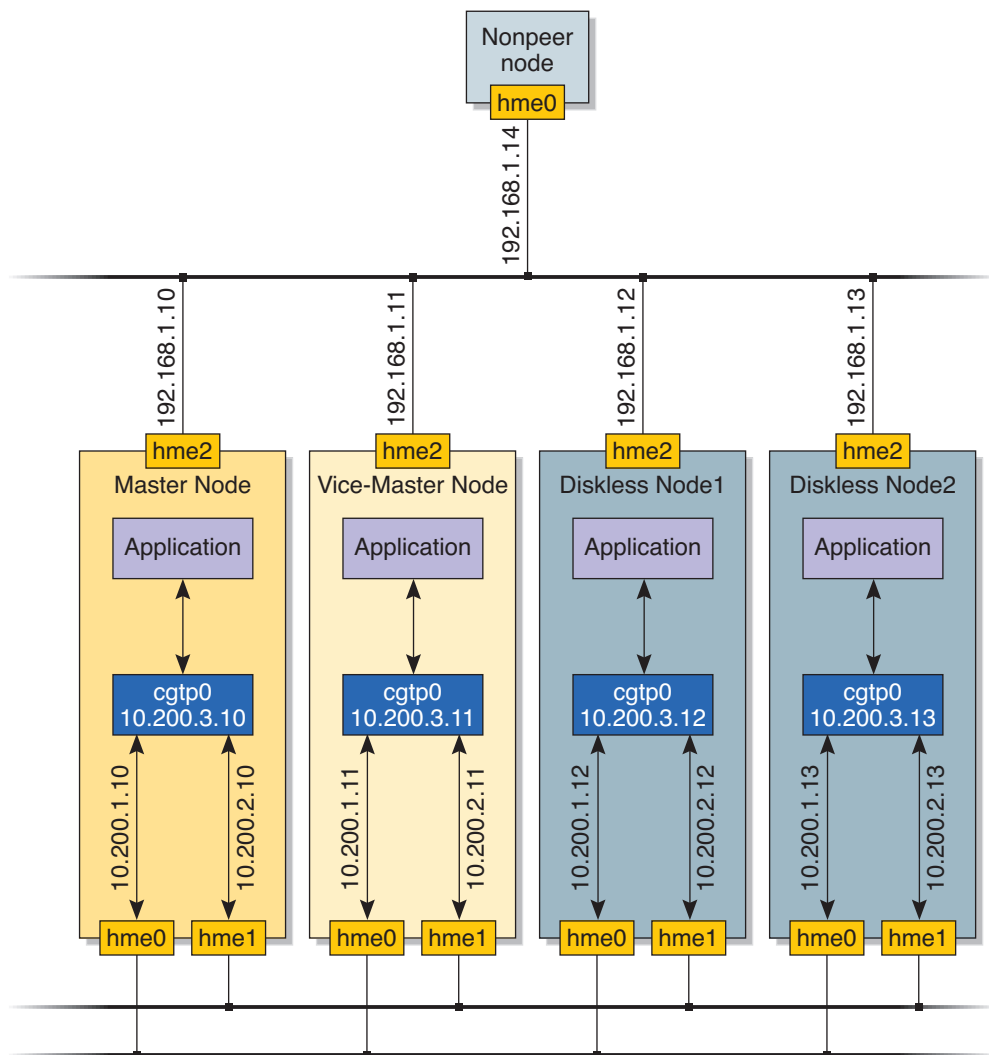
This section describes how a nonpeer node can be connected to the cluster network through additional physical interfaces on the peer nodes. This configuration is preferable to that in [“Connecting Nonpeer Nodes Directly to a Cluster Network” on page 57](#) for the following reasons:

- The cluster network is separate from the external network, giving better security and performance.
- The cluster network management is simplified.
- There are no restrictions on the external network addressing model.

All of the node hardware that is supported for use with the Netra HA Suite software can be configured with more than two physical interfaces. Also, more than one physical interface can be connected to the external network. In this case, external floating addresses are managed by IPMP on the Solaris OS and by the Linux bonding driver on Linux.

[FIGURE 8-2](#) shows an example of how you can connect a nonpeer node to a cluster through the physical interface hme2.

**FIGURE 8-2** Example of a Nonpeer Node Connected to the Cluster Network Through Additional Physical Interfaces on Peer Nodes



For simplicity, in [FIGURE 8-2](#) the nonpeer node is connected to each peer node through a single interface. This configuration would introduce a single point of failure. In highly available platforms, single points of failure must be avoided.

# Addressing Physical Interfaces That Are Connected to an External Network

TABLE 8-2 shows the IP addresses of the master node in FIGURE 8-2. In addition to the addresses in FIGURE 8-2, the master node has a floating address for each interface. Netra HA Suite software configures the floating external address, hme2:1.

TABLE 8-2 Example IP Addresses for a Master Node With Three Physical Interfaces

Address Group	Interface	IP Address
Master Node Addresses	hme0	10.200.1.10
	hme1	10.200.2.10
	cgtp0	10.200.3.10
	hme2	192.168.1.10
Floating Addresses	hme0:1	10.200.1.1
	hme1:1	10.200.2.1
	cgtp0:1	10.200.3.1
Floating External Address	hme2:1	192.168.1.0



## Connecting Nonpeer Nodes to the Cluster Network Through a Router

This section describes how to connect a nonpeer node to the cluster network through a router. The router node can contain the Network Address Translation (NAT) service to protect the cluster from unwanted external traffic.

The use of a router is advantageous compared to the scenario in “[Connecting Nonpeer Nodes Directly to a Cluster Network](#)” on page 57 for the following reasons:

- You can prevent internal traffic from leaving the cluster network.
- You can prevent external traffic from entering the cluster network.

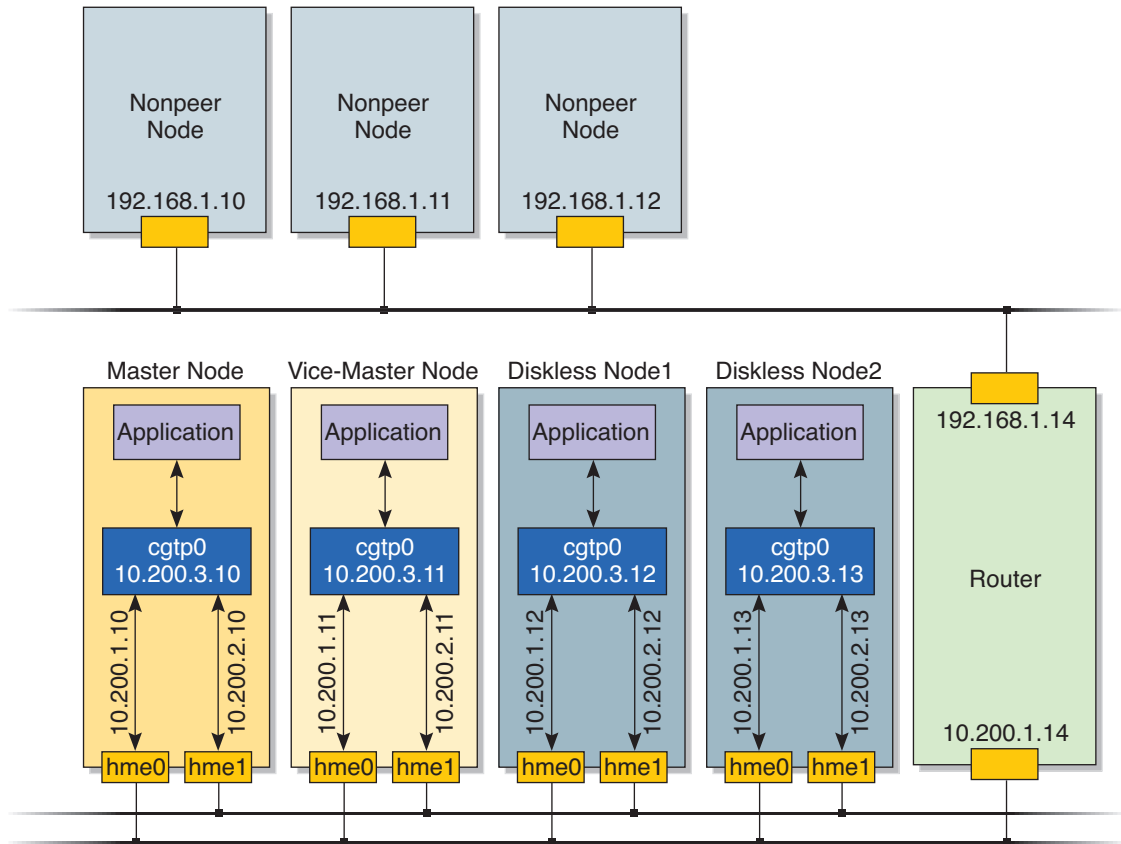
However, the use of a router is disadvantageous compared to the scenario in “[Connecting Nonpeer Nodes to the Cluster Through Additional Physical Interfaces](#)” on page 59 for the following reasons:

- It complicates the network configuration because routers must be configured.
- It could allow external traffic to enter the cluster network, reducing network performance.

- If the external network is connected to only one of the cluster networks, the traffic on the two cluster network paths can become asymmetric.

The following figure shows an example of how to connect several nonpeer nodes to a cluster network through a router.

**FIGURE 8-3** Example of Nonpeer Nodes Connected to the Cluster Network Through a Router



## Node State Manager

---

The Node State Manager (NSM) reacts to notifications from the Cluster Membership Manager (CMM) by executing user-provided scripts. This is a simple way to include an application that is not high availability-aware (non-HA-aware) in a cluster. For information about NSM notifications, see the following sections.

- [“Introduction to the Node State Manager” on page 63](#)
- [“Writing Scripts” on page 64](#)

---

## Introduction to the Node State Manager

On a running cluster, any new master or vice-master node is signaled to the applications that subscribed to receive notifications. To permit a non-HA-aware application to react to these notifications, the NSM will subscribe to the notifications and execute the user-provided scripts associated with each particular situation.

Scripts will be executed when the node on which the `nhnsmd` daemon is running becomes master or vice-master, but also when it stops being either of these. Scripts executed when a node enters a role are called `Ennxxxxxx`. Scripts executed when a node leaves a role are called `Lnnxxxxxx`. In both cases `nn` is a two-digit number and `xxxxxx` is the name of the script. Scripts are executed in alphabetical order, which means that the scripts `E01backup`, `E01server`, and `E02client` will be executed in that order.

The scripts must be located in the directories specified by the `NSM.Exec.MasterDir` and `NSM.Exec.ViceMasterDir` properties in the `/etc/opt/SUNWcgha/nhfs.conf` file on the Solaris OS, or in the `/etc/opt/sun/nhas/nhfs.conf` file on Linux.



On a role change, if the node was master or vice-master, it will first execute the Lnnxxxxxx scripts for the old role, and only then the Ennxxxxxx scripts for the new role. This procedure works in the same way as the system initialization rc.d Snnxxxxxx and Knnxxxxxx scripts.

---

## Writing Scripts

Scripts can perform whatever action is needed to launch or stop a service as long as their behavior doesn't interfere with the normal behavior of the Netra HA Suite software. For more information, see the `nhnsmdd1M` man page on the Solaris OS or the `nhnsmdd8` on Linux.

Two arguments are passed to the scripts you provide. The first is the action being executed, for which possible values are `enter` and `leave`. The second argument received by a user-provided script is the role of the node. The value for this argument can only be `master` or `vicemaster`. Using these arguments, the same script can be used for entering and leaving any of the roles.

Services must run on the master node. The script you provide that is invoked when the node enters the master role should launch the service (or activate it, if it is in standby state). The user-provided script that is invoked when a node leaves the master role should stop the service (or put it into standby state).

A script is not usually needed for nodes that are entering or leaving the vice-master role, but they are available for special cases where some special actions are needed on the vice-master node.

---

**Note** – You should not use the script that is invoked when a node enters the vice-master role to stop the services that are started when a node enters the master role. This script will not be invoked if the master node does not become vice-master (it is disqualified), which prevents the services from being stopped.

---

Scripts must have the “executable” flag set.

## Output From User-Defined Scripts

Any output (`stdout` and `stderr`) produced by these scripts will be directed to the log files. The log files will be placed in the directories that are specified in the `NSM.Log.MasterDir` and `NSM.Log.ViceMasterDir` properties and will be named with the script's name, followed by the `.log` extension. The logging directories must be writable by the superuser.

## Daemon Monitor

This chapter describes how the Daemon Monitor is used to survey other process daemons. It describes how the Daemon Monitor can be monitored and its recovery response changed and reset.

This chapter includes the following sections:

- [“The nhpmd Daemon” on page 65](#)
- [“Using the Node Management Agent With the Daemon Monitor” on page 66](#)

---

## The nhpmd Daemon

The nhpmd daemon provides the Daemon Monitor service. The nhpmd daemon runs at the multiuser level on all nodes in the cluster. The nhpmd daemon surveys other Foundation Services daemons, some operating system daemons (on the Solaris OS only), and some companion product daemons. If a daemon that provides a critical service fails, the nhpmd daemon detects the failure and triggers a recovery response. The recovery response is specific to the daemon that has failed. For a list of monitored daemons and their recovery responses, see the nhpmd1M man page.

The nhpmd daemon operates at a higher priority than the other Foundation Services daemons.

The Daemon Monitor is surveyed by a kernel module. When the kernel module detects an abnormal exit of the Daemon Monitor, it implements a panic that results in the crash and reboot of the node.

Foundation Services daemons and operating system daemons are launched by *scripts*. A *nametag* is assigned to the daemon or group of daemons that is launched by each startup script. In some cases, such as for `syslogd`, a *nametag* is assigned to only one daemon. In other cases, such as for `nfs_client`, a *nametag* is assigned to a group of daemons. If one of the daemons covered by a *nametag* fails, the recovery

response is performed on all of the daemons covered by that nametag. If the recovery response is to restart the failed daemon, all of the daemons grouped under that nametag are killed and then restarted.

Information about monitored daemons can be collected using the `nhpmdadm` command, as described in the `nhpmdadm1M` man page.

Information about the actions taken by the `nhpmd` daemon can be gathered from the system log files. For information on how to configure the system log files, see the *Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide*.

---

## Using the Node Management Agent With the Daemon Monitor

The Node Management Agent (NMA) can be used to collect the following information from a Daemon Monitor:

- Which daemons are monitored
- Which monitored processes have failed
- The number of times a failed daemon has been restarted
- The maximum number of times a failed daemon is allowed to be restarted

The NMA can be used to change the following parameters of the Daemon Monitor:

- The maximum number of times that the Daemon Monitor attempts to restart a daemon or group of daemons
- The reset of the current retry count for a monitored daemon

For information about the NMA, see [Chapter 11](#) and the *Netra High Availability Suite 3.0 1/08 Foundation Services NMA Programming Guide*.

## Node Management Agent

---

This chapter describes how the Node Management Agent (NMA) can monitor and manipulate a cluster. This chapter contains the following sections:

- [“Introduction to the Node Management Agent” on page 67](#)
- [“Monitoring Statistics With the NMA” on page 68](#)
- [“Manipulating the Cluster With the NMA” on page 71](#)
- [“Receiving Notifications With the NMA” on page 71](#)

---

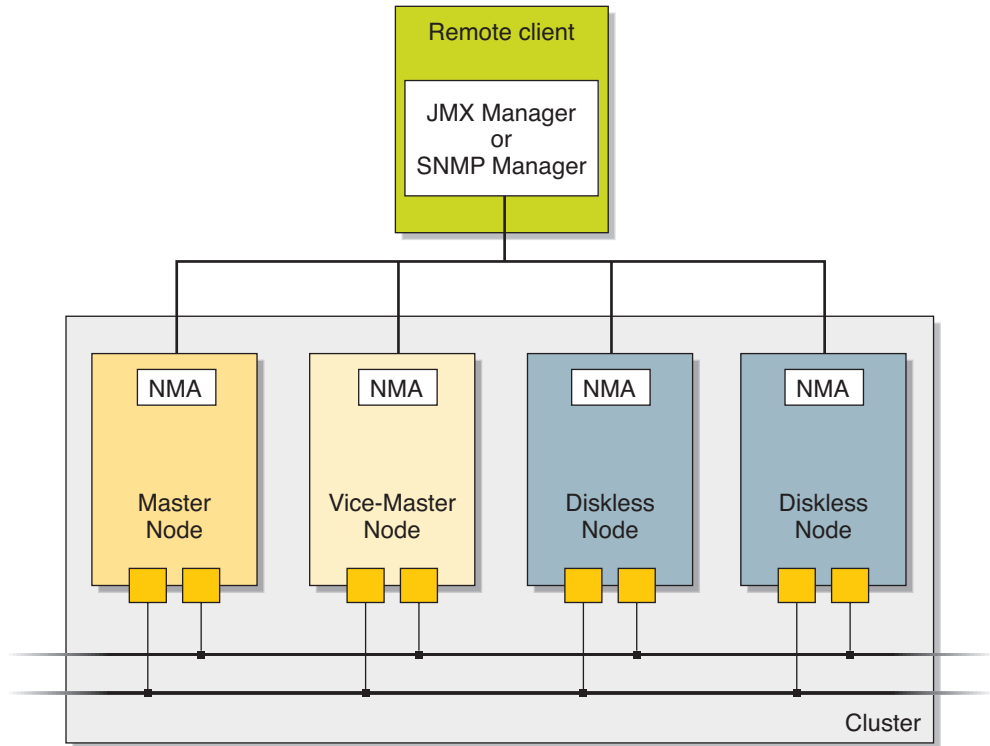
### Introduction to the Node Management Agent

The NMA is compliant with the Java Management Extensions (JMX) and based on the Java Dynamic Management Kit. The NMA provides access to cluster statistics through the Simple Network Management Protocol (SNMP) or through JMX clients using HTTP. The NMA supports the Internet Engineering Task Force standard RFC 2573.

The NMA retrieves statistics about the cluster membership, the reliable transport mechanism, the network file system, and the monitoring of process daemons. The NMA can be used to initiate a switchover, change the maximum number of times that it attempts to restart a daemon, reset the current retry count for a daemon, and listen for certain cluster notifications.

The following figure shows a remote client accessing nodes in a cluster.

**FIGURE 11-1** Remote Access to the Cluster



## Monitoring Statistics With the NMA

The NMA collects two types of statistics: *node-level statistics* and *cluster-level statistics*.

There is an NMA on each peer node that collects node-level statistics, that is, statistics for that node. Each NMA collects statistics about CGTP, CMM, and the Daemon Monitor. The NMA on each server node collects node-level statistics about Reliable NFS.

The NMA on the master node collects cluster-level statistics, that is, statistics about the cluster.

The following table describes the statistics that are collected by the NMA on each peer node and on the master node.

**TABLE 11-1** Statistics Collected by the NMA

Type of Statistics	Collected From:	Statistics
Node-level statistics	All peer nodes, including the master node	CGTP statistics, CMM statistics, and Daemon Monitor statistics Reliable NFS statistics on the server nodes
Cluster-level statistics	Master node only	CMM statistics, list of peer nodes, cluster Reliable NFS statistics, and high-level statistics for each peer node

The NMA can collect the following statistics:

■ CMM statistics

Some cluster membership statistics are collected on each node. Other cluster membership statistics can be collected by the NMA on the master node only. Statistics collected from the master node include the following:

- Information about mastership elections
- Information about switchovers
- Information about the direct link
- Details of the individual cluster nodes
- Activity of the nhcmmnd and nhprobed daemons

■ CGTP statistics

These statistics include a set of general CGTP statistics and a set of dedicated packet filtering statistics. Packet filtering statistics count the number of packets successfully received through each of the CGTP redundant links. In this way, packet filtering statistics measure the quality of the communication.

■ Reliable NFS statistics

These statistics give the file status and disk replication status. These statistics can be collected for server nodes only.

■ Daemon Monitor statistics

These statistics provide the following information:

- Which daemons are monitored
- Which monitored processes have failed
- The number of times a failed daemon has been restarted
- The maximum number of times a failed daemon is allowed to be restarted

- External Address Manager statistics

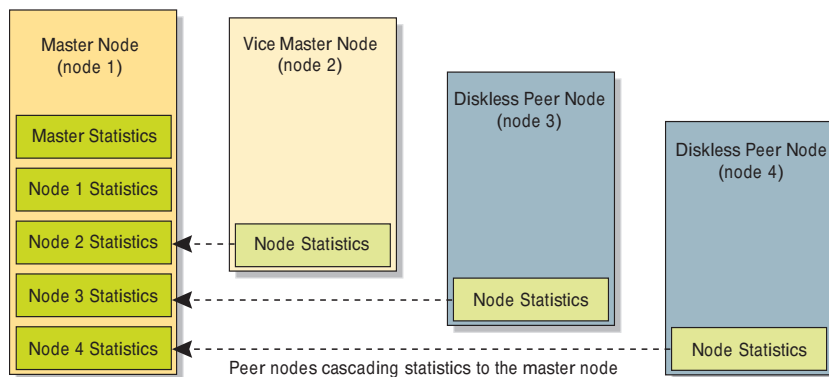
These statistics give the status of the floating addresses and IPMP groups configured in the `nhfs.conf` file. The PID of the `nheamd` daemon is also reported.

- Shared-Disk statistics

These statistics give the status of each replicated disk, its configuration, information about the diskset used by Netra HA Suite, the hosts that are connected to the shared devices, the owner of the diskset, and fencing information.

The NMA running on the master node *cascades* the statistics from the NMA on each of the peer nodes into its namespace. By cascading, the NMA on the master node can see the statistics on all of the peer nodes. In this way, the NMA on the master node has a view of the entire cluster. The following figure illustrates the cascade of statistics from the peer nodes to the master node.

**FIGURE 11-2** Cascading Data From Peer Nodes to the Master Node



A set of NMA APIs can be used to develop applications that monitor and react to the statistics produced by a cluster. For information about developing applications with the NMA APIs, see the *Netra High Availability Suite 3.0 1/08 Foundation Services NMA Programming Guide*.

The statistics that a cluster generates depend on the type, size, and arrangement of the cluster. Each cluster has an individual set of statistics. Knowing the statistics that your cluster generates when it runs correctly can help you to interpret the statistics when the cluster is failing. Use the NMA and its APIs to establish a set of statistics or a benchmark for your cluster when it is working correctly.

---

# Manipulating the Cluster With the NMA

The NMA can be configured to initiate a switchover or to change the following Daemon Monitor parameters:

- The maximum number of times that the Daemon Monitor attempts to restart a daemon or group of daemons
- The current retry count for a monitored daemon

For more information, see the *Netra High Availability Suite 3.0 1/08 Foundation Services NMA Programming Guide*.

---

# Receiving Notifications With the NMA

The NMA can be used to listen for notifications of the following cluster events:

- A node joins or leaves the cluster
- A master node or vice-master node is elected
- The maximum number of times that the Daemon Monitor attempts to restart a daemon or group of daemons is reached
- The maximum number of times that the Daemon Monitor attempts to restart a daemon or group of daemons is changed by the NMA
- The current number of times that the Daemon Monitor has attempted to restart a daemon or group of daemons is reset to zero by the NMA
- A nametag is created by the Daemon Monitor
- A nametag is removed by the Daemon Monitor

For more information, see the *Netra High Availability Suite 3.0 1/08 Foundation Services NMA Programming Guide*.





# Index

---

## A

- accessing data on the master node, 42, 47
- address
  - multicast, 30
  - triplets, 21
- address triplets, 21
- addressing
  - cluster, 19
  - external, 55, 56
- amnesia, 12
- APIs
  - Cluster Membership Manager (CMM), 28
  - Node Management Agent, 70
- architecture, Netra HA Suite, 2
- availability, 9

## B

- benchmark, cluster statistics, 70
- bitmap files, 39
- bitmap partition, 39
- boot policy
  - DHCP Client ID, 53
  - DHCP Static, 53
- booting diskless nodes, 51
- build server
  - installing software from, 55

## C

- Carrier Grade Transport Protocol *See* CGTP
- cascading, 70
- CGTP

- addresses, 21
  - collecting statistics on, 68, 69
  - configuring on peer nodes, 15
  - destination address, 16
  - Ethernet networks, 15
  - interfaces, 21
  - link failure, 17
  - monitoring statistics on, 69
  - not using, 16
  - redundancy, 17
  - routes, 15, 17
  - source address, 16
  - standalone, 15
  - summary of, 3
  - transfer of data packets, 16
  - using on nonpeer nodes, 15
- CLIENT\_ID parameter, 53
- cluster
  - addressing, 19
  - benchmarking statistics, 70
  - collecting statistics on, 68
  - definition of, 5
  - diagram of, 1
  - membership *See* Cluster Membership Manager
  - providing access *See* Node State Manager
  - remote access, 67
  - statistics
    - benchmarking, 70
    - retrieving from an external network, 55
    - types monitored, 69
- Cluster Membership Manager
  - API, 28
  - collecting statistics on, 68, 69

- configuring cluster membership, 28
- defining routing tables, 16
- detecting faults, 13
- detecting node failures, 27
- introduction, 27
- isolating faults, 14
- monitoring statistics on, 69
- reporting faults, 13
- sending notifications to Reliable NFS
  - daemon, 34
- summary of, 3
- using to configure CGTP, 15
- cluster network, 19
- CMM API, 28
- continuity of service, masterless cluster, 31
- critical services, detection of failure, 65

## D

- Daemon Monitor
  - collecting statistics on, 68, 69
  - detecting faults, 13
  - monitoring
    - daemons, 65
    - statistics, 69
  - restarting daemons, 71
  - starting the DHCP daemon, 52
  - summary of, 4
- daemons
  - information on monitored daemons, 66
  - monitoring *See* Daemon Monitor, 4
  - nametags, 65
  - nfs\_client, 65
  - nhcmmd, 27
  - nhcrfsd, 34
  - nhpmd, 65
  - nhprobed, 29
  - recovery from failure, 65
  - Reliable NFS, 34
  - starting the DHCP daemon, 52
  - syslogd, 65
- data cache, using when writing to shared file systems, 34
- data packet, 16
- data partition
  - mirroring
    - on Linux, 45
    - on Solaris, 42

- overview, 49
- dataless nodes, 8
- detecting faults, 13
- DHCP
  - administration utilities, 51
  - configuration files, 51
  - starting the DHCP daemon, 52
- dhcpconfig utility, 51
- dhtadm utility, 51
- diagnostics, 2
- direct link
  - description, 30
  - monitoring, 30, 69
- diskfull nodes, 7
- diskless nodes, 8
  - allocating IP addresses, 51
  - boot policies, 53
  - booting, 51, 52
- disks
  - failure, 47
  - mirroring logical, 46
  - partitioning, 42, 47
    - standard, 42, 44
    - virtual, 43, 44
  - replacement, 47
  - virtual
    - on Linux, 45
    - on Solaris, 43
- distributed services, 12
- double fault, 12
- DRBD disk replication, 39
- Dynamic Host Configuration Protocol *See* DHCP

## E

- error messages, 13
- Ethernet address
  - mapping to CLIENT\_ID parameter, 53
  - static mapping to IP address, 53
- Ethernet networks, requirements for CGTP, 15
- External Address Manager, summary, 3
- external addressing, 55, 56
- external network, 55

## F

- failover
  - control by Reliable NFS daemon, 34

- defined, 10
  - IP addresses, 34
  - replication, 36
- failure detection, 65
- faults
  - amnesia, 12
  - detecting, 13
  - double, 12
  - isolation, 14
  - recovery from, 12, 14
  - reporting, 13
  - single, 12
  - split brain, 12
  - types of, 12
- filtering packets, 17
- floating address triplet, 22
- H**
- hardware
  - hot-swap, 53
  - replacement, 2
  - upgrade, 2
- header of, 16
- heartbeats
  - defaults, 29
  - monitoring, 29
  - notifications, 29
- highly available services, 12
- host ID, 20
- host part of IP address, 20
- hot-swap, 53
- HTTP, using to provide cluster statistics, 67

- I**
- installation server, installing software from, 55
- interfaces
  - CGTP, 21
  - virtual, 21
- IP address
  - allocating for diskless nodes, 51, 53
  - class B, 20
  - class C, 20
  - generic format, 20
  - mapping from CLIENT\_ID parameter, 53
  - master node floating address triplet, 22
  - network part and host part, 20
  - node address triplets, 21

- node failover, 34
  - physical interfaces, 61
  - static mapping from Ethernet address, 53
- IP data packet, 16
- IP multipathing, 56
- IPv4 header, 16
- IPv6 header, 16
- isolating faults, 14

- J**
- Java Dynamic Management Kit, 67
- JMX clients, using to provide cluster statistics, 67

- L**
- Linux
  - DRBD for mirroring disks, 39
  - Volume Manager, 45
- log files
  - error message reporting, 13
  - information on nhpmd daemon actions, 66
- logical addresses, 22
- logical mirroring, 46

- M**
- master node
  - access to data by other nodes, 42, 47
  - collecting cluster statistics, 68
  - defined, 7
  - failover, 10
  - floating address triplet
    - diagram of, 23
    - diagram of after failover, 24
    - example, 24
    - overview, 22
  - partitioning, 42, 47
  - switchover, 10
- master-eligible nodes
  - defined, 7
  - logical mirroring, 47
- master-ineligible node, 8
- masterless cluster, continuity of service, 31
- membership of the cluster *See* Cluster Membership Manager
- metadevice, 44
- mirroring
  - data partitions

- on Linux, 45
  - on the Solaris OS, 42
  - overview, 49
- logical, 46
- monitoring
  - cluster statistics *See* Node Management Agent
  - daemons *See* Daemon Monitor, 4
  - statistics *See* Node Management Agent, 67
- multicast
  - address
    - overview, 30
    - recommendations for, 31
  - heartbeat, 29
  - transmission of heartbeat, 30

## N

nametags assigned to daemons, 65

netmask in IP address, 20

Netra HA Suite

- architecture diagram, 2
- definition of, 1
- summary of, 3

network

- cluster, 19
- external
  - accessing user application services, 55
  - connecting to the cluster network, 55
  - debugging the cluster, 55
  - developing applications, 55
  - installing software, 55
  - retrieving cluster statistics, 55

network ID, 20

network part of IP address, 20

network paths *See* routes, 15

nfs\_client daemon, 65

nhcmmd daemon, 27

nhcrfsd daemon, 34

nhpmd daemon, 65

nhpmdadm tool, 66

nhprobed daemon, 29

NMA *See* Node Management Agent, 4, 67

noac, 34

node address triplets, 21

node identity in IP address, 20

Node Management Agent

- APIs, 70

cascading, 70

monitoring cluster statistics, 67

monitoring the direct link, 30

reporting faults, 13

running on peer nodes, 68

running on the master node, 68

summary of, 4

Node State Manager

- introduction to, 63

- summary of, 4

nodes

- accessing data on master node, 22

- accessing data on the master node, 42, 47

- address triplets, 21

- addresstriplets, 21

- assigning roles, 3, 27

- backing up the master node, 7

- cluster membership, 27

- dataless

- in cluster, 8

- detecting failures, 3, 27

- diskfull, 7

- diskless

- accessing data on master node, 22

- allocating IP addresses, 3, 51, 53

- booting, 3, 51, 52

- in cluster, 8

- failure of master, 10

- heartbeats, 29

- master

- defined, 7

- floating address triplet, 22

- partitioning, 42, 47

- master-eligible, 7

- logical mirroring, 47

- master-ineligible, 8

- monitoring

- heartbeats, 29

- nodes *See* Node Management Agent

- statistics, 69

- nonpeer, 7

- peer, 7

- restarting independently, 14

- shutting down independently, 14

- switchover, 9

- types of, 7

- vice-master

- defined, 7

- partitioning, 42, 47

- nonpeer nodes, 7
  - connecting to the cluster network, 55
- notifications
  - changes in cluster membership or mastership, 27
  - changes in cluster state, 34
  - heartbeats, 29
  - listening for, 71
  - receiving, 28

## P

- packet, 16
- packets
  - filtering, 17
  - transfer using CGTP, 16
- partition
  - bitmap, 39
  - data
    - defined, 49
    - on Linux, 45
    - on the Solaris OS, 42
  - overview, 42
  - shared disk, 47
  - soft, 43
- partitioning
  - arranging partitions
    - on Linux, 44
    - on Solaris, 42
  - installation requirements
    - on Linux, 44
    - on Solaris, 42
  - virtual
    - on Linux, 45
    - on Solaris, 43
- peer nodes, 7
- pntadm utility, 51

## R

- recovery from faults, 12
- redundancy
  - CGTP, 17
  - defined, 9
  - model, 9
  - network paths *See* routes, 17
  - running critical systems, 27
  - support of, 2
- redundant routes, 15, 17

- reliability, 8
- Reliable Boot Service, 51
  - failure, 52
  - summary of, 3
- Reliable File Service, summary of, 3
- Reliable NFS
  - collecting statistics on, 68, 69
  - monitoring statistics on, 69
- reliable transport *See* CGTP, 15
- remote access to cluster, 67
- replication, 35
  - collecting statistics, 38
  - during failover or switchover, 36
  - scoreboard bitmap, 39
- reporting faults, 13
- RFC standards
  - RFC 2573, 67
- routes
  - nonredundant routes, 16
  - redundant routes, 17
- routing tables, 16

## S

- scoreboard bitmap, 39
- scripts
  - user defined, 63
  - writing, 64
- serviceability, 9
- services
  - critical, detection of failure, 65
  - distributed, 12
  - highly available, 12
- single fault, 12
- SNMP, using to provide cluster statistics, 67
- soft partition, 43
- Solaris Volume Manager, 43
- Solstice DiskSuite, 43
- split brain
  - defined, 12
  - preventing, 30
- standalone CGTP, 15
- statistics
  - retrieving from an external network, 55
  - types monitored, 69
- statistics *See* Node Management Agent

switching equipment, warning not to share, 15

switchover

- causing, 71

- control by Reliable NFS daemon, 34

- defined, 10

- replication, 36

- verifying feasibility of, 71

synchronization

- defined, 35

- verifying, 38

syslogd daemon, 65

## T

tools, nhpmdadm, 66

transfer of data packets, 16

transport mechanism *See* CGTP, 15

triplets, 21

## U

user-defined scripts

- overview, 63

- writing, 64

## V

vice-master node

- defined, 7

- partitioning, 42, 47

virtual disk partitioning on Solaris, 43

virtual interfaces, 21

volume, 44