



# Netra™ High Availability Suite 3.0 1/08 Foundation Services Linux Operating System Reference Manual

---

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

Part No. 819-5245-12  
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, 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, 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

---

<b>Preface</b>	<b>vii</b>
Intro	1
Intro	3
Intro	5
nhadm	7
nhcmmd	17
nhcmmqualif	19
nhcmmrole	21
nhcmmstat	23
nhcrfsadm	33
nhcrfsd	37
nheamd	39
nhenablesync	41
nhinstall	43
nhnsmd	47
nhpmd	49
nhpmdadm	53
nhprobed	63
nhsched	65
nhscsitool	67
cmm_cmc_filter	69
cmm_cmc_register	71
cmm_cmc_unregister	75

cmm\_config\_reload 79  
cmm\_connect 81  
cmm\_disconnect 83  
cmm\_master\_getinfo 85  
cmm\_mastership\_release 87  
cmm\_member\_getall 89  
cmm\_member\_getcount 91  
cmm\_member\_getinfo 93  
cmm\_member\_isdesynchronized 95  
cmm\_member\_isdisqualified 97  
cmm\_member\_iseligible 99  
cmm\_member\_isexcluded 101  
cmm\_member\_isfrozen 103  
cmm\_member\_ismaster 105  
cmm\_member\_isoutofcluster 107  
cmm\_member\_isqualified 109  
cmm\_member\_isvicemaster 111  
cmm\_member\_seizequalif 113  
cmm\_member\_setqualif 115  
cmm\_membership\_include 119  
cmm\_membership\_remove 121  
cmm\_node\_getid 123  
cmm\_notify\_dispatch 125  
cmm\_notify\_getfd 127  
cmm\_potential\_getinfo 129  
cmm\_strerror 131  
cmm\_vicemaster\_getinfo 133  
addon.conf 135  
cluster\_definition.conf 139  
cluster\_nodes\_table 159  
env\_installation.conf 163  
nhadmsync.conf 169  
nhfs.conf 173  
target.conf 199





# Preface

---

The *Netra High Availability Suite 3.0 1/08 Foundation Services Linux Operating System Reference Manual* describes the tools, API, configuration files, and drivers delivered in the Foundation Services. Both novice users and those familiar with Foundation Services can use online man pages to obtain information about the system and its features. A man page is intended to answer concisely the question “What does it do?” The man pages in general comprise the reference manual. They are not intended to be a tutorial.

---

## How This Book Is Organized

The following contains a brief description of each man page section and the information it references:

- Section 8 describes, in alphabetical order, the tools and commands that are used chiefly for installation and administration purposes.
- Section 3CMM describes, in alphabetical order, the functions in the Cluster Membership Manager (CMM) library. The CMM API provides applications with information about what nodes are in the cluster and the roles of nodes in the cluster.
- Section 5 describes, in alphabetical order, the installation and Foundation Services configuration files.
- Section 7 describes the Carrier Grade Transport Protocol (CGTP) driver.

---

**Note** – The Service Availability Forum (SA Forum) API provides processes with a means of retrieving information about the cluster nodes and cluster membership. For information about the functions in the Service Availability Forum (SA Forum) library, visit <http://www.saforum.org/home>.

---

Below is a generic format for man pages. The man pages of each manual section generally follow this format. All headings may not appear in each man page. See the `intro` pages for more information and detail about each section, and `man(1)` for more information about man pages in general.

NAME	This section gives the names of the command, function, or configuration file documented, followed by a brief description of what they do.
SYNOPSIS	<p>This section shows the syntax of the command, function, or configuration file. When a command or file does not exist in the standard path, its full path name is shown. Options and arguments are alphabetized, with single letter arguments first, and options with arguments next, unless a different argument order is required.</p> <p>The following special characters are used in this section:</p> <ul style="list-style-type: none"><li>[ ] Brackets. The option or argument enclosed in these brackets is optional. If the brackets are omitted, the argument must be specified.</li><li>. . . Ellipses. Several values can be provided for the previous argument, or the previous argument can be specified multiple times, for example, "filename . . .".</li><li>  Separator. Only one of the arguments separated by this character can be specified at a time.</li><li>{ } Braces. The options and/or arguments enclosed within braces are interdependent, such that everything enclosed must be treated as a unit.</li></ul>
DESCRIPTION	This section defines the functionality and behavior of the command, function, or configuration file. Thus it describes concisely what the command does. It does not discuss options or cite examples. Interactive commands, subcommands, requests, macros, and functions are described under <code>USAGE</code> .
OPTIONS	This section lists the command options with a concise summary of what each option does. The options are listed literally and in the order they appear in the <code>SYNOPSIS</code> section. Possible arguments to options are discussed under the option, and where appropriate, default values are supplied.
OPERANDS	This section lists the command operands and describes how they affect the actions of the command.
OUTPUT	This section describes the output – standard output, standard error, or output files – generated by the command.



RETURN VALUES	<p>If the man page documents functions that return values, this section lists these values and describes the conditions under which they are returned. If a function can return only constant values, such as 0 or -1, these values are listed in tagged paragraphs. Otherwise, a single paragraph describes the return values of each function. Functions declared <code>void</code> do not return values, so they are not discussed in RETURN VALUES.</p>
ERRORS	<p>On failure, most functions place an error code in the global variable <code>errno</code> indicating why they failed. This section lists alphabetically all error codes a function can generate and describes the conditions that cause each error. When more than one condition can cause the same error, each condition is described in a separate paragraph under the error code.</p>
USAGE	<p>This section lists special rules, features, and commands that require in-depth explanations. The subsections listed here are used to explain built-in functionality:</p> <ul style="list-style-type: none"> <li>Commands</li> <li>Modifiers</li> <li>Variables</li> <li>Expressions</li> <li>Input Grammar</li> </ul>
EXAMPLES	<p>This section provides examples of usage or of how to use a command or function. Wherever possible a complete example including command-line entry and machine response is shown. Whenever an example is given, the prompt is shown as <code>example%</code>, or if the user must be superuser, <code>example#</code>. Examples are followed by explanations, variable substitution rules, or returned values. Most examples illustrate concepts from the SYNOPSIS, DESCRIPTION, OPTIONS, and USAGE sections.</p>
ENVIRONMENT VARIABLES	<p>This section lists any environment variables that the command or function affects, followed by a brief description of the effect.</p>
EXIT STATUS	<p>This section lists the values the command returns to the calling program or shell and the conditions that cause these values to be returned. Usually, zero is returned for successful completion, and values other than zero for various error conditions.</p>
FILES	<p>This section lists all file names referred to by the man page, files of interest, and files created or required by commands. Each is followed by a descriptive summary or explanation.</p>
ATTRIBUTES	<p>This section lists characteristics of commands, utilities, and device drivers by defining the attribute type and its corresponding value. See <code>attributes(5)</code> for more information.</p>

SEE ALSO	This section lists references to other man pages, in-house documentation, and outside publications.
DIAGNOSTICS	This section lists diagnostic messages with a brief explanation of the condition causing the error.
WARNINGS	This section lists warnings about special conditions which could seriously affect your working conditions. This is not a list of diagnostics.
NOTES	This section lists additional information that does not belong anywhere else on the page. It takes the form of an aside to the user, covering points of special interest. Critical information is never covered here.
BUGS	This section describes known bugs and, wherever possible, suggests workarounds.

## 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
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

Application	Title	Part Number
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 Operating System 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 Operating System 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

## 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 Linux Operating System Reference Manual*, part number 819-5245-12.



NAME	Intro - introduction to daemons, system maintenance commands, and installation tool commands	
DESCRIPTION	This section lists the daemons, system maintenance commands, and installation tool commands.	
Daemons	This section lists the daemons.	
	nhcmmd (8)	Cluster Membership Manager (CMM) daemon
	nhcrfsd (8)	Reliable NFS supervisory daemon
	nheamd (8)	External address manager (EAM) daemon
	nhpmd (8)	Daemon Monitor daemon
	nhprobed(8)	Probe link and node accessibility daemon
	nhnsmd (8)	Node State Manager daemon
	nhwdtd (8)	Watchdog Timer daemon
Maintenance Commands	This section lists the system maintenance commands.	
	nhadm (8)	Foundation Services administration tool
	nhcmmqualif (8)	Command to qualify the current node as master
	nhcmmrole (8)	Command to obtain the role of the node
	nhcmmstat(8)	Command to display information about peer nodes, trigger a switchover, or force the qualification of a master-eligible node
	nhcrfsadm(8)	Command for Reliable NFS administration
	nhenablesync (8)	Command to trigger synchronization
	nhpmdadm (8)	Daemon Monitor administration tool
	nhsched (8)	Command to display the scheduling parameters of the Foundation Services processes
	nhscsitool (8)	Command to enable the removal of remaining SCSI3 PGR keys or SCSI2 reservations from a disk
nhinstall Command	This section contains the nhinstall command.	
	nhinstall (8)	Foundation Services installation and configuration tool

(1M)



<b>NAME</b>	Intro - introduction to the Cluster Membership Manager API functions	
<b>DESCRIPTION</b>	This section describes the Cluster Membership Manager (CMM) Application Programming Interface (API) functions.	
<b>LIST OF FUNCTIONS</b>		
	<code>cmm_cmc_filter(3cmm)</code>	Define notification filtering
	<code>cmm_cmc_register(3cmm)</code>	Register to receive notifications
	<code>cmm_cmc_unregister(3cmm)</code>	Unregister to stop receiving notifications
	<code>cmm_config_reload(3cmm)</code>	Reload the cluster node table
	<code>cmm_connect(3cmm)</code>	Prepare or test a connection to the Cluster Membership Manager
	<code>cmm_disconnect(3cmm)</code>	Close a connection between current calling process and the <code>nhcmmnd</code> daemon
	<code>cmm_master_getinfo(3cmm)</code>	Retrieve master node information
	<code>cmm_vicemaster_getinfo(3cmm)</code>	Retrieve the vice-master node information
	<code>cmm_mastership_release(3cmm)</code>	Trigger a switchover
	<code>cmm_member_getall(3cmm)</code>	Retrieve cluster membership information
	<code>cmm_member_getcount(3cmm)</code>	Retrieve number of nodes in the cluster
	<code>cmm_member_getinfo(3cmm)</code>	Retrieve information about a peer node
	<code>cmm_member_isdesynchronized(3cmm)</code>	Interpret the synchronization status of a master-eligible node
	<code>cmm_member_isdisqualified(3cmm)</code>	Remove peer node
	<code>cmm_member_iseligible(3cmm)</code>	Determine if a node is master-eligible
	<code>cmm_member_isexcluded(3cmm)</code>	Determine if a node is excluded from the cluster
	<code>cmm_member_isfrozen(3cmm)</code>	Determine if a node is frozen
	<code>cmm_member_ismaster(3cmm)</code>	Determine if a node is the master
	<code>cmm_member_isvicemaster(3cmm)</code>	Determine if a node is the vice master

<code>cmm_member_isoutofcluster(3cmm)</code>	Determine if a node is not participating in the cluster
<code>cmm_member_isqualified(3cmm)</code>	Determine if a node is eligible to be master
<code>cmm_membership_include(3cmm)</code>	Rejoin the cluster after having left it
<code>cmm_membership_remove(3cmm)</code>	Remove a peer node from the cluster
<code>cmm_member_seizequalif(3cmm)</code>	Force requalification of master-eligible node
<b><code>cmm_member_setqualif(3cmm)</code></b>	Give a new level of qualification to a node
<code>cmm_node_getid(3cmm)</code>	Retrieve ID of a node
<b><code>cmm_notify_dispatch(3cmm)</code></b>	Dispatch cluster membership change messages
<code>cmm_notify_getfd(3cmm)</code>	Receive cluster membership change messages by getting file descriptor associated with registration
<b><code>cmm_potential_getinfo(3cmm)</code></b>	Retrieve information about a peer node even if it has the <code>OUT_OF_CLUSTER</code> role.
<code>cmm_strerror(3cmm)</code>	Get error message string



NAME	Intro - introduction to the Foundation Services configuration files													
DESCRIPTION	<p>The following types of configuration files are included in this section:</p> <ul style="list-style-type: none"><li>■ Configuration files for the <code>nhinstall</code> tool</li><li>■ Configuration files for the cluster</li><li>■ Configuration files for the Node Management Agent (NMA)</li></ul>													
LIST OF FILES														
<code>nhinstall</code> Configuration Files	<p>The following files enable you to configure the <code>nhinstall</code> tool to install the Foundation Services on the cluster:</p> <table><tr><td><code>addon.conf(5)</code></td><td><code>nhinstall</code> configuration file to install additional patches and packages</td></tr><tr><td><code>cluster_definition.conf(5)</code></td><td><code>nhinstall</code> configuration file to define the cluster</td></tr><tr><td><code>dataless_nodeprof.conf(5)</code></td><td>file that permits the customization of the operating system installation on dataless nodes</td></tr><tr><td><code>diskless_nodeprof.conf(5)</code></td><td>file that permits the customization of the operating system installation for the diskless node environment</td></tr><tr><td><code>env_installation.conf(5)</code></td><td><code>nhinstall</code> configuration file defining the installation environment</td></tr><tr><td><code>nodeprof.conf(5)</code></td><td>file that permits the customization of the operating system installation</td></tr></table>		<code>addon.conf(5)</code>	<code>nhinstall</code> configuration file to install additional patches and packages	<code>cluster_definition.conf(5)</code>	<code>nhinstall</code> configuration file to define the cluster	<code>dataless_nodeprof.conf(5)</code>	file that permits the customization of the operating system installation on dataless nodes	<code>diskless_nodeprof.conf(5)</code>	file that permits the customization of the operating system installation for the diskless node environment	<code>env_installation.conf(5)</code>	<code>nhinstall</code> configuration file defining the installation environment	<code>nodeprof.conf(5)</code>	file that permits the customization of the operating system installation
<code>addon.conf(5)</code>	<code>nhinstall</code> configuration file to install additional patches and packages													
<code>cluster_definition.conf(5)</code>	<code>nhinstall</code> configuration file to define the cluster													
<code>dataless_nodeprof.conf(5)</code>	file that permits the customization of the operating system installation on dataless nodes													
<code>diskless_nodeprof.conf(5)</code>	file that permits the customization of the operating system installation for the diskless node environment													
<code>env_installation.conf(5)</code>	<code>nhinstall</code> configuration file defining the installation environment													
<code>nodeprof.conf(5)</code>	file that permits the customization of the operating system installation													
Cluster Configuration Files	<p>The following are the cluster configuration files. These files are installed and configured by the <code>nhinstall</code> tool. Only if you are installing the Foundation Services manually, you must modify these files:</p> <table><tr><td><code>cluster_nodes_table(5)</code></td><td>Node management file</td></tr><tr><td><code>nhadmsync.conf(5)</code></td><td>List of non shared files and the differences between them</td></tr><tr><td><code>nhfs.conf(5)</code></td><td>Foundation Services configuration file</td></tr><tr><td><code>target.conf(5)</code></td><td>Basic node configuration file</td></tr></table>		<code>cluster_nodes_table(5)</code>	Node management file	<code>nhadmsync.conf(5)</code>	List of non shared files and the differences between them	<code>nhfs.conf(5)</code>	Foundation Services configuration file	<code>target.conf(5)</code>	Basic node configuration file				
<code>cluster_nodes_table(5)</code>	Node management file													
<code>nhadmsync.conf(5)</code>	List of non shared files and the differences between them													
<code>nhfs.conf(5)</code>	Foundation Services configuration file													
<code>target.conf(5)</code>	Basic node configuration file													



NAME	<p>nhadm - cluster administration tool</p> <p><b>/opt/sun/sbin/nhadm</b> [ -d <i>data-file</i> ] <b>copy</b> [ <i>file ...</i> ] \</p>										
SYNOPSIS	<p><b>/opt/sun/sbin/nhadm</b> [ -f <i>file</i> ] [ -s ] [ -v ] <b>check</b> [ <i>stage</i> ]</p> <p><b>/opt/sun/sbin/nhadm</b> [ -f <i>file</i> ] [ -s ] [ -v ] <b>display</b></p> <p><b>/opt/sun/sbin/nhadm</b> -h</p> <p><b>/opt/sun/sbin/nhadm</b> [ -s ] [ -v ] <b>confshare</b> [ <i>shared_package_directory</i> ]</p> <p><b>/opt/sun/sbin/nhadm</b> [ -s ] [ -v ] [ -y <i>file</i> ] <b>synccheck</b></p> <p><b>/opt/sun/sbin/nhadm</b> [ -s ] [ -v ] [ -y <i>file</i> ] <b>syncgen</b></p> <p><b>/opt/sun/sbin/nhadm</b> [ -z ] [ <b>html</b>   <b>text</b> ]</p> <p><b>/opt/sun/sbin/nhadm</b> [ -f <i>file</i> ] [ -s ] [ -v ] <b>configure</b></p>										
DESCRIPTION	<p>The <b>nhadm</b> tool provides a suite of tools to configure and check the installation and configuration of the Foundation Services software and its prerequisite products. You must log in as superuser to use this command.</p>										
OPTIONS	<p>The options that you can use with the <b>nhadm</b> tool are as follows:</p> <table> <tr> <td>-d   --data</td><td>Specifies the name of the file that lists the files to be copied</td></tr> <tr> <td>-f   --fsconf</td><td>Specifies the name of the Foundation Services configuration file. If this option is not used, the default file is <code>/etc/opt/sun/nhas/nhfs.conf</code>. For information on this file, see the <code>nhfs.conf(5)</code> man page.</td></tr> <tr> <td>-h</td><td>Displays a help screen.</td></tr> <tr> <td>-s   --silent</td><td>Runs in silent mode. When using the <b>nhadm</b> check command in this mode, the tests being run are not displayed. Only the errors encountered by these tests are displayed.</td></tr> <tr> <td>-v   --verbose</td><td></td></tr> </table>	-d   --data	Specifies the name of the file that lists the files to be copied	-f   --fsconf	Specifies the name of the Foundation Services configuration file. If this option is not used, the default file is <code>/etc/opt/sun/nhas/nhfs.conf</code> . For information on this file, see the <code>nhfs.conf(5)</code> man page.	-h	Displays a help screen.	-s   --silent	Runs in silent mode. When using the <b>nhadm</b> check command in this mode, the tests being run are not displayed. Only the errors encountered by these tests are displayed.	-v   --verbose	
-d   --data	Specifies the name of the file that lists the files to be copied										
-f   --fsconf	Specifies the name of the Foundation Services configuration file. If this option is not used, the default file is <code>/etc/opt/sun/nhas/nhfs.conf</code> . For information on this file, see the <code>nhfs.conf(5)</code> man page.										
-h	Displays a help screen.										
-s   --silent	Runs in silent mode. When using the <b>nhadm</b> check command in this mode, the tests being run are not displayed. Only the errors encountered by these tests are displayed.										
-v   --verbose											

`-y | --syncfile`

Runs in verbose mode. In this mode, traces are displayed when performing operations. The level of detail provided in the traces increases every time this option is added.

`-z | --err [ html | text ]`

Specifies the name of the file that lists the nonreplicated files that you want to compare. The default is `/SUNWcgha/remote/etc/nhadmsync.conf`. For more information on this file, see the `nhadmsync.conf(5)` man page.

Prints messages corresponding to all the error scenarios tested by `nhadm` and provides explanations for these errors in html or text form (html is the default). You can use this command to help understand error messages generated by `nhadm` when you test a node or cluster.

**COMMANDS**

The commands and stages available with the `nhadm` tool are the following:

`check`

Verifies that the prerequisite software and hardware are correctly installed and configured, that your cluster has started correctly, and that all peer nodes are accessible. The tests run by `nhadm check` are broken down into the following stages. You can run an individual stage that suits the task you are performing by typing `nhadm check stage` or you can run all three stages by typing `nhadm check`.

- `installation`

Checks that the correct version of the operating system is installed. This command also checks that the Foundation Services packages and necessary patches are present and installed correctly.

- `configuration`

Checks that the configuration files required before starting up the Foundation Services are present and of the correct format. Also checks that the configuration has been performed successfully.

- `starting`

Tests node accessibility/disk replication on a running cluster.

**Note** – If a stage is not specified, all the stages are run.

`configure`

Performs Netra HA Suite post-installation commands, like enabling DRBD at startup, or adding Netra HA Suite services in `inetd.conf`.

This command is called by `nhinstall` after the installation of Netra HA Suite packages on Linux nodes.

`copy`

Copies files from the master node to the vice-master node. Files listed can be passed as an argument or listed in the *data-file* file.

<code>display</code>	Prints information to the console of the node on which it is run. The displayed information includes local and shared packages installed, node and cluster IDs, network interface information, local and mounted file systems, mount points, and shared partition information. This command can help you diagnose problems by listing the current node configuration.
<code>synccheck</code>	Uses the operating system's <code>diff</code> command to compare the files listed in <code>/SUNWcgha/remote/etc/nhadmsync.conf</code> and prints a warning on the console when the files are not identical on the master-eligible nodes. You must include files in <code>/SUNWcgha/remote/etc/nhadmsync.conf</code> that are on the master and the vice-master but are not replicated on a shared file system. The compared files can include the operating system's files and Foundation Services files. For more information on the <code>diff</code> command, see <code>diff(1)</code> .
<code>syncgen</code>	Accepts the differences between the two nodes found by <code>synccheck</code> for each file listed in <code>/SUNWcgha/remote/etc/nhadmsync.conf</code> .

## EXTENDED DESCRIPTION

The `nhadm` tool enables you to verify the status of your cluster. You should use the `nhadm` tool as part of regular maintenance or after you change a cluster configuration in any way.

The `nhadm display` command prints the current status of a node to the console. Further cluster status information can be obtained by running the `nhcmmstat` command, as described in `nhcmmstat(8)`.

The `nhadm` tool displays an `ok` message for every check that passes. If any of the tests performed by `nhadm check` fail, an error message is displayed on the console. The error message describes the error and identifies the command that has failed, or the likely problem area. For an explanation of the possible error messages, use the `nhadm -err` option.

In a correctly functioning cluster, replicated files are the same on the master node and the vice-master node. Some system files cannot be placed on shared file systems but must contain the same information on both master-eligible nodes. You can list the differences between nonreplicated files and manage the differences between these files using the `nhadm synccheck` and `nhadm syncgen` commands.

The list of nonreplicated files compared by `nhadm synccheck` is shared between the master node and the vice-master node. A template for this file is installed at `/opt/sun/nhas/templates/nhadm/nhadmsync.conf.template`. The default location of this shared file is `/SUNWcgha/remote/etc/nhadmsync.conf`. For further information on the `nhadmsync.conf` file, see the `nhadmsync.conf(5)` man page.

**EXAMPLES**

This section gives examples of how to use the `nhadm` tool and its commands.

**Using `nhadm check`**

This section contains examples of using `nhadm check installation`, `nhadm check conf` and `nhadm check starting`.

Note that the checking stages might vary, depending on the node configuration. For example, the disk check mechanism changes if you are using the logical volume manager. Therefore, the following examples are guidelines only.

After installing the hardware and software, log in to the machine you want to examine and run the `nhadm check installation` command:

**EXAMPLE 1 To Verify Software Installation**

```
# nhadm check installation
```

The `nhadm` tool verifies that:

- All required software packages and patches are installed
- The operating system and patches have the correct version
- The same MAC address is not used twice

When the Foundation Services software has been configured, log in to the peer node you want to examine, and run the `nhadm check configuration` command. This command checks that the configuration files that are required before starting the Foundation Services have been correctly configured.

**EXAMPLE 2 To Verify Software Configuration**

```
# nhadm check configuration
```

This command tests the following:

- The cluster definition files are present and in the correct format.
- The network configuration is correctly defined in `/etc/hostname` and `/etc/hosts`.
- The boot-device is configured to be `disk` for master-eligible and dataless nodes and `net` for diskless nodes.

For a Netra 20 peer node, boot-device is `*disk*` where `disk` in this case cannot match `disk` for the OpenBoot PROM.

- The auto-boot option is set to `true`.

**Note** – If the `auto-boot-retry` variable exists on your system, it must be set to `true`; if it does not exist on your system, disregard this reference to it.

- The `local-mac-address` for this node is set to `true`.

### Comparing Nonreplicated Files

- The root file system is defined in `/etc/fstab` and the partitions listed in `/etc/fstab` exist.

Use `nhadm check starting` to list any cluster network, or disk replication problems, by logging in to a peer node in a running cluster and typing:

**EXAMPLE 3** To Verify Cluster Network and Disk Replication

```
# nhadm check starting
```

The `nhadm` tool verifies that:

- Each node interface exists and is functioning correctly
- Each peer node is accessible from the current node
- The shared file systems are accessible and replicated if required

This section provides examples for using the `nhadm synccheck` and `nhadm syncgen` commands.

To use these commands, both master-eligible nodes must have remote access to each other. To enable this, make sure that the CGTP address of the other master-eligible node is set in the `.rhosts` file on each master-eligible node. For example:

On node `cgtp10` the `.rhosts` file must contain the CGTP address of the other master-eligible node, `cgtp11`:

```
cgtp11 root
```

On node `cgtp11`, the `.rhosts` file should contain the CGTP address of the other master-eligible node, `cgtp10`:

```
cgtp10 root
```

This enables `nhadm` to perform `rsh` between the master eligible nodes.

To use the `nhadm synccheck` command, you must specify the nonreplicated files that you want to compare. By default, you specify the list of files in `/SUNWcgha/remote/etc/nhadmsync.conf`. The `nhadm synccheck` command compares the copies of these files on the master and the vice-master nodes, printing any differences to the console. You can accept the differences using `nhadm syncgen`. Accepted differences are not printed to the console when you run `nhadm synccheck` again.

You can change the name and location of the file that stores the list of nonreplicated files to be compared. You can also have several files containing different lists of nonreplicated files.



**EXAMPLE 4** To specify the Nonreplicated Files to be Compared

Create the `/SUNWcgha/remote/etc/` directory. Copy the template file `/opt/SUNWcgha/config.standard/adm/nhadmsync.conf.template` to `/SUNWcgha/remote/etc/nhadmsync.conf`.

Add the names of the files to be compared to the `nhadmsync.conf` file. The files that you add should have the following criteria:

- The files exist on both master-eligible nodes
- The files are not replicated on a shared file system

Use the following syntax for all entries in the file:

```
FILE=filename
```

For further information on the `nhadmsync.conf` file see the `nhadmsync.conf(5)` man page.

**EXAMPLE 5** To Determine the Differences Between Nonreplicated Files

```
# nhadm synccheck
```

This command compares the differences between the files listed in `nhadmsync.conf`, and displays a list of the files that differ. For each file name displayed to the console, the difference that exists between the two copies of the same file is also given.

If you decide that the differences between the nonreplicated files displayed by `nhadm synccheck` are not detrimental to your cluster, update `nhadmsync.conf` with the differences by typing:

**EXAMPLE 6** To accept the Differences Between Nonreplicated Files

```
# nhadm syncgen
```

If the `nodeid` of the master node and vice-master node were not entered in the `nhadmsync.conf` file by a previously run `nhadm syncgen` command, a `NODEID` parameter is generated at the top of the file using the `nodeid` of the master and the vice-master nodes in the following format.

```
NODEID=node1 node2
```

Where *node1* and *node2* are the *nodeids* of the master and vice-master nodes, respectively, when `nhadm syncgen` is first run.

The *nodeid* can be preceded by a blank line or a comment. This *nodeid* defines the order of the comparison performed by `synccheck`. The order will remain the same even if a switchover or failover occurs.

Any differences displayed on the console are written to the `nhadmsync.conf` file. The differences for a specific file are printed under the entry for that file in the following format:

```
=BEGIN
...
=END
```

For example, if the `nhadmsync.conf` file contained the following files:

```
FILE=/etc/ethers
FILE=/etc/hosts
FILE=/etc/netmasks
```

After a `syncgen` the `nhadmsync.conf` file might contain the following information:

```
NODEID=10 20
FILE=/etc/ethers
FILE=/etc/hosts
=BEGIN
5c5,6
  10.250.1.10    MEN-C250-N10    loghost
---
> 10.250.1.20    MEN-C250-N20    loghost
> 10.250.1.10    MEN-C250-N10
8d8
  10.250.1.20    MEN-C250-N20
=END
FILE=/etc/netmasks
```

The differences printed to the `nhadmsync.conf` file are the differences that would be found by running `diff -b` on the files listed in `nhadmsync.conf`. For more information on the `diff` command, see the `diff(1)` man page.

For further information on the `nhadmsync.conf` file, see the `nhadmsync.conf(5)` man page.

Log in to one of the master-eligible nodes and type:

**EXAMPLE 7** To Check for New Differences Between Nonreplicated Files

```
# nhadm synccheck
```

For nonreplicated files listed in `nhadmsync.conf`, any differences that are not already stored in `nhadmsync.conf` are displayed to the console.

The `nhadmsync.conf` file is shared by the master and vice-master nodes. To change the name or location of the `nhadmsync.conf` file, use the `-y` option as follows:

**EXAMPLE 8** To Change the Location of the File Listing the Nonreplicated Files Compared by `nhadm synccheck`

```
# nhadm -y pathname syncgen
```

**Copying Local Files**

This section provides an example for using the `nhadm copy` command to copy local files from the master node to the vice-master node.

```
# nhadm [-d data-file ] copy [ file ]
```

where:

*data-file*                      a list of files to be copied; one file per line

*file*                            additional files to be copied, for example files that you want to copy once only. You can include the files to be copied repeatedly in the *data-file* file.

The `.rhosts` files must be correctly configured on both nodes to enable remote access via `rcp`.

The following example shows how to copy the local DHCP configuration file, `dhcp.dat` from the master node to the vice-master node. The contents of a DHCP configuration file such as `dhcp.dat` could be as follows:

**EXAMPLE 9** To Copy Local DHCP Configuration Files

```
/var/dhcp/SUNWrbs1_10_1_1_0
/var/dhcp/SUNWrbs1_10_1_2_0
/var/dhcp/SUNWrbs1_dhcptab
```

To copy `dhcp.dat` from the master node to the vice-master node, log on to the master node and use the `nhadm` command as follows:

```
# nhadm -d dhcp.dat copy
```

(1m)

**SEE ALSO**

The vice-master node now has a copy of the files listed in the `dhcp.dat` file.

`diff(1)`, `nhcmmstat(8)`, `nhadmsync.conf(5)`

NAME	nhcmmnd - manage cluster membership
SYNOPSIS	<code>/opt/sun/sbin/nhcmmnd [-h] [-u URL]</code>
DESCRIPTION	<p>The Cluster Membership Manager is implemented by the <code>nhcmmnd</code> daemon. There is a <code>nhcmmnd</code> daemon on each peer node.</p> <p>The <code>nhcmmnd</code> daemon on the master node has the current view of the cluster configuration and communicates its view to the <code>nhcmmnd</code> daemons on the other peer nodes. The <code>nhcmmnd</code> daemon on the master node determines which nodes are members of the cluster, assigns the roles and attributes to the nodes, detects the failure of nodes and configures routes for reliable transport.</p> <p>The <code>nhcmmnd</code> daemon on the vice-master node monitors the health of the master node. If the master node fails, the vice-master node is able to take over as the master node.</p> <p>The <code>nhcmmnd</code> daemon on each of the peer nodes do not communicate with each other. Each <code>nhcmmnd</code> daemon exports an API to the notify clients of changes to the cluster, and to notify services and applications when the cluster membership or master changes. Notification messages describe the membership change and the <i>nodeid</i> of the affected node, making it possible for clients to maintain an accurate view of the peer nodes of the cluster.</p> <p>For information about the CMM API, see the <code>Intro(3cmm)</code> man page.</p>
OPTIONS	<p><code>-h</code> Displays help information.</p> <p><code>-u URL</code> You must specify the URL of the <code>nhfs.conf</code> file.</p>
EXIT STATUS	<p>The following exit values are returned:</p> <p>0 Successful completion.</p> <p>255 An error occurred.</p>
FILES	<p><code>cluster_nodes_table</code> The file that lists the configured nodes of the cluster and describes the nodes' attributes.</p> <p><code>nhfs.conf</code> The file that contains configuration and addressing information for the different Foundation Services.</p> <p><code>target.conf</code> The file that contains the <i>domainid</i>, attributes, and election roles for each node in the cluster</p>

(1m)

**SEE ALSO** | `Intro(3cmm)`, `cluster_nodes_table(4)`, `nhfs.conf(4)`, and `target.conf(4)`.

<b>NAME</b>	<code>nhcmmqualif</code> - qualify the current node as master
<b>SYNOPSIS</b>	<code>/opt/sun/sbin/nhcmmqualif [-v] [-t <i>timeout</i>]</code>
<b>DESCRIPTION</b>	<p>The <code>nhcmmqualif</code> command calls the <code>cmm_member_seizequalif(3cmm)</code> function to qualify the current node as master-eligible and start a new master election. This call is only successful if there is no active master node in the cluster and if the current node is master-eligible. If this call is not successful, a 255 exit code is returned. If this call is successful, this command forces the qualification of the current node so that this node becomes the master node. Use this command when no node is qualified to become master. Note that an unsynchronized node will only be elected as master node if its former role was master.</p> <p>The <code>nhcmmqualif</code> command can only be called from a master-eligible node. This call has one of two outcomes for this node; either the node becomes master, or it reverts to the qualification level it had prior to the <code>nhcmmqualif</code> call.</p> <p>If you attempt to call <code>nhcmmqualif</code> from a node that is not master-eligible, the command exits with a 255 exit code. If a master is already running when <code>nhcmmqualif</code> is called, the command exits with a 255 exit code. If <code>nhcmmqualif</code> provokes a change in the status of a peer node, a notification is sent by the CMM API.</p>
<b>OPTIONS</b>	<p>The following options are supported by <code>nhcmmqualif</code>:</p> <ul style="list-style-type: none"> <li><code>-t <i>timeout</i></code>      Wait for a specified period of time for a MASTER ELECTED notification. If a master is elected within this period of time, <code>nhcmmqualif</code> is successful and returns 0. If no master is elected within this period of time, <code>nhcmmqualif</code> fails and returns 255.</li> <li><code>-v</code>                Verbose mode.</li> </ul>
<b>EXIT STATUS</b>	<p>The following exit values are returned:</p> <ul style="list-style-type: none"> <li>0                    Successful completion.</li> <li>255                  The current node is not master-eligible, there is already a master in the cluster, or no master was elected within the specified timeout (if <code>-t</code> was used).</li> </ul>
<b>EXAMPLES</b>	<p>This section contains examples of how to use the <code>nhcmmqualif</code> command.</p> <p><b>EXAMPLE 1</b>    To Force Qualification of a Master-Eligible Node</p> <p>When there is no current master node in the cluster and no node is qualified to be the master node:</p>

- Log in to a master-eligible node.
- Run:

```
# nhcmmqualif
# echo $?
0
```

The master-eligible node on which you ran the `nhcmmqualif` command is temporarily qualified as the master node.

If you run `nhcmmqualif` on a master-eligible node in a cluster with a valid master node, the node is not forced to become master and the 255 exit status is produced.

**EXAMPLE 2** To Requalify a Node Synchronously

Use the `timeout` option to requalify a node synchronously.

- Log in to one of the master-eligible nodes.
- Run:

```
# nhcmmqualif -t timeout
```

This command is synchronous. The option `-t` blocks the node until a `MASTER ELECTED` notification is received or the timeout is reached.

**SEE ALSO**

`cmm_member_seizequalif(3cmm)`.



NAME	nhcmmrole - get the role of the current node										
SYNOPSIS	<b>opt/sun/sbin/nhcmmrole</b> [ -hv ] [ -t <b>timeout</b> ]										
DESCRIPTION	<p>The <code>nhcmmrole</code> command gets the role of the current node.</p> <p>This executable can be used in scripts. Its exit code represents the current role of the node.</p>										
OPTIONS	<p>The <code>nhcmmrole</code> command can be used with the following options:</p> <table> <tr> <td>-h</td><td>Provides information about the use of the command.</td></tr> <tr> <td>-t <i>timeout</i></td><td>Sets the timeout in seconds for this call. The default timeout is 5 seconds.</td></tr> <tr> <td>-v</td><td> <p>Verbose. This option displays the role of the node, as follows:</p> <pre>nhcmmrole: current role <i>role</i></pre> <p>The value for <i>role</i> is one of the following: MASTER, VICEMASTER, OUT_OF_CLUSTER, or IN_CLUSTER.</p> </td></tr> </table>	-h	Provides information about the use of the command.	-t <i>timeout</i>	Sets the timeout in seconds for this call. The default timeout is 5 seconds.	-v	<p>Verbose. This option displays the role of the node, as follows:</p> <pre>nhcmmrole: current role <i>role</i></pre> <p>The value for <i>role</i> is one of the following: MASTER, VICEMASTER, OUT_OF_CLUSTER, or IN_CLUSTER.</p>				
-h	Provides information about the use of the command.										
-t <i>timeout</i>	Sets the timeout in seconds for this call. The default timeout is 5 seconds.										
-v	<p>Verbose. This option displays the role of the node, as follows:</p> <pre>nhcmmrole: current role <i>role</i></pre> <p>The value for <i>role</i> is one of the following: MASTER, VICEMASTER, OUT_OF_CLUSTER, or IN_CLUSTER.</p>										
EXIT STATUS	<p>The following exit values are returned by <code>nhcmmrole</code>:</p> <table> <tr> <td>255</td><td>A failure has occurred or the node is not configured in the cluster node table</td></tr> <tr> <td>0</td><td>Out of cluster node</td></tr> <tr> <td>1</td><td>Master node</td></tr> <tr> <td>2</td><td>Vice-master node</td></tr> <tr> <td>3</td><td>In cluster node that is neither master nor vice-master.</td></tr> </table>	255	A failure has occurred or the node is not configured in the cluster node table	0	Out of cluster node	1	Master node	2	Vice-master node	3	In cluster node that is neither master nor vice-master.
255	A failure has occurred or the node is not configured in the cluster node table										
0	Out of cluster node										
1	Master node										
2	Vice-master node										
3	In cluster node that is neither master nor vice-master.										
EXAMPLES	<p>The following example shows how to run <code>nhcmmrole</code> to determine the role of a node.</p> <p><b>EXAMPLE 1</b> Running <code>nhcmmrole</code> to determine if a node is the master</p> <ul style="list-style-type: none"> <li>■ Log in to a node.</li> <li>■ Run: <pre># <b>nhcmmrole</b> # echo \$? 1</pre> </li> </ul> <p>In this example, the current node is the master node.</p>										

(1m)



NAME	nhcmmstat - display information about peer nodes, trigger a switchover, or force the qualification of a master-eligible node
SYNOPSIS	<b>nhcmmstat</b> [-h] [-c <i>command</i> [-t]] [-n <i>nodeid</i> ]
DESCRIPTION	<p>The <b>nhcmmstat</b> tool displays information about a peer node or a group of peer nodes, displays notifications sent by the <b>nhcmmmd</b> daemon, and performs operations on the cluster. Use this tool at regular intervals when you are performing tasks that might change the status of a node.</p> <p>The <b>nhcmmstat</b> tool provides the following information:</p> <ul style="list-style-type: none"> <li>■ The node ID of a node</li> <li>■ That the cluster configuration files contain coherent information</li> <li>■ The role of a node</li> <li>■ The attributes of a node</li> <li>■ That the master and vice-master disks contain the same shared information</li> </ul> <p>You can use the <b>nhcmmstat</b> tool to modify the state of the cluster as shown in the examples section below.</p> <p>For information about the role and attributes of a node, see the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>
OPTIONS	<p>The following options are supported by <b>nhcmmstat</b>:</p> <ul style="list-style-type: none"> <li>-c        You can specify the <b>nhcmmstat</b> command to be executed. The specified command is executed and <b>nhcmmstat</b> exits. If this option is not used, you use the <b>nhcmmstat</b> command in an interactive mode. In this case, you must exit using the <b>exit</b> or <b>quit</b> command to return to the cursor.</li> <li>-h        Displays help.</li> <li>-n        You can specify the <i>nodeid</i> of the node on which you want to run <b>nhcmmstat</b>. This option is obligatory when using the <b>info</b> or <b>potential</b> commands.</li> <li>-t        Shows the start and end times.</li> </ul>
EXTENDED DESCRIPTION	<p>The following commands can be used with the <b>nhcmmstat</b> tool to get information about a single node:</p> <ul style="list-style-type: none"> <li><b>info</b>        Get information about a node in the cluster. You must provide the <i>nodeid</i> of the node.</li> <li><b>local</b>       Get the <i>nodeid</i> of the current node.</li> <li><b>master</b>       Get information about the master node.</li> </ul>

mynode	Get information about the current node.
potential	Get information about a node that is in the cluster node table but has the <code>CMM_OUT_OF_CLUSTER</code> role. You must provide the <i>nodeid</i> of the node.  When a node has the <code>CMM_OUT_OF_CLUSTER</code> role, the <code>nhcmmstat</code> tool gives meaningless values for the following administrative attributes: <code>CMM_ELIGIBLE_MEMBER</code> , <code>CMM_FLAG_DISQUALIFIED</code> , and <code>CMM_FLAG_SYNCHRO_NEEDED</code> .
stat_get	Get statistics about CMM API usage. For every variable, the current value, lowest value reached, highest value reached, and mean value are printed. The following variables are included in the output: <ul style="list-style-type: none"> <li>■ <code>clients_of_API</code>: Number of clients connected to CMM</li> <li>■ <code>calls_to_API</code>: Numbers of API calls that CMM received</li> <li>■ <code>notified_clients</code>: Number of clients connected to CMM and receiving notifications</li> <li>■ <code>switchovers</code>: Numbers of switch overs done</li> <li>■ <code>absolute_election</code>: Current election number (persistent across reboots)</li> <li>■ <code>relative_elections</code>: Number of elections performed</li> <li>■ <code>election_delay</code>: Length of election</li> <li>■ <code>up_time</code>: Uptime of CMM</li> <li>■ <code>cluster_size</code>: Number of nodes currently in the cluster</li> <li>■ <code>frame_time</code>: The time of the last Cluster View frame</li> <li>■ <code>allowed_failovers</code>: Number of fail-overs permitted by the direct link (split-brain-prevention serial cable)</li> <li>■ <code>denied_failovers</code>: Number of fail-overs denied by the direct link (split-brain-prevention serial cable)</li> <li>■ <code>dlink_status</code>: Status of the direct link: 1 = active, 0 = inactive</li> </ul>
timeout	Set a new timeout (in seconds) for the commands. The default value is 5 seconds.
vice	Get information about the vice-master node.

The following commands can be used with the `nhcmmstat` tool to get information about all peer nodes.

<code>all</code>	Get information about all peer nodes except those with the role <code>CMM_OUT_OF_CLUSTER</code> .
<code>count</code>	Get a count of the nodes in the cluster.

The following commands can be used with the `nhcmmstat` tool to modify the cluster. You must log in as superuser to use these commands.

<code>reload</code>	Force a reload of the <code>cluster_nodes_table</code> configuration. This command can be run from the master node only. The supported operations are add and remove a node in the <code>cluster_nodes_table</code> file, with the node powered off.  To add nodes that are not included in the original cluster definition, you must consider how the cluster was installed. For information about how to add new diskless node or dataless nodes to a cluster, see the <i>Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide</i> . You cannot add a master-eligible node to the cluster because there is a limit of two master-eligible nodes per cluster.
<code>so</code>	Force mastership change to the vice-master node. This command can be used on the master node only, and when the vice-master is present and able to take the master role.
<code>squalif</code>	Force the requalification for the current node to make it start an election. This command can be used on a master-eligible node when no peer node is qualified to be master. This command displays the following warning message:  <b>Warning!</b> This asynchronous command might take up to 300 s to succeed!

Note that an unsynchronized master-eligible node can only be elected as the master node if its former role was master.

The following commands can be used to exit from or get help with `nhcmmstat`:

<code>exit</code>	Exit.
<code>help</code>	Display help information.
<code>quit</code>	Exit.

If an `nhcmmstat` command fails, an error is displayed.

<b>Notifications</b>	When <code>nhcmmstat</code> is used in interactive mode, the following notifications are emitted by the <code>nhcmm</code> daemon and displayed.
<code>CMM_INVALID_CLUSTER</code>	"[USER CB] INVALID CLUSTER" is displayed. The cluster is not in a coherent state. No information can be returned by the CMM API until the cluster has been returned to a coherent state.
<code>CMM_VALID_CLUSTER</code>	"[USER CB] VALID CLUSTER" is displayed. The cluster is in a coherent state.
<code>CMM_STALE_CLUSTER</code>	"[USER CB] STALE CLUSTER" is displayed. The cluster has not had a master node for the preceding ten seconds.
<code>CMM_MASTER_DEMOTED</code>	"[USER CB] master demoted = %d" is displayed. %d is replaced by the <i>nodeid</i> of the demoted master.
<code>CMM_MASTER_ELECTED</code>	"[USER CB] master elected = %d" is displayed. %d is replaced by the <i>nodeid</i> of the new master. When this notification is received, all available information about the master is displayed.
<code>CMM_MEMBER_LEFT</code>	"[USER CB] member left cluster = %d" is displayed. %d is replaced by the <i>nodeid</i> of the node that has left the cluster.
<code>CMM_MEMBER_JOIN</code>	"[USER CB] new node in cluster = %d" is displayed. %d is replaced by the <i>nodeid</i> of the new node.
<code>CMM_VICEMASTER_DEMOTED</code>	"[USER CB] vice-master demoted = %d" is displayed. %d is replaced by the <i>nodeid</i> of the demoted vice-master node.
<code>CMM_VICEMASTER_ELECTED</code>	"[USER CB] vice-master elected = %d" is displayed. %d is replaced by the <i>nodeid</i> of the new vice-master. When this notification is received, all available information about the vice-master is displayed.
	When <code>nhcmmstat</code> is used in command line mode the notifications are not displayed.

**Node Information  
Displayed**

When information is requested for a node, or when notifications of cluster changes are received, the following information is displayed in the order shown:

nodeid	The <i>nodeid</i> of the current node, followed by "[This is the current node]" when the displayed information concerns the current node.
domain_id	The <i>domainid</i> of the cluster of the current node.
name	The name of the node as specified in the <code>/etc/opt/sun/nhas/cluster_nodes_table</code> file.
role	The role of the node in the cluster: master, vice-master, in, or out. The master, vice-master and out roles correspond to CMM_MASTER, CMM_VICEMASTER, and CMM_OUT_OF_CLUSTER. A node that does not have the CMM_OUT_OF_CLUSTER role, is <i>in</i> .
qualified	YES or NO is displayed. If YES, the node is qualified to be master. If NO, the node is not qualified to be master. This information is relevant for master-eligible nodes only.
synchro	NEEDED or READY is displayed. If NEEDED, the master and vice-master shared file systems do not contain the same information. If READY, the master and vice-master node file systems contain the same information. This information is relevant for master-eligible nodes only.
frozen	YES or NO is displayed. If YES, the node is frozen. When a node is frozen, the master cannot change the role of this node even if events require it. If NO, the node is not frozen.
excluded	YES or NO is displayed. If YES, the node is excluded from the cluster. An excluded node acts as if it has the CMM_OUT_OF_CLUSTER role. If NO, the node is not excluded.
eligible	YES or NO is displayed. If YES, this node can participate in an election and be elected master if it is sufficiently qualified. If NO, this node cannot participate in an election.
incarn	The incarnation number of the time that the node was last booted. The value is an integer (number of seconds since 00:00 universal coordinated time Jan 1 1970 ) and a literal representation of this date. For example, 1005833787 (15/11/2001 - 15:16:27).
swload_id	This string indicates the Foundation Services software version. The string 1 is displayed for the Foundation Services.
CGTP @	This is the address of the node of the cgtp0 interface

**USAGE**

You must log in as superuser to use the `so`, `reload`, and `squalif` commands of the `nhcmmstat` tool.

**EXAMPLES**

This section contains examples of how to use the `nhcmmstat` tool:

**EXAMPLE 1** Get Information About the Master Node

Log in to a peer node.

Type:

```
# nhcmmstat -c master
```

An output similar to the following is displayed:

```
-----
node_id      = 20
domain_id    = 250
name         = netraMEN2-cgtp0
role         = MASTER
qualified    = YES
synchro.     = READY
frozen       = NO
excluded     = NO
eligible     = YES
incarn.      = 1145893631 (24/04/2006 - 17:47:11)
swload_id    = NHAS3.0
CGTP @       = 10.25.3.26
init. election = 1678
-----
```

**EXAMPLE 2** Get the *nodeid* of the Current Node

Log in to a peer node.

This node becomes the current node.

Type:

```
# nhcmmstat -c local
```

An output similar to the following is displayed:

```
Local Node id is 10
```

In this example the *nodeid* of the current node is 10.

You can also find the *nodeid* of a node by using the `ifconfig` command. The *nodeid* corresponds to the host part of the nodes IP address. For more information, see the `ifconfig(8)` man page.



**EXAMPLE 3** Get Current, Minimum, Maximum, or Mean Values for Specified Variables

Type:

# **nhcmmstat** -c **stat\_get** -o master-cgtp

An output similar to the following is displayed:

Name	Cur.	Min	Max	Mean
clients_of_API	5	2	5	3
calls_to_API	249	3	249	174
notified_clients	3	1	4	2
switchovers	0	0	0	0
absolute_election	709	666	709	696
relative_elections	1	0	1	0
election_delay	0	0	0	0
up_time	88212	52	88212	61354
cluster_size	3	1	3	2
frame_time	1157532365	0	1157532365	67023207
allowed_failovers	0	0	0	0
denied_failovers	2	1	2	1
dlink_status	1	0	1	0

where *cur* is the current value, *Min* is the lowest value reached (not always applicable), *Max* is the highest value reached (not always applicable), *mean* is the mean value (not always applicable).

Values are provided in seconds, ticks from 1.1.1970, disabled (0), enabled (nonzero), and whole values (for example, the number of times an event occurs).

**EXAMPLE 4** Get Information About a Specific Node

Log in to a peer node.

Type:

# **nhcmmstat** -c **info** -n *nodeid*

An output similar to the following is displayed:

```

-----
node_id      = nodeid
domain_id    = 250
name         = netraMEN1-cgtp0
role         = VICE-MASTER
qualified    = YES
synchro.     = READY
frozen       = NO
excluded     = NO
eligible     = YES
incarn.      = 1145893631 (25/04/2006 - 15:40:50)
swload_id    = NHAS3.0
CGTP @       = 10.25.3.25
init. election = 1685
-----

```

#### EXAMPLE 5 Force the Qualification of a Node Asynchronously

Log in to a master-eligible node.

Type:

```
# nhcmmstat -c squalif
```

The `nhcmmstat` and `squalif` tool forces the requalification of the current node to make it the master node. This function can only be successful when there is no active master node in the cluster and the current node is a master-eligible node.

The `squalif` tool is asynchronous. The tool is not blocked while qualification is taking place.

#### EXAMPLE 6 Get Information About All Peer Nodes

Log in to any peer node.

Type:

```
# nhcmmstat -c all
```

Information about each peer node is printed to the console.

**EXAMPLE 7** Trigger a Switchover

Log in to the master node.

Type:

```
# nhcmmstat -c so
```

If there is a vice-master qualified to become master, it is elected master, the master becomes the vice-master, and the disks are synchronized. If there is no potential master, `nhcmmstat` does not perform a switchover.

**SEE ALSO**

`Intro(3cmm)`, `nhcmmd(8)`, `cluster_nodes_table(4)`, and `nhinstall(8)`.

(1m)



NAME	nhcrfsadm - command line tool for Reliable NFS administration
SYNOPSIS	<pre> /opt/sun/sbin/nhcrfsadm [ -hv ]  <b>IP-Based Replication Only:</b> /opt/sun/sbin/nhcrfsadm -a [ -s server ]  /opt/sun/sbin/nhcrfsadm -r [ -s server ]  /opt/sun/sbin/nhcrfsadm [ -s server ] -f all  /opt/sun/sbin/nhcrfsadm [ -s server ] -f partition-name  Shared Disk: No options.</pre>
DESCRIPTION	<p>The nhcrfsadm tool is a command line tool for the RNFS supervisory daemon administration.</p> <p>The nhcrfsadm tool performs the following tasks if IP-based replication is used:</p> <ul style="list-style-type: none"> <li>■ Authorizes or refuses the Reliable NFS to start on the vice-master node when a change of disk has been detected</li> <li>■ Performs a replication of one or all of the replicated partitions</li> </ul> <p>Replication must be authorized to start on the vice-master node in the following circumstances:</p> <ul style="list-style-type: none"> <li>■ The vice-master node has been stopped because of failure or maintenance</li> <li>■ A vice-master node disk containing a replicated partition has been changed</li> </ul> <p>The demand for authorization ensures that the new vice-master node disk is not corrupted.</p> <p>While the vice-master node is rebooting, the master node nhcrfsd daemon asks the operator whether the nhcrfsd daemon on the vice-master is allowed to follow the boot procedure or not. The nhcrfsd daemon on the vice-master node does not start until an order is given to the nhcrfsd daemon on the master node.</p>
OPTIONS	<p>The following options are supported (note which options are for IP-based replication only and which apply for IP replication and shared disk):</p> <p>-a (IP replication)</p> <p>Authorize replication to start on the vice-master node. This option can be used on the master node only. Replication will start automatically.</p> <p>-f all (IP replication)</p>

	<p>-f <i>partition-name</i> (IP replication)</p> <p>-h (Shared disk or IP replication)</p> <p>-r (IP replication)</p> <p>-s <i>server</i> (IP replication)</p> <p>-v (Shared disk or IP replication)</p>	<p>Perform a full replication of all partitions present in the RNFS configuration.</p> <p>Perform a full replication of the partition specified by <i>partition-name</i>. The partition must be specified by its name, for example, /dev/rdsk/c0t0d0s3.</p> <p>Display help on options.</p> <p>Refuse permission for replication to startup on the vice-master node. This option can be used on the master node only. The nhcrfsd daemon on the vice-master node will stop. The vice-master disk can then be removed, replaced, and rebooted.</p> <p>Specify the server whose nhcrfsd is to be administrated. The default server is <i>localhost</i>. The specified machine should always be the master node.</p> <p>Display the version and the build date of the tool.</p>				
USAGE	You must be logged on as superuser to run the nhcrfsadm command.					
EXIT STATUS	<table><tr><td>0</td><td>Command succeeded</td></tr><tr><td>1</td><td>Command failed</td></tr></table>		0	Command succeeded	1	Command failed
0	Command succeeded					
1	Command failed					
EXAMPLES	<p>The following examples demonstrates how nhcrfsadm is used.</p> <p>After you have changed a disk:</p> <ol style="list-style-type: none"><li>1. Log in to the master node as superuser.</li><li>2. Accept the start of replication by typing:</li></ol>					

**EXAMPLE 1** To Start Replication When Using IP-Based Replication

```
# /opt/sun/sbin/nhcrfsadm -a
```

**EXAMPLE 2** To Resynchronize the Master Node Disk and Vice-Master Node Disk When Using IP-Based Replication

1. Log in to the master node as superuser.
2. Resynchronize the master and vice-master disks.

```
# nhcrfsadm -f all
```

**SEE ALSO**

nhcrfsd(8)

(1m)





NAME	nhcrfsd - Reliable NFS supervisor daemon																		
SYNOPSIS	<b>opt/sun/sbin/nhcrfsd</b> [-h] [-v] [-u <i>URL</i> ]																		
DESCRIPTION	The nhcrfsd daemon manages the Reliable NFS feature of the Foundation Services. The nhcrfsd daemon is started when the system comes up (at init level 2). By default, the nhcrfsd daemon is monitored by the Daemon Monitor, nhpmd(8).																		
OPTIONS	<p>The following options are supported:</p> <table><tr><td>-h</td><td>Display help on options and exit.</td></tr><tr><td>-u <i>URL</i></td><td>You must specify the URL of the nhfs.conf file.</td></tr><tr><td>-v</td><td>Display the version and the build date of the daemon and exit.</td></tr></table>	-h	Display help on options and exit.	-u <i>URL</i>	You must specify the URL of the nhfs.conf file.	-v	Display the version and the build date of the daemon and exit.												
-h	Display help on options and exit.																		
-u <i>URL</i>	You must specify the URL of the nhfs.conf file.																		
-v	Display the version and the build date of the daemon and exit.																		
EXTENDED DESCRIPTION	<p>You must be logged on as superuser to run the nhcrfsd daemon.</p> <p>The nhcrfsd daemon uses the nhfs.conf file and the /etc/vfstab file.</p>																		
The nhfs.conf File	<p>The following parameters must be set in the nhfs.conf file:</p> <ul style="list-style-type: none"><li>■ RNFS.Slice</li><li>■ RNFS.NFSAlternatePath</li><li>■ RNFS.Share</li></ul> <p>For more information about these parameters, see the nhfs.conf(5) man page.</p>																		
The /etc/vfstab File	<p>Entries in vfstab use the syntax described in vfstab(4). For each master-eligible node, the vfstab entry must contain the following:</p> <ul style="list-style-type: none"><li>■ Information for mounting slices managed by Reliable NFS so that they can be exported by the NFS server.</li><li>■ Information for mounting the file systems to be exported by NFS.</li></ul> <p>The following paths are used by applications:</p> <ul style="list-style-type: none"><li>■ Mount slices managed by Reliable NFS locally (on the master-eligible nodes).</li></ul> <table><tr><td>/dev/drbd0</td><td>/SUNWcgha/local</td><td>ext3</td><td>noauto</td><td>0</td><td>0</td></tr><tr><td>/dev/drbd1</td><td>/export</td><td>ext3</td><td>noauto</td><td>0</td><td>0</td></tr><tr><td>/dev/drbd1</td><td>/export</td><td>ext3</td><td>noauto</td><td>0</td><td>0</td></tr></table> <p>This example mounts the Reliable NFS replicated partition /dev/drbd1 on /export and /dev/drbd0 on /SUNWcgha/local/.</p> <p>The RNFS.Share property allows /export to be exported by NFS from the master node.</p>	/dev/drbd0	/SUNWcgha/local	ext3	noauto	0	0	/dev/drbd1	/export	ext3	noauto	0	0	/dev/drbd1	/export	ext3	noauto	0	0
/dev/drbd0	/SUNWcgha/local	ext3	noauto	0	0														
/dev/drbd1	/export	ext3	noauto	0	0														
/dev/drbd1	/export	ext3	noauto	0	0														

	<div>■ Mount the Reliable NFS replicated slices by NFS (on the master-eligible nodes).</div> <div>10.28.3.1:/SUNWcggha/local/export - /SUNWcggha/remote nfs - no fg,hard,intr,noac</div> <div>This example mounts /SUNWcggha/local/export of the master node on the /SUNWcggha/remote mount point of the master and vice-master nodes.</div> <div>The specified mount options must be selected according to the type of mount entry.</div> <div>You can use the noac option if the impact on performance is acceptable. For information about how to enable and disable the noac option, see the <i>Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide</i>.</div>
EXIT STATUS	<div>The exit status for nhcrfsd is one of the following:</div> <div>0           Daemon started successfully</div> <div>1           Daemon failed to start</div>
FILES	<div>/etc/opt/sun/nhas/nhfs.conf</div> <div>Configuration and addressing information for the different Foundation Services. The URL for this file could be file:///etc/opt/sun/nhas/nhfs.conf.</div>
NOTES	<div>All nhcrfsd daemon messages are logged with the syslog facility. This facility uses message priorities of the facility.level form. The nhcrfsd daemon uses the local0 facility and a level that can be alert, err, or info.</div>
SEE ALSO	<div>nhfs.conf(5), nhcrfsadm(8), nhpmd(8), and syslog.conf(5).</div>

<b>NAME</b>	nheamd - Foundation Services external access management
<b>SYNOPSIS</b>	<b>/opt/sun/sbin/nheamd</b> -h   -u <i>URL</i>
<b>DESCRIPTION</b>	<p>The External Access Manager (EAM) is implemented using the nheamd daemon. An nheamd daemon exists on each peer node.</p> <p>The daemon has two main functions: monitoring bonding groups and reacting to role change notifications to make floating addresses available on only the master node.</p> <p>When bonding is configured on the node, if a monitored group fails on the master node, a switchover will occur to allow the vice master to become the master node, thereby achieving higher availability.</p> <p>On master-eligible nodes, when a node becomes the master, the nheamd daemon will bring UP all the floating addresses under its management. When a node becomes the vice master, those addresses will be brought DOWN.</p> <p>Requests to terminate the daemon with a signal TERM will be ignored.</p>
<b>OPTIONS</b>	<p>-h                      Displays the help messages.</p> <p>-u                      You must specify the URL of the nhfs.conf file.</p>
<b>EXIT STATUS</b>	<p>The following exit values for nheamd are returned:</p> <p>0                      Daemon completed successfully or terminated by SIGTERM</p> <p>1                      An error occurred</p>
<b>FILES</b>	<p>/etc/opt/sun/nhas/nhfs.conf</p> <p>Configuration and addressing information for the different Foundation Services. The URL for this file could be <code>file:///etc/opt/sun/nhas/nhfs.conf</code>.</p>
<b>SEE ALSO</b>	nhfs.conf(5)

(1m)



NAME	nhenablesync - trigger disk synchronization when using IP-based replication.
SYNOPSIS	<code>/opt/sun/sbin/nhenablesync</code>
DESCRIPTION	<p>The <code>nhenablesync</code> tool enables you to trigger disk synchronization. By default, disk synchronization starts automatically during the boot sequence. You can delay the start of synchronization by setting the <code>RNFS.EnableSync</code> property to <code>False</code> in the <code>nhfs.conf</code> file. If the start of synchronization has been delayed in this way, trigger synchronization by logging into the master node and executing the <code>nhenablesync</code> command as follows:</p> <pre># <b>nhenablesync</b></pre> <p><b>Note</b> – The <code>nhenablesync</code> tool has no affect when it is used in a shared disk configuration.</p>
SEE ALSO	<code>cluster_nodes_table(5)</code> , <code>nhfs.conf(5)</code>

(1m)



NAME	nhinstall - Foundation Services installation and configuration tool
SYNOPSIS	<pre>/opt/sun/sbin/nhinstall -h</pre> <pre>/opt/sun/sbin/nhinstall -r <i>directory</i> [ -l <i>logfile</i> ] [ <i>stage</i> ]</pre>
DESCRIPTION	<p>The <code>nhinstall</code> tool enables you to install and configure the operating system and the Foundation Services on all the nodes of your cluster.</p> <p>Before running the <code>nhinstall</code> tool, configure it using the following configuration files:</p> <ul style="list-style-type: none"> <li>■ The <code>env_installation.conf</code> file to define the installation environment. See <code>env_installation.conf(5)</code>.</li> <li>■ The <code>cluster_definition.conf</code> file to define the cluster environment. See <code>cluster_definition.conf(5)</code>.</li> <li>■ (Optional) The <code>addon.conf</code> file specifying additional patches and packages that you want to install. This file is useful for upgrading Foundation Services at a later stage. See <code>addon.conf(5)</code>.</li> <li>■ (Optional) The <code>nodeprof.conf</code> file permits the customization of the operating system installation. See <code>nodeprof.conf(5)</code>.</li> <li>■ (Optional) The <code>dataless_nodeprof.conf</code> file permits the customization of the operating system installation on dataless nodes. See <code>dataless_nodeprof.conf(5)</code>.</li> <li>■ (Optional) The <code>diskless_nodeprof.conf</code> file permits the customization of the operating system installation on diskless nodes. See <code>diskless_nodeprof.conf(5)</code>.</li> </ul> <p>To install the operating system and the Foundation Services on the cluster, type the following command as a superuser on the installation server:</p> <pre># /opt/sun/sbin/nhinstall -r <i>config-file-directory</i></pre> <p>where <i>config-file-directory</i> is the directory containing the configuration files.</p> <p>The <code>nhinstall</code> tool also supports a recovery mechanism based on a progress indicator. In case of a failure during installation, you can restart the installation at the point where the failure occurred by running the same command.</p>
OPTIONS	<pre>-h</pre> <p>Help.</p> <pre>-r <i>directory</i></pre>

	Path to the directory containing the configuration files.
<code>-l logfile</code>	Name of a log file. If you specify a log file, the output is recorded in the file in addition to being displayed in the console. In case of an error, the logfile helps to trace the error and to identify the point at which the installation will restart.
<code>stages</code>	Specify the action you require:
<code>reset</code>	Reset the progress indicator to force the next installation to restart from the beginning.
<code>add nodeID nodeID ...</code>	Add new dataless and diskless nodes to a cluster that is already running. <i>nodeID</i> is the ID of the new dataless or diskless node as defined in the <code>cluster_definition.conf</code> file. Before you run the add stage: <ul style="list-style-type: none"> <li>■ Define the new dataless and diskless nodes in the <code>cluster_definition.conf</code> file by using the <code>NODE</code> parameter or by using the <code>DATALESS</code> and <code>DISKLESS</code> parameters.. For more information, see the <b>cluster_definition.conf(5)</b> man page.</li> </ul> <hr/> <p><b>Caution</b> – Define only the new dataless and diskless nodes that are to be added to the cluster. Do not define nodes that do not exist and that you might want to add to the cluster in the future. If you do so, the <code>nhinstall</code> tool will fail during installation.</p> <hr/> <ul style="list-style-type: none"> <li>■ Execute the <code>nhinstall</code> command with the <code>reset</code> stage.</li> </ul>

## EXAMPLES

In the following examples, the configuration files are located in the `/home/nhasconf` directory on the installation server.

**EXAMPLE 1** To Run the `nhinstall` Command to Start an Installation

```
# /opt/sun/sbin/nhinstall -r /home/nhasconf
```

**EXAMPLE 2** To Reset the Progress Indicator to Restart an Installation From the Beginning

```
# /opt/sun/sbin/nhinstall -r /home/nhasconf reset
```



**EXAMPLE 3** To Add a New Diskless Node to An Existing Cluster

```
# /opt/sun/sbin/nhinstall -r /home/nhasconf reset
# /opt/sun/sbin/nhinstall -r /home/nhasconf add 40
```

Where 40 is the *nodeID* of the new diskless node.

**SEE ALSO**

```
addon.conf(5), cluster_definition.conf(5),
dataless_nodeprof.conf(5), diskless_nodeprof.conf(5),
env_installation.conf(5), nodeprof.conf(5)
```

(1m)



NAME	nhnsmd - Node State Manager daemon
SYNOPSIS	<code>/opt/sun/sbin/nhnsmd [-u URL]</code>
DESCRIPTION	<p>The Node State Manager (NSM) uses the Cluster Membership Manager (CMM) notifications to determine when the node on which it runs is promoted to or demoted from being the master or vice-master node. When the NSM is notified that a node has changed its role, it executes the corresponding script provided by the user (if one has been provided).</p>
OPTIONS	<p>The nhnsmd daemon takes the following option:</p> <p><code>-u URL</code>                      You must specify the URL of the <code>nhfs.conf</code> file.</p>
EXTENDED DESCRIPTION	<p>Launch the nhnsmd manually as a superuser:</p> <pre># ./nhnsmd -u URL</pre> <p>The nhnsmd daemon is started at system boot time after the nhcmmmd daemon. The nhnsmd daemon registers to receive the following notifications at cluster startup:</p> <ul style="list-style-type: none"> <li>■ CMM_MASTER_ELECTED</li> <li>■ CMM_MASTER_DEMOTED</li> <li>■ CMM_VICEMASTER_ELECTED</li> <li>■ CMM_VICEMASTER_DEMOTED</li> </ul> <p>The nhnsmd daemon executes a response to notifications in the order in which it receives them. It does not act upon a notification that does not pertain to the current state of the node for which the notification is received.</p> <p>The nhnsmd daemon maintains persistent state across failures so that when restarted by the Daemon Monitor it can determine whether it has missed any notifications and can take appropriate action. This persistent state is not maintained across a node reboot.</p> <p>The scripts executed by the nhnsmd daemon must be executable shell scripts.</p> <p>The first argument of the scripts is the <i>action</i> parameter. The action parameter can have two values, <i>enter state</i> and <i>leave state</i>. You can use the same script for both actions. For an <i>enter state</i> action, the script passes the string <i>enter</i> as the first argument. For a <i>leave state</i> action, the script is passed the string <i>leave</i> as the first argument.</p> <p>The second argument of the scripts is the <i>node role</i> parameter. The <i>node role</i> parameter is passed as a lower-case character string. The <i>node role</i> parameter can have two values, <i>master</i> and <i>vice-master</i>.</p>

The scripts executed by the `nhnsmd` daemon should not perform actions that change the startup behavior of the node. For any notification received, the `nhnsmd` daemon executes a script that invokes a node to leave an existing state and then a script that invokes that node to enter a new state.

Scripts used by the `nhnsmd` daemon run as asynchronous processes and do not take account of any changes in the cluster state. When writing your own action scripts for the `nhnsmd` daemon do not write scripts that will take a long time to execute or that depend on cluster behavior. Such scripts should not be used as a way of controlling applications or as a replacement for a management framework.

**FILES**

<code>nhfs.conf</code>	Configuration and addressing information for the different Foundation Services. The URL for this file could be <code>file:///etc/opt/SUNWcgha/sun/nhas/nhfs.conf</code>
<code>/opt/sun/actions/master</code>	This directory contains scripts for transitions to and from the master state.
<code>/opt/sun/actions/vicemaster</code>	This directory contains scripts for transitions to and from the vice-master state.

The script names have the `Ennnxxxxxx` or `Lnnnnxxxxxx` form, where `E` denotes an script to enter a state and `L` denotes a script to leave a state, `nn` is a two-digit numeric code, and `xxxxxx` is an arbitrary string of characters. Any files that do not use this naming scheme will be ignored by `nhnsmd`.

**SEE ALSO**

`nhfs.conf`(5) and `nhpmd`(8).

NAME	nhpmd - Process Monitor Daemon (PMD)
SYNOPSIS	<code>/opt/sun/sbin/amd64/nhpmd</code> , or <code>/opt/sun/sbin/nhpmd</code> (on 32-bit x64 machines)
DESCRIPTION	<p>The <code>nhpmd</code> daemon provides the Daemon Monitor service. The <code>nhpmd</code> daemon runs at the multiuser level on all nodes in the cluster. The <code>nhpmd</code> daemon monitors other Foundation Services daemons and may monitor many operating system daemons and some companion product daemons. If a daemon that provides a critical service fails, the <code>nhpmd</code> daemon detects the failure and triggers a recovery response. The recovery response is specific to the daemon that failed.</p> <p>The <code>nhpmd</code> daemon operates at a higher priority than the other Foundation Services daemons.</p> <p>Foundation Services daemons and operating system daemons are launched by a <i>startup script</i>. A <i>nametag</i> is assigned to the daemon or group of daemons that is launched by each startup script. In some cases, a <i>nametag</i> is assigned to only one daemon, however, it can also be assigned to a group of daemons. If one of the daemons covered by a <i>nametag</i> fails, the recovery response is performed by the <code>nhpmd</code> daemon on all of the daemons covered by that <i>nametag</i>. If the recovery response is to restart the failed daemon, all of the daemons grouped under that <i>nametag</i> are killed and restarted. For a list of monitored daemons and their associated recovery responses, see <code>MONITORED DAEMONS</code>.</p> <p>Information about monitored daemons can be collected using the <code>nhpmdadm</code> command, as described in the <code>nhpmdadm(8)</code> man page. This man page lists the Foundation Services, operating system, and companion product daemons that are monitored by the <code>nhpmd</code> daemon and describes the recovery action taken by the <code>nhpmd</code> daemon on the node on which the monitored daemon failed.</p>
EXTENDED DESCRIPTION	<p>Note the following before using the <code>nhpmd</code> daemon:</p> <ul style="list-style-type: none"> <li>■ The initialization process of the Foundation Services uses <code>/etc/init.d/rc.HA</code> instead of the default <code>/etc/init.d/rc</code> script. Do not modify or overwrite <code>rc.HA</code>.</li> <li>■ The <code>nhpmd</code> daemon server is started automatically when the system starts up at init level 2 (multi-user mode).</li> <li>■ Files in the <code>/var/run/sun/nhas/pmd</code> directory, and the directory itself, must not be removed while the <code>nhpmd</code> daemon is running.</li> <li>■ The script provided as an action program to any <code>nhpmdadm</code> command must not be removed; it must exist when the <code>nhpmd</code> daemon attempts to execute it. If the system is out of main resources (memory or processes), the <code>nhpmd</code> daemon might not be able to launch or relaunch any executables.</li> <li>■ To avoid collisions with other controlling processes, <code>strace(1)</code> does not allow a process to be traced that it detects as being controlled by another process by way of the <code>ptrace(2)</code> system call. The <code>nhpmd</code> daemon uses the <code>/proc</code> interface to</li> </ul>

- monitor processes and their descendents, therefore, those processes that are submitted to the nhpmd daemon using the nhpmdadm tool cannot be traced or debugged.
- When you list the processes that are running on the Foundation Services, you see the Foundation Services daemons. Some of the daemons delivered with the Foundation Services are part of the Foundation Services internal subsystem and cannot be publicly accessed. Some daemons run only on the master and vice-master nodes, and some run on all peer nodes.

**MONITORED  
DAEMONS**

The following lists give the nametag and associated recovery response of the daemons that are monitored by the nhpmd daemon. The recovery responses listed are the default values. You can specify the number of times the nhpmd daemon tries to restart a daemon if you create the nhpmd.conf file. For a description of these daemons, see their man pages. For information about the nhpmd.conf file, see the nhpmd.conf (5) man page.

**Monitored  
Daemons in the  
Foundation  
Services**

The following list gives the nametag and recovery response of the monitored daemons in the Foundation Services.

Daemon - nhcrfsd	Nametag - nhcrfsd
	Recovery response - relauches the daemon up to three times. In some cases, the nhcrfsd daemon detects a fatal error and reboots the node. If either of the following commands are executed, the daemon is terminated and nhpmd does not relauch it: kill -15 <pid nhcrfsd> OR kill -TERM <pid nhcrfsd>
Daemon - nhcrfscIntd	Nametag - nhcrfscIntd
	Recovery response - relauches the daemon up to three times.

**DIAGNOSTICS**

Daemon - nhcmmd

Nametag - nhcmmd

Recovery response - does not restart the daemon; reboots the node on which it failed.

Daemon - nheamd

Nametag - nheamd

Recovery response - relaunches the daemon up to two times.

Daemon - nhprobed

Nametag - cgha\_probe

Recovery response - does not restart the daemon; reboots the node on which it failed

Diagnostic messages are logged to the console or in a file, depending on the system's syslog local0 facility settings.

Diagnostic messages conform to the following rules:

- Tags are enclosed in lower-upper brackets (< and >), for example: <nhcmmd>.
- Diagnostic messages related to a specific tag adhere to these formats:

- <tag>: *error message*

- *Sentence*

For example:

---

```
<nhcmmd>: rebooting node because tag is declared as critical
```

---



---

```
<nhcmmd>: failed to stay up
```

---



---

```
PMD is not restarting tag <nma> because limit for number of exit 0 has been reached
```

---

- Generic diagnostic messages (that is, those not specific to a tag) adhere to this format:

- *Sentence*

For example:

---

```
Must be root to start nhpmd
```

---

- Internal diagnostic messages (typically requiring support from your service representative) adhere to this format:

- *function*: *error message*

(1m)

For example:

---

fork: failed

---

**SEE ALSO**

nhpmdadm(8)



NAME	nhpmdadm - process monitor facility administration
SYNOPSIS	<pre> /opt/sun/sbin/nhpmdadm -c nametag [-a action [-r]] [-e ENV_VAR= \ env.var...] [-E] [-n retries] [-t period] [-C level#] [-D daemon name...] \ [-T daemon name...] [-S] [-N limit] &lt;command&gt; [args-to-command...] /opt/sun/sbin/nhpmdadm -m nametag [-n retries] [-t period] /opt/sun/sbin/nhpmdadm -s nametag [-w timeout] [signal] /opt/sun/sbin/nhpmdadm -k nametag [-w timeout] [signal] /opt/sun/sbin/nhpmdadm -l nametag /opt/sun/sbin/nhpmdadm -q nametag /opt/sun/sbin/nhpmdadm -L </pre>
DESCRIPTION	<p>The <code>nhpmdadm</code> tool provides the administrative command-line interface to the process monitor facility, <code>nhpmd(8)</code>.</p> <p>The process monitor facility provides a means of monitoring processes, and their descendants, and restarting them if they fail to remain alive. The total number of failures allowed can be specified, and limited to a specific time period. After the maximum number of failures has occurred within the specified time period, a message is logged to the console, and the process is no longer restarted.</p> <p>If an action program has been specified, it is called when the number of failures allowed has been reached. If the action program exits with non-zero status, the process <code>nametag</code> is removed from the process monitor facility. Otherwise, the process is restarted with the original parameters passed into <code>nhpmdadm</code>.</p> <p>Processes that are started under control of the process monitor are run as the uid of the user that initiated the request. Only the original user, or root, can manipulate the <code>nametag</code> associated with those processes. Status information, however, is available to any caller.</p> <p>All spawned processes, and their descendent spawned processes, of the process that initially started are monitored, unless explicitly specified. Only when the last process/sub-process exits does the process monitor attempt to restart the process, unless daemon grouping is specified.</p> <p>See the <code>nhpmd</code> man page for more information about this daemon.</p>

**OPTIONS**

The following options are supported:

**-a *action***

The action program to be called when the process fails to stay alive. This program must be specified in a single argument to the -a flag, but can be a quoted string that contains multiple components. In either case, the string is executed as specified, with two additional arguments, the event that occurred (currently only failed), and the nametag associated with the process. The current directory, and PATH environment variable, are reinstantiated before the command is executed. No other environment variables are preserved.

If the action program exits with status 0, the process is started over again with the original arguments that were given to nhpmdadm. Any other exit status causes the nametag to cease to exist within the scope of the process monitor. If no -a *action* is specified, the result is the same as if there were an action script specified, which always exits non-zero.

If an action script is specified, it is possible to add a criticality property using the "-r" flag. When criticality is specified and the PMD fails to execute the action script (for example, because it lacks memory), the PMD will reboot the node immediately. This is useful for critical daemons.

If an action script is specified, it is possible to add a delay property using the -A *delay* flag. When a delay is specified and the action script takes more time to complete than the period of time defined (in seconds) by *delay*, the PMD will abort the action script execution and consider the action script as "having failed" (exit status != 0). In case criticality was specified, this will make the PMD reboot the node immediately.

**-A *delay***

Refer to -a *action* for information. The period of time defined by *delay* is in seconds.

**-c *nametag***

Start a process, with nametag as an identifier. All arguments that follow the command-line flags are executed as the process of interest. The current directory, and PATH environment variable, are reinstantiated by the process monitor facility before the command is executed. No other environment variables are, or should be assumed to be, preserved.

If nametag already exists, nhpmdadm exits with exit status 1, with no side effects.

I/O redirection is not supported in the command line arguments. If this is necessary, a script should be created that performs this redirection, and used as the command that nhpmdadm executes.

#### *-C level#*

When starting a process, monitor it and its children up to and including level *level#*. The value of *level#* must be an integer greater than or equal to zero. The original process executed is at level 0, its children are executed at level 1, their children are executed at level 2, and so on. Any new fork operation produces a new level of children.

This option provides more control over which processes get monitored. It is useful for monitoring servers that fork new processes. When this option is not specified, all children are monitored, and the original process is not restarted until it and all its children have died.

If a server forks new processes to handle client requests, it might be desirable to monitor only the server. The server needs to be restarted if it dies even if some client processes are still running. The appropriate monitoring level is *-C 0*. If, after forking a child, the parent exits, then it is the child that needs monitoring. The level to use to monitor the child is *-C 1*. When both processes die, the server is restarted.

#### *-D daemon\_name*

When set to a daemon name which is spawned as part of the command (usually a startup script launching more than one daemon) and that daemon dies or exits with a non-zero exit value, every daemon monitored in the nametag will be killed by PMD before any restart is attempted.

#### *-e ENV\_VAR= env.value*

	<p>An environment variable in the form <i>ENV_VAR=env.value</i> that is passed to the execution environment of the new process. This option can be repeated, so multiple environment variables can be passed. The default is not to use this option, in which case the <i>nhpmd</i> environment plus the path of the <i>nhpmdadm</i> environment are passed.</p>
-E	<p>Pass the whole <i>nhpmdadm</i> environment to the new process. The default is not to use this option, in which case the <i>nhpmd</i> environment plus the path of the <i>nhpmdadm</i> environment are passed.</p> <p>The <i>-e</i> and <i>-E</i> options are mutually exclusive, that is, both cannot be used in the same command.</p>
-k <i>nametag</i>	<p>Send the specified signal to the processes associated with <i>nametag</i>, including any processes associated with the action program if it is currently running. The default signal, <i>SIGKILL</i>, is sent if none is specified. If the process and its descendants exit, and there are remaining retries available, the process monitor restarts the process. The signal specified is the same set of names recognized by the <i>kill(1)</i> command.</p>
-l <i>nametag</i>	<p>Prints out status information about <i>nametag</i>. The output from this command is useful mainly for diagnostic purposes.</p>
-L	<p>Returns a list of all tags currently running that belong to the user that issued the command or, if the user is superuser, all tags running on the server.</p> <p>For a list of the <i>nametags</i> and the daemons to which they correspond, see the <i>nhpmd(8)</i> man page.</p>
-m <i>nametag</i>	<p>Modify the number of retries, or time period over which to observe retries, for <i>nametag</i>. Once these parameters have been changed, the history of earlier failures is cleared.</p>
-M <i>delay</i>	

	<p>When starting a process, in case the <code>-M delay</code> flag is specified and the process starting takes more time to complete than the period of time defined (in seconds) by <code>delay</code>, the PMD will abort the process startup and consider the process as "having failed."</p>
<code>-n retries</code>	<p>Number of retries allowed within the specified time period. The default value for this field is 0, which means that the process is not restarted once it exits. The maximum value allowed is 100. A value of -1 indicates that the number of retries is infinite..</p>
<code>-N limit</code>	<p>Number of allowed actions exiting with an exit status of 0. The default value for this flag is -1, which equates to infinity. If this parameter is specified and the action script has already executed and exited with an exit status of 0 that number of times, then the action script is no longer executed and the process will not restart again.</p>
<code>-q nametag</code>	<p>Indicate whether nametag is registered and running under the process monitor. Returns 0 if it is, 1 if it is not. Other return values indicate an error.</p>
<code>-s nametag</code>	<p>Stop restarting the command associated with nametag. The signal, if specified, is sent to all processes, including the action script and its processes if they are currently executing. If a signal is not specified, none is sent. Stopping the monitoring of processes does not imply that they no longer exist. The processes remain running until they, and all of their descendents, have exited. The signal specified is the same set of names recognized by the <code>kill(1)</code> command.</p>
<code>-S</code>	<p>If set, nhpmdadm will block until the process has been successfully scheduled in the system. The default behavior is to have nhpmdadm enqueue the process and return immediately.</p>
<code>-t period</code>	<p>Minutes over which to count failures. The default value for this flag is -1, which equates to infinity. If this parameter is specified, process failures that have occurred outside of the specified period are not counted.</p>
<code>-T daemon_name</code>	

The same as `-D`, but no kill is effectively performed. Only a message is printed on the console.

`-w timeout`

When used in conjunction with the `-s nametag` or `-k nametag` flags, wait up to the specified number of seconds for the processes associated with `nametag` to exit. If the timeout expires, `nhpmdadm` exits with exit status 2. The default value for this flag is 0, meaning that the command returns immediately without waiting for any process to exit.

If a value of -1 is given, `nhpmdadm` waits indefinitely for the processes associated with the tag to exit. The `nhpmdadm` process does not release the RPC server thread that it uses until the RPC timeout period is reached. Therefore, avoid setting the `-w timeout` value to -1 unnecessarily.

**Note** – The options `-l` and `-L` are supported for troubleshooting purposes. They can be used to get information about monitored processes. Do not kill or stop `nhpmdadm -L` or `nhpmdadm -l nametag` commands.

## EXAMPLES

This section provides examples of using the `nhpmdadm` command.

### EXAMPLE 1 How to Use `nhpmdadm`

- To get all nametags:

```
# nhpmdadm -L
```

Result:

```
tags: nhcrfscslntd nhcrfsd nhcmmd nhprobed
```

- To get detailed information about a specific nametag, for example, `nhcrfsd`:

```
# nhpmdadm -l nhcrfsd
# nhpmdadm -c nhcrfsd -n 3 -a /etc/opt/sun/nhas/init.d/
crfs.launcher.HA.fail /etc/opt/sun/nhas/init.d/crfs.launcher.pmd
start
environment:
...
PATH=/usr/sbin:/usr/bin
TZ=MET
...
=/opt/sun/sbin/nhpmdadm
retries: 0
```

```

owner: root
monitor children: up to level 3
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206
pids: 2206

```

### EXAMPLE 2 To Verify a Daemon Is Being Monitored

- Log in to a node, as superuser.
- Select a daemon to investigate.
- Confirm that the daemon is running.

```
$ pgrep -x daemon-name
```

- Note the process ID for the daemon.
- Find the *nametag* for the daemon:

```
# /opt/sun/sbin/nhpmddadm -L
```

Alternatively, use the tables in the **nhadm(8)** man page to find the Daemon Monitor *nametag* that corresponds to the daemon that you want to investigate.

- Using the daemon *nametag*, run:

```
# /opt/sun/sbin/nhpmddadm -l nametag
```

A list of process IDs is displayed for the Daemon Monitor *nametag*.

- Confirm that the process ID entry for this daemon in the list is the same as the process ID returned by the `pgrep` command.
- If this is the case, the daemon is being monitored. If not, the daemon is not being monitored.

### EXAMPLE 3 To Start a Sleep Process That Will Not be Restarted

The following example starts a sleep process named `sleep.once` that will not be restarted once it exits:

```
# /opt/sun/sbin/nhpmadm -c sleep.once /bin/sleep 5
```

**EXAMPLE 4** To Start a Sleep Process and Restart It

The following example starts a sleep process and restarts it, at most, one time:

```
# /opt/sun/sbin/nhpmadm -c sleep.twice -n 1 /bin/sleep 5
```

**EXAMPLE 5** To Start a Sleep Process and Restart It

The following examples start a sleep process and restarts it, at most, twice per minute. It calls `/bin/true` when it fails to remain running beyond the acceptable number of failures:

```
# /opt/sun/sbin/nhpmadm -c sleep.forever -n 2 -t 1 -a \
/bin/true /bin/sleep 60
```

**EXAMPLE 6** To List the Current Status of the `sleep.forever` Nametag

The following command lists the current status of the `sleep.forever` nametag:

```
# /opt/sun/sbin/nhpmadm -c sleep.forever -n 2 -t 1 -a \
/bin/true /bin/sleep 60
```

**EXAMPLE 7** To Send a SIGHUP to All Processes

The following command sends a SIGHUP to all processes associated with `sleep.forever`, waiting up to five seconds for all processes to exit.

```
# /opt/sun/sbin/nhpmadm -w 5 -k sleep.forever HUP
```

**EXAMPLE 8** To Stop Monitoring the Processes and Sending a SIGHUP

The following command stops monitoring (restarting) processes associated with `sleep.forever`, and sends a SIGHUP to any processes related to it. This command returns as soon as the signals have been delivered, but possibly before all processes have exited.

```
# /opt/sun/sbin/nhpmadm -s sleep.forever HUP
```



**EXAMPLE 9** To List Tags Running That Belong to the User

If a user issues the following commands:

```
# /opt/sun/sbin/nhpmddadm -c sleep.once /bin/sleep 30
# /opt/sun/sbin/nhpmddadm -c sleep.twice /bin/sleep 60
# /opt/sun/sbin/nhpmddadm -c sleep.forever /bin/sleep 90
```

typing the following command:

```
# /opt/sun/sbin/nhpmddadm -L
```

produces the following output:

```
sleep.once sleep.twice sleep.forever
```

**EXIT STATUS**

- |    |   |
|----|---|
| <0 | An error occurred.  |
| 0  | The command was completed successfully.   |
| 1  | nametag doesn't exist, or there was an attempt to create a nametag that already exists. |
| 2  | The command timed out.  |

**MESSAGE LISTS**

Log file outputs from the nhpmddadm tool are made through stderr.

The following is a list of messages that are output to the log files by the nhpmddadm daemon.

- <tagname> No such <nametag> registered  
The specified nametag is not recognized.
- Missing command argument  
The nametag argument is missing from the -l option.
- Too many command line arguments  
There are too many options specified in the command line.

**SEE ALSO**

nhpmd(8)

(1m)



NAME	nhprobed - test accessibility of physical interfaces
SYNOPSIS	<code>/opt/sun/sbin/nhprobed [-h] [-u URL]</code>
DESCRIPTION	<p>Each peer node runs a daemon, <code>nhprobed</code>, that periodically sends a heartbeat in the form of an IP packet. The <code>nhprobed</code> daemon sends the heartbeats by multicast.</p> <p>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 node or one of its physical interfaces is considered to have failed. If both of the node's physical interfaces fail, the node itself is considered to have failed. Heartbeats are broadcast at the rate of 3 per 900 milliseconds, and at least one heartbeat must be detected each 900 milliseconds.</p> <p>On the master-eligible nodes, the <code>nhprobed</code> daemon receives a list of nodes from the <code>nhcmmnd</code> daemon. The <code>nhprobed</code> daemon monitors the heartbeats of the nodes on the list. On the master node, the list contains all of the master-ineligible nodes and the vice-master node. On the vice-master node, the list contains the master node only.</p> <p>On the master-eligible nodes, the <code>nhprobed</code> daemon notifies the <code>nhcmmnd</code> daemon when, for any node on its list, any of the following events occur:</p> <ul style="list-style-type: none"> <li>■ One link becomes available, indicating that the node is accessible through the link.</li> <li>■ One link becomes unavailable, indicating that the node is not accessible through the link.</li> <li>■ The node becomes available, indicating that the first link to the node becomes available.</li> <li>■ The node becomes unavailable, indicating that the last available link to the node becomes unavailable.</li> </ul>
EXTENDED DESCRIPTION	<p>The <code>nhprobed</code> daemon uses a kernel module, <code>ffmod</code>, and a kernel driver, <code>hbxdrv</code>. The kernel module and kernel driver manage the send and receive mechanisms of the <code>nhprobed</code> daemon. Because the kernel module and kernel driver operate in the kernel space, the <code>nhprobed</code> daemon is more robust against system overload.</p> <p><b>Note</b> – The probe heartbeat of the Foundation Services 3.0 is incompatible with any previous Netra HA Suite probe heartbeat. In general, Foundation Services 3.0 can't be used with nodes running previous versions of the Foundation Services. All nodes in a cluster must have the same probe installation. For information about how to install the packages for the <code>nhprobed</code> daemon, the kernel module and kernel driver, see the <i>Netra High Availability Suite 3.0 1/08 Foundation Services Manual Installation Guide for the Solaris OS</i>.</p>

The `nhcmmd` daemon and `nhprobed` daemon communicate through an Internet socket, `AF_INET`, in connection oriented mode. The socket port number can be configured in the `/etc/services` file, using the service name `cmm_cgtp_probe`.

**OPTIONS**

The `nhprobed` daemon takes the following options:

- `-h`                      Displays help information.
- `-u URL`                You must specify the URL of the `nhfs.conf` file.

For more information, see the `nhfs.conf(5)` man page.

**EXIT STATUS**

The following exit values are returned:

- 0                      Successful completion.
- 1                      An error occurred.

**FILES**

`URL`                    Common configuration file. The file that contains configuration and addressing information for the individual Foundation Services.

The URL for this file could, for example, be  
`file:///etc/opt/sun/nhas/nhfs.conf`

**SEE ALSO**

`Intro(3cmm)`, `nhcmmd(8)`, and `nhfs.conf(5)`

NAME	nhsched - Display the scheduling parameters of the Foundation Services processes								
SYNOPSIS	<pre>/opt/sun/sbin/nhsched -a /opt/sun/sbin/nhsched -h /opt/sun/sbin/nhsched -i pid process-name /opt/sun/sbin/nhsched -u URL</pre>								
DESCRIPTION	The nhsched command displays the scheduling parameters of the Foundation Services processes.								
OPTIONS	<p>The nhsched command can be used with the following parameters:</p> <table><tr><td>-a</td><td>Display the current scheduling base priority configuration.</td></tr><tr><td>-i pid process-name</td><td>Display the scheduling parameters for the specified process.</td></tr><tr><td>-h</td><td>Display help information.</td></tr><tr><td>-u URL</td><td>Specify an alternative URL for the nhfs.conf file. If you do not use this option, the URL file:///etc/opt/sun/nhas/nhfs.conf is used.</td></tr></table>	-a	Display the current scheduling base priority configuration.	-i pid process-name	Display the scheduling parameters for the specified process.	-h	Display help information.	-u URL	Specify an alternative URL for the nhfs.conf file. If you do not use this option, the URL file:///etc/opt/sun/nhas/nhfs.conf is used.
-a	Display the current scheduling base priority configuration.								
-i pid process-name	Display the scheduling parameters for the specified process.								
-h	Display help information.								
-u URL	Specify an alternative URL for the nhfs.conf file. If you do not use this option, the URL file:///etc/opt/sun/nhas/nhfs.conf is used.								
EXIT STATUS	<p>The exit status for the nhsched command is one of the following:</p> <table><tr><td>0</td><td>Success</td></tr><tr><td>1</td><td>Error</td></tr></table>	0	Success	1	Error				
0	Success								
1	Error								
EXAMPLES	<p>This section gives examples of how to use the nhsched command.</p> <p><b>EXAMPLE 1</b> Get Information About the Current Configuration</p> <p>The maximum and minimum values are retrieved from system information.</p> <p>Display the scheduling parameters for the nhpmd process:</p> <p><b>EXAMPLE 2</b> Display the Scheduling Parameters for a Process</p> <pre># /opt/sun/sbin/nhsched -i nhpmd process id          : 62</pre>								

(1m)

```
process name      : nhpmd
scheduling policy  : RR
scheduling priority : 40
scheduling priority : 40
scheduling priority : 40
scheduling priority : 40
```

**SEE ALSO** `nhfs.conf(5)`

NAME	nhscsitool - Foundation Services configuration tool
SYNOPSIS	<code>[/opt/sun/sbin/]nhscsitool [-h] [-p] &lt;device&gt;</code>
DESCRIPTION	<p>This tool enables the administrator to remove any remaining SCSI3 PGR key or SCSI2 reservation from a disk. It is usually used when installing a cluster and when attempts to read from or write to a disk (for example, using the commands <code>fdisk</code>, <code>sfdisk</code>, or <code>parted</code>) result in error messages "Input/output error."</p> <p>Usually, "Input/output error" means that a SCSI3 PGR key is still on the disk or that a SCSI2 reservation was issued by the other node sharing the disk before being shut down.</p> <p><b>Note</b> – This tool should be used only while in non-clustered mode.</p>
OPTIONS	<p><code>-h</code> Prints the help message.</p> <p><code>-p</code> Prints the SCSI3 PGR keys on the disk, but does not remove them. Without this option, the tool removes any SCSI3 PGR key on the disk.</p> <p><i>device</i> The shared disk's device name (for example, <code>/dev/sdb</code>).</p>
EXAMPLES	<p>The following example describes how an administrator could reinstall the cluster from scratch after a cluster running the Foundation Services was improperly shut down.</p> <p><b>EXAMPLE 1</b> To Run the <code>nhscsitool</code> Command</p> <pre># sfdisk -l /dev/sdb Disk /dev/sdb: 1112 cylinders, 255 heads, 63 sectors/track read: Input/output error  sfdisk: read error on /dev/sdb - cannot read sector 0 /dev/sdb: unrecognized partition  No partitions found No partitions found No partitions found No partitions found No partitions found No partitions found No partitions found No partitions found No partitions found No partitions found No partitions found</pre>

The preceding screen shows that the shared disk /dev/sdb cannot be seen by the node. After using the nhscsitool command, as shown in the following example, the disk geometry appears again and you can modify the disk.

```
# opt/sun/sbin/nhscsitool /dev/sdb
Performing a SCSI bus reset...done
There are no keys on disk '/dev/sdb'
# echo $?
0
# sfdisk -l /dev/sdb
Disk /dev/sdb: 1112 cylinders, 255 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting from 0

Device      Boot Start      End    #cyls   #blocks   Id  System
/dev/sdb1           0+      122     123-    987966    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
/dev/sdb2          123      245     123     987997+    83   Linux
```

**SEE ALSO**    nhfs.conf(5), nhcrfsd(8)



NAME	cmm_cmc_filter - define notification filtering																
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_cmc_filter(cmm_cmcfiler_t const action, cmm_cmchanges_t const * const notifications_list, uint32_t const notifications_count);</pre>																
DESCRIPTION	<p>The <code>cmm_cmc_filter()</code> function defines the list of notifications sent to an application that is registered to receive Cluster Membership Manager notifications. An application registers to receive notifications by calling the <code>cmm_cmc_register(3cmm)</code> function. By default, when an application calls this function, the application receives a notification for every change that occurs in the cluster state. An application can operate when viewing a subset of the notifications. By using the <code>cmm_cmc_filter()</code> function, an application defines the list of notifications that it receives.</p> <p><b>Note</b> – For information on the notification sequences and the various scenarios, see the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>																
PARAMETERS	<p>The <code>cmm_cmc_filter()</code> function takes the following parameters:</p> <table> <tr> <td><i>action</i></td><td>Specifies the action to be performed on the current filter (add, remove, set).</td></tr> <tr> <td><i>notifications_list</i></td><td>An array of <code>cmm_cmchanges_t</code>. This array represents the modifications to be applied to the current filter.</td></tr> <tr> <td><i>notifications_count</i></td><td>Specifies the number of elements in <i>notifications_list</i>.</td></tr> </table> <p>■ The <i>action</i> parameter is set to:</p> <table> <tr> <td>CMM_CMC_NOTIFY_REM</td><td>To remove some notifications from the set currently in the filter.</td></tr> <tr> <td>CMM_CMC_NOTIFY_ADD</td><td>To receive a given set of notifications in addition to the set currently in the filter.</td></tr> <tr> <td>CMM_CMC_NOTIFY_SET</td><td>To define a completely new set of notifications.</td></tr> <tr> <td>CMM_CMC_NOTIFY_ALL</td><td>To receive all notifications.</td></tr> <tr> <td>CMM_CMC_NOTIFY_NONE</td><td>To receive no notifications. This is not a way to remove a registration. Use <code>cmm_cmc_unregister()</code> to stop <code>nhcmmmd</code> sending notifications.</td></tr> </table>	<i>action</i>	Specifies the action to be performed on the current filter (add, remove, set).	<i>notifications_list</i>	An array of <code>cmm_cmchanges_t</code> . This array represents the modifications to be applied to the current filter.	<i>notifications_count</i>	Specifies the number of elements in <i>notifications_list</i> .	CMM_CMC_NOTIFY_REM	To remove some notifications from the set currently in the filter.	CMM_CMC_NOTIFY_ADD	To receive a given set of notifications in addition to the set currently in the filter.	CMM_CMC_NOTIFY_SET	To define a completely new set of notifications.	CMM_CMC_NOTIFY_ALL	To receive all notifications.	CMM_CMC_NOTIFY_NONE	To receive no notifications. This is not a way to remove a registration. Use <code>cmm_cmc_unregister()</code> to stop <code>nhcmmmd</code> sending notifications.
<i>action</i>	Specifies the action to be performed on the current filter (add, remove, set).																
<i>notifications_list</i>	An array of <code>cmm_cmchanges_t</code> . This array represents the modifications to be applied to the current filter.																
<i>notifications_count</i>	Specifies the number of elements in <i>notifications_list</i> .																
CMM_CMC_NOTIFY_REM	To remove some notifications from the set currently in the filter.																
CMM_CMC_NOTIFY_ADD	To receive a given set of notifications in addition to the set currently in the filter.																
CMM_CMC_NOTIFY_SET	To define a completely new set of notifications.																
CMM_CMC_NOTIFY_ALL	To receive all notifications.																
CMM_CMC_NOTIFY_NONE	To receive no notifications. This is not a way to remove a registration. Use <code>cmm_cmc_unregister()</code> to stop <code>nhcmmmd</code> sending notifications.																

- The *notifications\_list* parameter is the set of notifications. In the case of CMM\_CMC\_NOTIFY\_ALL and CMM\_CMC\_NOTIFY\_NONE, this argument is ignored and not tested.
- The *notifications\_count* parameter is the number of elements contained in *notifications\_list*. This must be a positive integer. If CMM\_CMC\_NOTIFY\_ALL and CMM\_CMC\_NOTIFY\_NONE are used with the *action* parameter, this argument is ignored and not tested.

An application can call the `cmm_cmc_filter()` function as many times as needed. The changes to the filter take effect when the call returns successfully. The filter is evaluated in `cmm_notify_dispatch(3cmm)`, so you must define it before calling this dispatching function. An application calling `cmm_cmc_register()` after `cmm_cmc_filter()` does not receive unsolicited notifications.

**RETURN  
VALUES**

The `cmm_cmc_filter()` function returns one of the following values:

- CMM\_EINVAL      Invalid argument such as *notification\_count* is a NULL or *action* is not valid.
- CMM\_OK            Operation succeeds.

See `attributes(5)` for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	SPARC and x64
Availability	SUNWnhas-cmm-headers
Interface Stability	Evolving
MT-Level	MT-Safe
Cancel-Safety	Deferred-Cancel-Safe Asynchronous-Cancel-Unsafe

**SEE ALSO**

`Intro(3cmm)`, `nhcmmnd(8)`, `cmm_cmc_register(3cmm)`,  
`cmm_notify_getfd(3cmm)`, `cmm_notify_dispatch(3cmm)`

<b>NAME</b>	cmm_cmc_register - register to receive notifications; remove registration and stop receiving notifications				
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  typedef struct {     cmm_cmchanges_t cmchange;     cmm_nodeid_t nodeid; } cmm_cmc_notification_t;  typedef void (*cmm_notify_t)     (const cmm_cmc_notification_t *change_notification,      void *client_data);  cmm_error_t cmm_cmc_register(cmm_notify_t const callback, void * client_data);  cmm_error_t cmm_cmc_unregister( );</pre>				
<b>DESCRIPTION</b>	The <code>cmm_cmc_register()</code> function enables a system service or application to receive change notifications by registering the callback function indicated.				
<b>PARAMETERS</b>	<p>The <code>cmm_cmc_register()</code> function takes the following parameters:</p> <table> <tr> <td><i>callback</i></td><td>A pointer to a callback function defined by the service or application.</td></tr> <tr> <td><i>client_data</i></td><td>A parameter used by the callback function. Its type and value are defined by the registered service or application. It is passed as an argument to the callback and it is valid in the calling process address space. No sanity check is run on this parameter, as <code>nhcmmnd(8)</code> does not know its meaning.</td></tr> </table>	<i>callback</i>	A pointer to a callback function defined by the service or application.	<i>client_data</i>	A parameter used by the callback function. Its type and value are defined by the registered service or application. It is passed as an argument to the callback and it is valid in the calling process address space. No sanity check is run on this parameter, as <code>nhcmmnd(8)</code> does not know its meaning.
<i>callback</i>	A pointer to a callback function defined by the service or application.				
<i>client_data</i>	A parameter used by the callback function. Its type and value are defined by the registered service or application. It is passed as an argument to the callback and it is valid in the calling process address space. No sanity check is run on this parameter, as <code>nhcmmnd(8)</code> does not know its meaning.				
<b>EXTENDED DESCRIPTION</b>	<p>If the <code>cmm_cmc_register()</code> function is called while a callback is registered, a <code>CMM_EEXIST</code> error is returned because only one function can be registered at a time. To change the registration, <code>cmm_cmc_unregister()</code> must be called prior to registering the new node with <code>cmm_cmc_register()</code>.</p> <p>Registration only needs to be done once to receive cluster membership change notifications. When a process attempts to register more than once, the first callback is kept and an error is returned.</p> <p>The calling process must use the <code>cmm_notify_getfd(3cmm)</code> and <code>cmm_notify_dispatch(3cmm)</code> functions to receive and dispatch messages from <code>nhcmmnd(8)</code>. An application defines the list of notifications it receives by calling <code>cmm_cmc_filter(3cmm)</code>.</p>				

Note that the order of callback notifications is the same as that of the Cluster Membership Manager (CMM) notifications; one call to `cmm_notify_dispatch(3cmm)` can lead to several calls to the callback (in fact as many as the number of pending notifications). Within these callbacks, if functions are invoked concerning the state of the cluster (for instance to get the number of nodes in the cluster), the results of the functions do not refer to the state that the cluster is in when the notification has been generated. Instead, the results of the functions refer to the state of the cluster when the function is invoked. In the meantime, some other modifications might have occurred in the cluster.

The callback function is invoked by the same thread as the one that calls the `cmm_notify_dispatch(3cmm)` function. The function is invoked by a library linked to the process. The library communicates with the CMM API that supplies the membership change information passed as an argument to the callback function.

The `cmm_cmc_unregister()` function removes the calling process's registration so that no further delivery of cluster membership change notifications is made. Only the process on which a callback is called can remove the caller's registration. A child or a parent process cannot do this.

If the calling process callback function is active when the unregister request is made, it is not canceled.

In case of `fork()`, the created child process does not inherit the registration from its parent. It has to make its own registration.

**RETURN  
VALUES**

The `cmm_cmc_register()` function returns one of the following values:

<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.
<code>CMM_ECONN</code>	No <code>nhcmmnd</code> is currently accessible on the local node.
<code>CMM_ENOENT</code>	The maximum number of clients that can register with <code>cmm_cmc_register</code> at any one time has been reached.
<code>CMM_EEXIST</code>	The calling process has already registered a callback.
<code>CMM_ENOTSUP</code>	Unexpected service error. The cluster might be in a critical state.
<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.
<code>CMM_OK</code>	Operation succeeds.

The `cmm_cmc_unregister()` function returns one of the following values:

<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.
	Another possible meaning of <code>CMM_EBUSY</code> is that a call is made to the <code>cmm_cmc_unregister()</code> function when the calling process's callback function is active.
<code>CMM_ECONN</code>	There is no <code>nhcmmd</code> currently accessible on the local node.
<code>CMM_ENOENT</code>	The registration does not exist.
<code>CMM_ENOTSUP</code>	Unexpected service error. The cluster might be in a critical state.
<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.
<code>CMM_OK</code>	Operation succeeds.

#### SEE ALSO

`nhcmmd(8)`, `fork(2)`, `cmm_cmc_filter(3cmm)`,  
`cmm_notify_dispatch(3cmm)`, `cmm_notify_getfd(3cmm)`

(3cmm)



NAME	cmm_cmc_unregister - register to receive notifications; remove registration and stop receiving notifications				
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  typedef struct {     cmm_cmchanges_t    cmchange;     cmm_nodeid_t nodeid; } cmm_cmc_notification_t;  typedef void (*cmm_notify_t)     (const cmm_cmc_notification_t *change_notification,      void *client_data);  cmm_error_t cmm_cmc_register(cmm_notify_t  const callback, void * client_data);  cmm_error_t cmm_cmc_unregister( );</pre>				
DESCRIPTION	The <code>cmm_cmc_register()</code> function enables a system service or application to receive change notifications by registering the callback function indicated.				
PARAMETERS	<p>The <code>cmm_cmc_register()</code> function takes the following parameters:</p> <table><tr><td><i>callback</i></td><td>A pointer to a callback function defined by the service or application.</td></tr><tr><td><i>client_data</i></td><td>A parameter used by the callback function. Its type and value are defined by the registered service or application. It is passed as an argument to the callback and it is valid in the calling process address space. No sanity check is run on this parameter, as <b>nhcmmmd(8)</b> does not know its meaning.</td></tr></table>	<i>callback</i>	A pointer to a callback function defined by the service or application.	<i>client_data</i>	A parameter used by the callback function. Its type and value are defined by the registered service or application. It is passed as an argument to the callback and it is valid in the calling process address space. No sanity check is run on this parameter, as <b>nhcmmmd(8)</b> does not know its meaning.
<i>callback</i>	A pointer to a callback function defined by the service or application.				
<i>client_data</i>	A parameter used by the callback function. Its type and value are defined by the registered service or application. It is passed as an argument to the callback and it is valid in the calling process address space. No sanity check is run on this parameter, as <b>nhcmmmd(8)</b> does not know its meaning.				
EXTENDED DESCRIPTION	<p>If the <code>cmm_cmc_register()</code> function is called while a callback is registered, a <code>CMM_EEXIST</code> error is returned because only one function can be registered at a time. To change the registration, <code>cmm_cmc_unregister()</code> must be called prior to registering the new node with <code>cmm_cmc_register()</code>.</p> <p>Registration only needs to be done once to receive cluster membership change notifications. When a process attempts to register more than once, the first callback is kept and an error is returned.</p> <p>The calling process must use the <b>cmm_notify_getfd</b>(3cmm) and <b>cmm_notify_dispatch</b>(3cmm) functions to receive and dispatch messages from <b>nhcmmmd(8)</b>. An application defines the list of notifications it receives by calling <b>cmm_cmc_filter</b>(3cmm).</p>				

Note that the order of callback notifications is the same as that of the Cluster Membership Manager (CMM) notifications; one call to **cmm\_notify\_dispatch**(3CMM) can lead to several calls to the callback (in fact as many as the number of pending notifications). Within these callbacks, if functions are invoked concerning the state of the cluster (for instance to get the number of nodes in the cluster), the results of the functions do not refer to the state that the cluster is in when the notification has been generated. Instead, the results of the functions refer to the state of the cluster when the function is invoked. In the meantime, some other modifications might have occurred in the cluster.

The callback function is invoked by the same thread as the one that calls the **cmm\_notify\_dispatch**(3cmm) function. The function is invoked by a library linked to the process. The library communicates with the CMM API that supplies the membership change information passed as an argument to the callback function.

The **cmm\_cmc\_unregister()** function removes the calling process's registration so that no further delivery of cluster membership change notifications is made. Only the process on which a callback is called can remove the caller's registration. A child or a parent process cannot do this.

If the calling process callback function is active when the unregister request is made, it is not canceled.

In case of **fork()**, the created child process does not inherit the registration from its parent. It has to make its own registration.

**RETURN  
VALUES**

The **cmm\_cmc\_register()** function returns one of the following values:

CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.
CMM_ECONN	No nhcmmnd is currently accessible on the local node.
CMM_ENOENT	The maximum number of clients that can register with <b>cmm_cmc_register()</b> at any one time has been reached.
CMM_EEXIST	The calling process has already registered a callback.
CMM_ENOTSUP	Unexpected service error. The cluster might be in a critical state.
CMM_ETIMEOUT	The call timeout expired before the action was completed.
CMM_OK	Operation succeeds.



The `cmm_cmc_unregister()` function returns one of the following values:

<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.  Another possible meaning of <code>CMM_EBUSY</code> is that a call is made to the <code>cmm_cmc_unregister()</code> function when the calling process's callback function is active.
<code>CMM_ECONN</code>	There is no <code>nhcmmd</code> currently accessible on the local node.
<code>CMM_ENOENT</code>	The registration does not exist.
<code>CMM_ENOTSUP</code>	Unexpected service error. The cluster might be in a critical state.
<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.
<code>CMM_OK</code>	Operation succeeds.

#### SEE ALSO

`nhcmmd(8)`, `fork(2)`, `cmm_cmc_filter(3cmm)`,  
`cmm_notify_dispatch(3cmm)`, `cmm_notify_getfd(3cmm)`

(3cmm)



NAME	cmm_config_reload - reload the cluster node table								
SYNOPSIS	<pre>cc [ flag... ] file...  lcgha_cmm  -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_config_reload( );</pre>								
DESCRIPTION	<p>The <code>cmm_config_reload()</code> function forces the <code>nhcmmnd(8)</code> daemon to reload the cluster node table. When a node is added to or removed from this table, the <code>nhcmmnd</code> daemon must be informed. There are two ways to inform the <code>nhcmmnd</code> daemon of the changes to the cluster node table: call the <code>cmm_config_reload()</code> function or type one of the following:</p> <pre>pkill  -HUP nhcmmnd kill   -HUP &lt;CMM process Id&gt;  nhcmmstat -c reload</pre> <p>This function can only be called from the master node. If it is called from a non-master node it returns a <code>CMM_EPERM</code> error. As a result of this call, notifications are sent indicating the modifications occurring in the cluster because of the new configuration read from the cluster node table. See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information on notifications. The supported operations are add a node and remove a node, with the node powered off. The attributes of a node must not be changed.</p> <p>To remove a node from the cluster, ensure that the node is powered off before changing the cluster nodes table. Only after a node has been powered off should you perform a <code>cmm_config_reload()</code>.</p>								
RETURN VALUES	<p>The <code>cmm_config_reload()</code> function returns one of the following values:</p> <table><tr><td>CMM_EBUSY</td><td>The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr><tr><td>CMM_ECONN</td><td>No <code>nhcmmnd</code> is accessible on the current node.</td></tr><tr><td>CMM_ENOCLUSTER</td><td>Calling node is not yet in a cluster.</td></tr><tr><td>CMM_ENOTSUP</td><td>Unexpected service error.</td></tr></table>	CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.	CMM_ECONN	No <code>nhcmmnd</code> is accessible on the current node.	CMM_ENOCLUSTER	Calling node is not yet in a cluster.	CMM_ENOTSUP	Unexpected service error.
CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.								
CMM_ECONN	No <code>nhcmmnd</code> is accessible on the current node.								
CMM_ENOCLUSTER	Calling node is not yet in a cluster.								
CMM_ENOTSUP	Unexpected service error.								

(3cmm)

**SEE ALSO**

CMM_EPERM	Permission denied. The function was not called from the master node.
CMM_ETIMEOUT	The call timeout expired before the action was completed.
CMM_OK	Operation succeeds.
Intro(3cmm), nhcmmnd(8)	

NAME	cmm_connect - prepare or test a connection to the Cluster Membership Manager (CMM)										
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_connect(const struct timespec timeout);</pre>										
DESCRIPTION	<p>The <code>cmm_connect()</code> function is implicit in the first call to the Cluster Membership Manager (CMM) Application Programming Interface (API). You do not need to call this function to create a CMM connection; but use this function to test the availability of a CMM connection, or to set the timeout value. The default timeout is five seconds.</p> <p>The <i>timeout</i> parameter is globally used by the CMM API to signify the maximum amount of time for which a call can block. The type of the timeout is a <i>timespec_t</i> and the value must be greater than 0 seconds, 0 nanoseconds. Note that if the value of the timeout is too short, you risk being unable to use the CMM API. This is because every call would fail since the timeout would be expired before the call finished.</p> <p><b>Note</b> – This function is not related to cluster information; therefore, can be called from any node - even a node that is not part of a cluster.</p>										
RETURN VALUES	<p>The <code>cmm_connect()</code> function returns one of the following values:</p> <table><tr><td>CMM_ECONN</td><td>No <code>nhcmmnd(8)</code> is currently accessible on the local node.</td></tr><tr><td>CMM_EINVAL</td><td>The given <i>timeout</i> is invalid.</td></tr><tr><td>CMM_ENOTSUP</td><td>Unexpected service error occurs.</td></tr><tr><td>CMM_ETIMEDOUT</td><td>Fails to connect before the previous timeout expired.</td></tr><tr><td>CMM_OK</td><td>Operation succeeds.</td></tr></table> <p>This call never returns CMM_ENOCLUSTER</p>	CMM_ECONN	No <code>nhcmmnd(8)</code> is currently accessible on the local node.	CMM_EINVAL	The given <i>timeout</i> is invalid.	CMM_ENOTSUP	Unexpected service error occurs.	CMM_ETIMEDOUT	Fails to connect before the previous timeout expired.	CMM_OK	Operation succeeds.
CMM_ECONN	No <code>nhcmmnd(8)</code> is currently accessible on the local node.										
CMM_EINVAL	The given <i>timeout</i> is invalid.										
CMM_ENOTSUP	Unexpected service error occurs.										
CMM_ETIMEDOUT	Fails to connect before the previous timeout expired.										
CMM_OK	Operation succeeds.										
SEE ALSO	<code>Intro(3cmm)</code> , <code>nhcmmnd(8)</code>										

(3cmm)



NAME	cmm_disconnect - close a connection between the current calling process and the nhcmmnd daemon						
SYNOPSIS	<pre>cc [ flag... ] file... lcgha_cmm lrt #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_disconnect( );</pre>						
DESCRIPTION	<p>The <code>cmm_disconnect()</code> function closes the connection between the current calling process and the <code>nhcmmnd</code> daemon. This frees the resources allocated to the client connection. Notifications are no longer managed by the library. If notifications were registered before this function was called, they are no longer sent.</p> <p>The connection is automatically re-established and resources reallocated when a function that needs the connection is called. However, the configuration of the notifications (callback function, filters, etc) is not recreated. You must reconfigure the notification registration.</p> <p><b>Note</b> – If an application or service calls <code>cmm_disconnect()</code> when a <code>cmm_notify_dispatch()</code> call is being executed, the <code>cmm_notify_dispatch()</code> call is terminated.</p>						
RETURN VALUES	<p>The <code>cmm_disconnect()</code> function returns one of the following values:</p> <table><tr><td>CMM_ENOTSUP</td><td>Unexpected service error or no local <code>nhcmmnd</code> is accessible.</td></tr><tr><td>CMM_ETIMEOUT</td><td>The call timeout expired before the action was completed.</td></tr><tr><td>CMM_OK</td><td>Operation succeeds.</td></tr></table>	CMM_ENOTSUP	Unexpected service error or no local <code>nhcmmnd</code> is accessible.	CMM_ETIMEOUT	The call timeout expired before the action was completed.	CMM_OK	Operation succeeds.
CMM_ENOTSUP	Unexpected service error or no local <code>nhcmmnd</code> is accessible.						
CMM_ETIMEOUT	The call timeout expired before the action was completed.						
CMM_OK	Operation succeeds.						
SEE ALSO	<code>Intro(3cmm)</code> , <code>nhcmmnd(8)</code> , <code>cmm_connect(3cmm)</code>						

(3cmm)





NAME	cmm_master_getinfo - retrieve information about the master node or the vice-master node																			
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_master_getinfo(cmm_member_t * const member);  cmm_error_t cmm_vicemaster_getinfo(cmm_member_t * const member);</pre>																			
DESCRIPTION	<p>The <code>cmm_master_getinfo()</code> function returns information about the current master node. The <code>cmm_vicemaster_getinfo()</code> function returns information about the current vice-master node. The information returned by these two functions has the same type and meaning as that returned by the <code>cmm_member_getinfo()</code> function.</p> <p>The <i>member</i> parameter is a pointer to a member structure where the function stores the member's information, such as its current state.</p> <p>See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information on the <code>cmm_member_t</code> structure.</p>																			
RETURN VALUES	<p>The <code>cmm_master_getinfo()</code> and <code>cmm_vicemaster_getinfo()</code> functions return one of the following values:</p> <table><tr><td>CMM_EAGAIN</td><td>The information might be deprecated, because a node has been out of communication with the master node for a period of time.</td></tr><tr><td>CMM_EBUSY</td><td>The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr><tr><td>CMM_ECONN</td><td>No <code>nhcmmnd</code> is currently accessible on the local node.</td></tr><tr><td>CMM_EINVAL</td><td>Invalid parameter. <i>member</i> is a NULL pointer.</td></tr><tr><td>CMM_ENOCLUSTER</td><td>The calling node is not yet in a cluster.</td></tr><tr><td>CMM_ENOTSUP</td><td>An unexpected service error occurred. The cluster might be in a critical state.</td></tr><tr><td>CMM_ESRCH</td><td>No such member. This return value is only applicable for the vice-master node.</td></tr><tr><td>CMM_ETIMEOUT</td><td>The call timeout expired before the action was completed.</td></tr><tr><td>CMM_OK</td><td>Operation succeeds.</td></tr></table>		CMM_EAGAIN	The information might be deprecated, because a node has been out of communication with the master node for a period of time.	CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.	CMM_ECONN	No <code>nhcmmnd</code> is currently accessible on the local node.	CMM_EINVAL	Invalid parameter. <i>member</i> is a NULL pointer.	CMM_ENOCLUSTER	The calling node is not yet in a cluster.	CMM_ENOTSUP	An unexpected service error occurred. The cluster might be in a critical state.	CMM_ESRCH	No such member. This return value is only applicable for the vice-master node.	CMM_ETIMEOUT	The call timeout expired before the action was completed.	CMM_OK	Operation succeeds.
CMM_EAGAIN	The information might be deprecated, because a node has been out of communication with the master node for a period of time.																			
CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.																			
CMM_ECONN	No <code>nhcmmnd</code> is currently accessible on the local node.																			
CMM_EINVAL	Invalid parameter. <i>member</i> is a NULL pointer.																			
CMM_ENOCLUSTER	The calling node is not yet in a cluster.																			
CMM_ENOTSUP	An unexpected service error occurred. The cluster might be in a critical state.																			
CMM_ESRCH	No such member. This return value is only applicable for the vice-master node.																			
CMM_ETIMEOUT	The call timeout expired before the action was completed.																			
CMM_OK	Operation succeeds.																			

(3cmm)

**SEE ALSO** | `Intro(3cmm), cmm_member_getinfo(3cmm)`

NAME	cmm_mastership_release - trigger a switchover																
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_mastership_release( );</pre>																
DESCRIPTION	<p>The <code>cmm_mastership_release()</code> function triggers a switchover. This function must be called from the master node. If a service or application attempts to call this function from another node, <code>CMM_EPERM</code> is returned.</p> <p>If the vice-master node is qualified to be master when <code>cmm_mastership_release()</code> is called, then this node becomes master. The calling node remains master until the vice-master node has taken the master role.</p> <p>If no node is qualified to become master when the <code>cmm_mastership_release()</code> function is called, <code>CMM_ECANCELED</code> is returned.</p> <p>See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information on the notifications returned in different scenarios.</p> <p>Any program on the master node can execute this function. No authentication is performed.</p>																
RETURN VALUES	<p>The <code>cmm_mastership_release()</code> function returns one of the following values:</p> <table><tr><td><code>CMM_EBUSY</code></td><td>The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr><tr><td><code>CMM_ECANCELED</code></td><td>Operation cancelled. There was no vice-master to take the master role.</td></tr><tr><td><code>CMM_ECONN</code></td><td>No <code>nhcmmnd</code> is currently accessible on the local node.</td></tr><tr><td><code>CMM_ENOCLUSTER</code></td><td>Not in cluster.</td></tr><tr><td><code>CMM_ENOTSUP</code></td><td>An unexpected service error occurred.</td></tr><tr><td><code>CMM_EPERM</code></td><td>Permission denied as the function was not called from a master node.</td></tr><tr><td><code>CMM_ETIMEOUT</code></td><td>The timeout expired before the action was completed.</td></tr><tr><td><code>CMM_OK</code></td><td>Operation succeeds.</td></tr></table>	<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.	<code>CMM_ECANCELED</code>	Operation cancelled. There was no vice-master to take the master role.	<code>CMM_ECONN</code>	No <code>nhcmmnd</code> is currently accessible on the local node.	<code>CMM_ENOCLUSTER</code>	Not in cluster.	<code>CMM_ENOTSUP</code>	An unexpected service error occurred.	<code>CMM_EPERM</code>	Permission denied as the function was not called from a master node.	<code>CMM_ETIMEOUT</code>	The timeout expired before the action was completed.	<code>CMM_OK</code>	Operation succeeds.
<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.																
<code>CMM_ECANCELED</code>	Operation cancelled. There was no vice-master to take the master role.																
<code>CMM_ECONN</code>	No <code>nhcmmnd</code> is currently accessible on the local node.																
<code>CMM_ENOCLUSTER</code>	Not in cluster.																
<code>CMM_ENOTSUP</code>	An unexpected service error occurred.																
<code>CMM_EPERM</code>	Permission denied as the function was not called from a master node.																
<code>CMM_ETIMEOUT</code>	The timeout expired before the action was completed.																
<code>CMM_OK</code>	Operation succeeds.																

(3cmm)

**SEE ALSO** | `Intro(3cmm)`, `cmm_membership_remove(3cmm)`,  
`cmm_member_setqualif(3cmm)`

<b>NAME</b>	cmm_member_getall - retrieve information on the cluster						
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_member_getall(uint32_t const  table_size, cmm_member_t * const member_table, uint32_t * const member_count);  cmm_error_t cmm_member_getcount(uint32_t * const member_count);</pre>						
<b>DESCRIPTION</b>	<p>The <code>cmm_member_getall()</code> function fills <i>member_table</i> with information about all nodes in the cluster. There is a table entry for each node. The information in this table is of the same type and meaning as that returned by <code>cmm_member_getinfo()</code>. If <i>member_table</i> is a null pointer, <code>cmm_member_getall()</code> behaves like the <code>cmm_member_getcount()</code> function.</p> <p>The <code>cmm_member_getcount()</code> function returns the number of nodes in the cluster, including the node from which the function is called. The value is stored in the area pointed to by <i>member_count</i>. See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for further information on the state of the node.</p>						
<b>PARAMETERS</b>	<p>The <code>cmm_member_getall()</code> function takes the following parameters:</p> <table> <tr> <td><i>table_size</i></td><td>Specifies the maximum number of entries in <i>member_table</i>. The maximum number of entries is 1024.</td></tr> <tr> <td><i>member_table</i></td><td>A pointer to an array of structures where the requested information is placed.</td></tr> <tr> <td><i>member_count</i></td><td>Specifies the number of nodes in the cluster.</td></tr> </table>	<i>table_size</i>	Specifies the maximum number of entries in <i>member_table</i> . The maximum number of entries is 1024.	<i>member_table</i>	A pointer to an array of structures where the requested information is placed.	<i>member_count</i>	Specifies the number of nodes in the cluster.
<i>table_size</i>	Specifies the maximum number of entries in <i>member_table</i> . The maximum number of entries is 1024.						
<i>member_table</i>	A pointer to an array of structures where the requested information is placed.						
<i>member_count</i>	Specifies the number of nodes in the cluster.						
<b>EXTENDED DESCRIPTION</b>	<p>The process calling the <code>cmm_member_getall()</code> and <code>cmm_member_getcount()</code> functions allocates and frees all data structures used to return membership information, including the appropriate number of entries in the cluster node table.</p> <p>If there are more peer nodes than entries in <i>member table</i>, the table is not modified, <i>member_count</i> is updated, and a CMM_ERANGE error is returned. If there are more member entries than peer nodes, the excess member entries are zeroed out.</p> <p>If requested membership information is temporarily unavailable, as when a switchover is taking place, a CMM_ENOCLUSTER error is returned.</p> <p>The calling process is in charge of allocating the memory and indicating the number of entries by <i>table_size</i>.</p>						

	See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information on the <code>cmm_member_t</code> structure.																		
RETURN VALUES	<p>The <code>cmm_member_getall()</code> and <code>cmm_member_getcount()</code> functions return one of the following values:</p> <table><tr><td><code>CMM_EAGAIN</code></td><td>The information might be deprecated because the node has been out of communication with the master for a period of time.</td></tr><tr><td><code>CMM_EBUSY</code></td><td>The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr><tr><td><code>CMM_ECONN</code></td><td>No <code>nhcmmnd</code> is currently accessible on the local node.</td></tr><tr><td><code>CMM_EINVAL</code></td><td>Invalid argument such as <i>member_count</i> is a NULL pointer, or when <i>table_size</i> is greater than 1024.</td></tr><tr><td><code>CMM_ENOCLUSTER</code></td><td>The calling node is not yet in a cluster.</td></tr><tr><td><code>CMM_ENOTSUP</code></td><td>An unexpected service error occurred. The cluster might be in a critical state.</td></tr><tr><td><code>CMM_ERANGE</code></td><td>Not enough entries in the member table to provide the requested information.</td></tr><tr><td><code>CMM_ETIMEOUT</code></td><td>The call timeout expired before the action was completed.</td></tr><tr><td><code>CMM_OK</code></td><td>Operation succeeds.</td></tr></table>	<code>CMM_EAGAIN</code>	The information might be deprecated because the node has been out of communication with the master for a period of time.	<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.	<code>CMM_ECONN</code>	No <code>nhcmmnd</code> is currently accessible on the local node.	<code>CMM_EINVAL</code>	Invalid argument such as <i>member_count</i> is a NULL pointer, or when <i>table_size</i> is greater than 1024.	<code>CMM_ENOCLUSTER</code>	The calling node is not yet in a cluster.	<code>CMM_ENOTSUP</code>	An unexpected service error occurred. The cluster might be in a critical state.	<code>CMM_ERANGE</code>	Not enough entries in the member table to provide the requested information.	<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.	<code>CMM_OK</code>	Operation succeeds.
<code>CMM_EAGAIN</code>	The information might be deprecated because the node has been out of communication with the master for a period of time.																		
<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.																		
<code>CMM_ECONN</code>	No <code>nhcmmnd</code> is currently accessible on the local node.																		
<code>CMM_EINVAL</code>	Invalid argument such as <i>member_count</i> is a NULL pointer, or when <i>table_size</i> is greater than 1024.																		
<code>CMM_ENOCLUSTER</code>	The calling node is not yet in a cluster.																		
<code>CMM_ENOTSUP</code>	An unexpected service error occurred. The cluster might be in a critical state.																		
<code>CMM_ERANGE</code>	Not enough entries in the member table to provide the requested information.																		
<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.																		
<code>CMM_OK</code>	Operation succeeds.																		
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>																		

<b>NAME</b>	<code>cmm_member_getcount</code> - retrieve information on the cluster						
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_member_getall(uint32_t const  table_size, cmm_member_t * const member_table, uint32_t * const member_count);  cmm_error_t cmm_member_getcount(uint32_t * const member_count);</pre>						
<b>DESCRIPTION</b>	<p>The <code>cmm_member_getall()</code> function fills <i>member_table</i> with information about all nodes in the cluster. There is a table entry for each node. The information in this table is of the same type and meaning as that returned by <code>cmm_member_getinfo()</code>. If <i>member_table</i> is a null pointer, <code>cmm_member_getall()</code> behaves like the <code>cmm_member_getcount()</code> function.</p> <p>The <code>cmm_member_getcount()</code> function returns the number of nodes in the cluster, including the node from which the function is called. The value is stored in the area pointed to by <i>member_count</i>. See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for further information on the state of the node.</p>						
<b>PARAMETERS</b>	<p>The <code>cmm_member_getall()</code> function takes the following parameters:</p> <table> <tr> <td><i>table_size</i></td><td>Specifies the maximum number of entries in <i>member_table</i>. The maximum number of entries is 1024.</td></tr> <tr> <td><i>member_table</i></td><td>A pointer to an array of structures where the requested information is placed.</td></tr> <tr> <td><i>member_count</i></td><td>Specifies the number of nodes in the cluster.</td></tr> </table>	<i>table_size</i>	Specifies the maximum number of entries in <i>member_table</i> . The maximum number of entries is 1024.	<i>member_table</i>	A pointer to an array of structures where the requested information is placed.	<i>member_count</i>	Specifies the number of nodes in the cluster.
<i>table_size</i>	Specifies the maximum number of entries in <i>member_table</i> . The maximum number of entries is 1024.						
<i>member_table</i>	A pointer to an array of structures where the requested information is placed.						
<i>member_count</i>	Specifies the number of nodes in the cluster.						
<b>EXTENDED DESCRIPTION</b>	<p>The process calling the <code>cmm_member_getall()</code> and <code>cmm_member_getcount()</code> functions allocates and frees all data structures used to return membership information, including the appropriate number of entries in the cluster node table.</p> <p>If there are more peer nodes than entries in <i>member table</i>, the table is not modified, <i>member_count</i> is updated, and a <code>CMM_ERANGE</code> error is returned. If there are more member entries than peer nodes, the excess member entries are zeroed out.</p> <p>If requested membership information is temporarily unavailable, as when a switchover is taking place, a <code>CMM_ENOCLUSTER</code> error is returned.</p> <p>The calling process is in charge of allocating the memory and indicating the number of entries by <i>table_size</i>.</p> <p>See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information on the <code>cmm_member_t</code> structure.</p>						

RETURN VALUES	The <code>cmm_member_getall()</code> and <code>cmm_member_getcount()</code> functions return one of the following values:	
	CMM_EAGAIN	The information might be deprecated because the node has been out of communication with the master for a period of time.
	CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.
	CMM_ECONN	No <code>nhcmmnd</code> is currently accessible on the local node.
	CMM_EINVAL	Invalid argument such as <i>member_count</i> is a NULL pointer, or when <i>table_size</i> is greater than 1024.
	CMM_ENOCLUSTER	The calling node is not yet in a cluster.
	CMM_ENOTSUP	An unexpected service error occurred. The cluster might be in a critical state.
	CMM_ERANGE	Not enough entries in the member table to provide the requested information.
	CMM_ETIMEOUT	The call timeout expired before the action was completed.
	CMM_OK	Operation succeeds.

SEE ALSO      `Intro(3cmm)`, `cmm_member_getinfo(3cmm)`



<b>NAME</b>	cmm_member_getinfo - retrieve information about a peer node						
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_potential_getinfo (cmm_nodeid_t const <i>nodeid</i>, cmm_member_t * const <i>member</i>);  cmm_error_t cmm_member_getinfo (cmm_nodeid_t const <i>nodeid</i>, cmm_member_t * const <i>member</i>);</pre>						
<b>DESCRIPTION</b>	<p>The <code>cmm_potential_getinfo()</code> function retrieves the information contained in the <code>cmm_member_t</code> structure for a node identified by its <i>nodeid</i>. You can use <code>cmm_potential_getinfo()</code> to get into any peer node, even if it has the <code>CMM_OUT_OF_CLUSTER</code> state.</p> <p>See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information about the <code>cmm_member_t</code> structure.</p> <p>The <code>cmm_member_getinfo()</code> function retrieves the information in the <code>cmm_member_t</code> structure for a peer node.</p> <p>If the requested membership information is temporarily unavailable the operation is retried until it succeeds or a timeout occurs.</p> <p>In the case of a timeout, the <code>CMM_ETIMEDOUT</code> error is returned. If the <i>nodeid</i> specified is not in the cluster node table, a <code>CMM_ESRCH</code> error is returned.</p>						
<b>PARAMETERS</b>	<p>The <code>cmm_potential_getinfo()</code> and <code>cmm_member_getinfo()</code> functions take the following parameters:</p> <table> <tr> <td><i>member</i></td><td>Points to the <code>cmm_member_t</code> structure, which contains information about the node.</td></tr> <tr> <td><i>nodeid</i></td><td>Identifies the node on which information is requested.</td></tr> </table>	<i>member</i>	Points to the <code>cmm_member_t</code> structure, which contains information about the node.	<i>nodeid</i>	Identifies the node on which information is requested.		
<i>member</i>	Points to the <code>cmm_member_t</code> structure, which contains information about the node.						
<i>nodeid</i>	Identifies the node on which information is requested.						
<b>RETURN VALUES</b>	<p>The <code>cmm_potential_getinfo()</code> and <code>cmm_member_getinfo()</code> functions return one of the following values:</p> <table> <tr> <td><code>CMM_EAGAIN</code></td><td>The information might no longer be valid, as the node has been out of communication with the master node for a period of time.</td></tr> <tr> <td><code>CMM_EBUSY</code></td><td>The CMM API server is temporarily unable to respond to the requested operation. Wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr> <tr> <td><code>CMM_ECONN</code></td><td>The <code>nhcmmnd</code> daemon cannot be accessed on the current node.</td></tr> </table>	<code>CMM_EAGAIN</code>	The information might no longer be valid, as the node has been out of communication with the master node for a period of time.	<code>CMM_EBUSY</code>	The CMM API server is temporarily unable to respond to the requested operation. Wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.	<code>CMM_ECONN</code>	The <code>nhcmmnd</code> daemon cannot be accessed on the current node.
<code>CMM_EAGAIN</code>	The information might no longer be valid, as the node has been out of communication with the master node for a period of time.						
<code>CMM_EBUSY</code>	The CMM API server is temporarily unable to respond to the requested operation. Wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.						
<code>CMM_ECONN</code>	The <code>nhcmmnd</code> daemon cannot be accessed on the current node.						

SEE ALSO

CMM_EINVAL	Invalid argument such as the <i>member</i> is a NULL pointer or invalid <i>nodeid</i> .
CMM_ENOCLUSTER	The calling node is not part of any cluster.
CMM_ENOTSUP	Unexpected service error. The cluster might be in a critical state.
CMM_ESRCH	Returned by the <code>cmm_member_getinfo()</code> function if the node is in the local cluster node table but has the <code>CMM_OUT_OF_CLUSTER</code> role. Returned by both <code>cmm_potential_getinfo()</code> and <code>cmm_member_getinfo()</code> functions if the node is not in the local cluster nodes table.
CMM_ETIMEOUT	The call timeout expired before the action was completed.
CMM_OK	Operation succeeds.

Intro(3cmm)

<b>NAME</b>	cmm_member_isdesynchronized - interpret the status of a member
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>
<b>DESCRIPTION</b>	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>
<b>PARAMETERS</b>	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <p><i>member</i>                      A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b>(3cmm).</p>
<b>EXTENDED DESCRIPTION</b>	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <p><code>cmm_member_isdesynchronized()</code></p> <p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p> <p><code>cmm_member_isdisqualified()</code></p>

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_ismvicemaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	

NAME	cmm_member_isdisqualified - interpret the status of a member
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>
DESCRIPTION	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>
PARAMETERS	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <p><i>member</i>                      A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b>(3cmm).</p>
EXTENDED DESCRIPTION	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <p><code>cmm_member_isdesynchronized()</code></p> <p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p> <p><code>cmm_member_isdisqualified()</code></p>

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_isvicemaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	

<b>NAME</b>	cmm_member_iseligible - interpret the status of a member				
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>				
<b>DESCRIPTION</b>	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>				
<b>PARAMETERS</b>	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <table> <tr> <td><i>member</i></td><td>A pointer to a member structure that contains the member's information, such as a structure filled by <code>cmm_member_getinfo(3cmm)</code>.</td></tr> </table>	<i>member</i>	A pointer to a member structure that contains the member's information, such as a structure filled by <code>cmm_member_getinfo(3cmm)</code> .		
<i>member</i>	A pointer to a member structure that contains the member's information, such as a structure filled by <code>cmm_member_getinfo(3cmm)</code> .				
<b>EXTENDED DESCRIPTION</b>	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <table> <tr> <td><code>cmm_member_isdesynchronized()</code></td><td> <p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p> </td></tr> <tr> <td><code>cmm_member_isdisqualified()</code></td><td></td></tr> </table>	<code>cmm_member_isdesynchronized()</code>	<p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p>	<code>cmm_member_isdisqualified()</code>	
<code>cmm_member_isdesynchronized()</code>	<p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p>				
<code>cmm_member_isdisqualified()</code>					

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	



NAME	cmm_member_isexcluded - interpret the status of a member
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>
DESCRIPTION	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>
PARAMETERS	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <p><i>member</i>                      A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b>(3cmm).</p>
EXTENDED DESCRIPTION	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <p><code>cmm_member_isdesynchronized()</code></p> <p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p> <p><code>cmm_member_isdisqualified()</code></p>

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	

NAME	cmm_member_isfrozen - interpret the status of a member				
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>				
DESCRIPTION	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>				
PARAMETERS	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <table> <tr> <td><i>member</i></td><td>A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b>(3cmm).</td></tr> </table>	<i>member</i>	A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b> (3cmm).		
<i>member</i>	A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b> (3cmm).				
EXTENDED DESCRIPTION	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <table> <tr> <td><code>cmm_member_isdesynchronized()</code></td><td>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</td></tr> <tr> <td><code>cmm_member_isdisqualified()</code></td><td></td></tr> </table>	<code>cmm_member_isdesynchronized()</code>	The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.	<code>cmm_member_isdisqualified()</code>	
<code>cmm_member_isdesynchronized()</code>	The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.				
<code>cmm_member_isdisqualified()</code>					

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_isvicemaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	

NAME	cmm_member_ismaster - interpret the status of a member
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>
DESCRIPTION	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>
PARAMETERS	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <p><i>member</i>                      A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b>(3cmm).</p>
EXTENDED DESCRIPTION	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <p><code>cmm_member_isdesynchronized()</code></p> <p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p> <p><code>cmm_member_isdisqualified()</code></p>

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_isvicemaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	

<b>NAME</b>	cmm_member_isoutofcluster - interpret the status of a member
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>
<b>DESCRIPTION</b>	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>
<b>PARAMETERS</b>	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <p><i>member</i>                      A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b>(3cmm).</p>
<b>EXTENDED DESCRIPTION</b>	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <p><code>cmm_member_isdesynchronized()</code></p> <p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p> <p><code>cmm_member_isdisqualified()</code></p>

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_isvicemaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	



<b>NAME</b>	cmm_member_isqualified - interpret the status of a member				
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>				
<b>DESCRIPTION</b>	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>				
<b>PARAMETERS</b>	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <table> <tr> <td><i>member</i></td><td>A pointer to a member structure that contains the member's information, such as a structure filled by <code>cmm_member_getinfo(3cmm)</code>.</td></tr> </table>	<i>member</i>	A pointer to a member structure that contains the member's information, such as a structure filled by <code>cmm_member_getinfo(3cmm)</code> .		
<i>member</i>	A pointer to a member structure that contains the member's information, such as a structure filled by <code>cmm_member_getinfo(3cmm)</code> .				
<b>EXTENDED DESCRIPTION</b>	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <table> <tr> <td><code>cmm_member_isdesynchronized()</code></td><td>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</td></tr> <tr> <td><code>cmm_member_isdisqualified()</code></td><td></td></tr> </table>	<code>cmm_member_isdesynchronized()</code>	The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.	<code>cmm_member_isdisqualified()</code>	
<code>cmm_member_isdesynchronized()</code>	The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.				
<code>cmm_member_isdisqualified()</code>					

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	

<b>NAME</b>	cmm_member_isvicemaster - interpret the status of a member
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  int cmm_member_isdesynchronized(cmm_member_t  const * member); int cmm_member_isdisqualified(cmm_member_t  const * member); int cmm_member_iseligible(cmm_member_t  const * member); int cmm_member_isexcluded(cmm_member_t  const * member); int cmm_member_isfrozen(cmm_member_t  const * member); int cmm_member_ismaster(cmm_member_t  const * member); int cmm_member_isoutofcluster(cmm_member_t  const * member); int cmm_member_isqualified(cmm_member_t  const * member); int cmm_member_isvicemaster(cmm_member_t  const * member);</pre>
<b>DESCRIPTION</b>	<p>These functions enable an application to obtain the status of a peer node. The status information provided includes the membership attributes and role of the cluster as defined in the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i>.</p>
<b>PARAMETERS</b>	<p>The <code>cmm_member_is*()</code> function takes the following parameter:</p> <p><i>member</i>                      A pointer to a member structure that contains the member's information, such as a structure filled by <b>cmm_member_getinfo</b>(3cmm).</p>
<b>EXTENDED DESCRIPTION</b>	<p>The information provided by the <code>cmm_member_is*()</code> functions is as follows:</p> <pre>cmm_member_isdesynchronized()</pre> <p>The node is desynchronized if !=0 is returned. The desynchronization flag is set for this node. For the master, !=0 means that it owns the only up-to-date disk; the disks on all other nodes are stale. Remember to check the eligibility of this node to determine if it is a potential master.</p> <pre>cmm_member_isdisqualified()</pre>

		The node is disqualified if !=0 is returned. A node can be disqualified by sending a call to <code>cmm_member_setqualif()</code> to set the flag <code>CMM_MEMBER_DISQUALIFIED</code> . Check the eligibility of the node to verify that it can become master.
	<code>cmm_member_iseligible()</code>	
		The node is a master-eligible node if !=0 is returned.
	<code>cmm_member_isexcluded()</code>	
		The node is excluded if !=0 is returned.
	<code>cmm_member_isfrozen()</code>	
		The node is frozen if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the master node if !=0 is returned.
	<code>cmm_member_isoutofcluster()</code>	
		The node is not currently participating in cluster services.
	<code>cmm_member_isqualified()</code>	
		The node is qualified if !=0 is returned.
	<code>cmm_member_ismaster()</code>	
		The node is the vice-master if !=0 is returned.
	<b>Note</b> – !=0 is returned if, and only if, the node is neither disqualified nor desynchronized. A diskless node can also have this state. Check the eligibility of the node.	
RETURN VALUES	The <code>cmm_member_is*()</code> functions return one of the following values:	
	TRUE (!=0)	Condition is verified.
	FALSE (==0)	Condition is not satisfied.
SEE ALSO	<code>Intro(3cmm)</code> , <code>cmm_member_getinfo(3cmm)</code>	

<b>NAME</b>	cmm_member_seizequalif - requalify current master-eligible node												
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_member_seizequalif();</pre>												
<b>DESCRIPTION</b>	<p>The <code>cmm_member_seizequalif()</code> function qualifies the current node as the master node when there is no master node.</p> <p>The <code>cmm_member_seizequalif()</code> function must be called from a node that is master-eligible and has the attribute <code>CMM_ELIGIBLE_MEMBER</code>. If there is no master in the cluster, <code>cmm_member_setqualif()</code> cannot be called. If a node already exists with the attribute <code>CMM_QUALIFIED_MEMBER</code>, this call returns <code>CMM_EPERM</code>.</p>												
<b>EXTENDED DESCRIPTION</b>	<p>There are two outcomes of calling <code>cmm_member_seizequalif()</code> from a master-eligible node: either the node becomes master or it reverts to the qualification level it had prior to the <code>cmm_member_seizequalif()</code> function. This function returns a <code>CMM_EPERM</code> error if a master is already up and running or if the current node is not master-eligible. Note that if the node was previously <code>CMM_SYNCHRO_NEEDED</code> (flag S) it will not be elected as master if its former role was master.</p> <p>See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for further information on qualification levels.</p> <p>Necessary notifications are sent according to the impact of this call. See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information on possible notifications.</p>												
<b>RETURN VALUES</b>	<p>The <code>cmm_member_seizequalif()</code> function returns one of the following values:</p> <table> <tr> <td><code>CMM_EBUSY</code></td><td>The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr> <tr> <td><code>CMM_ECONN</code></td><td>No <code>nhcmmnd</code> is currently accessible to the local node.</td></tr> <tr> <td><code>CMM_ENOTSUP</code></td><td>An unexpected service error occurred.</td></tr> <tr> <td><code>CMM_EPERM</code></td><td>Permission denied. Either the function was not called from a master-eligible node, or a master is already running.</td></tr> <tr> <td><code>CMM_ETIMEOUT</code></td><td>The call timeout expired before the action was completed.</td></tr> <tr> <td><code>CMM_OK</code></td><td>Operation succeeds.</td></tr> </table>	<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.	<code>CMM_ECONN</code>	No <code>nhcmmnd</code> is currently accessible to the local node.	<code>CMM_ENOTSUP</code>	An unexpected service error occurred.	<code>CMM_EPERM</code>	Permission denied. Either the function was not called from a master-eligible node, or a master is already running.	<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.	<code>CMM_OK</code>	Operation succeeds.
<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.												
<code>CMM_ECONN</code>	No <code>nhcmmnd</code> is currently accessible to the local node.												
<code>CMM_ENOTSUP</code>	An unexpected service error occurred.												
<code>CMM_EPERM</code>	Permission denied. Either the function was not called from a master-eligible node, or a master is already running.												
<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.												
<code>CMM_OK</code>	Operation succeeds.												

(3cmm)

**SEE ALSO** | `Intro(3cmm), cmm_member_setqualif(3cmm)`

NAME	cmm_member_setqualif - give a new level of qualification to a node				
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_member_setqualif(cmm_nodeid_t const nodeid, cmm_qualif_t const new_qualif);</pre>				
DESCRIPTION	<p>The <code>cmm_member_setqualif()</code> function assigns a new qualification level to a node. A qualification level is only meaningful for a node with the administrative attribute <code>CMM_ELIGIBLE_MEMBER</code> in the <code>sflag</code> field of the <code>cmm_member_t</code> structure. See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for further information on the <code>cmm_member_t</code> structure.</p>				
PARAMETERS	<p>The <code>cmm_member_setqualif()</code> function takes the following parameters:</p> <table><tr><td><i>nodeid</i></td><td>Specifies the ID of the node whose qualification level is to be changed.</td></tr><tr><td><i>new_qualif</i></td><td>Specifies the new level of qualification assigned to the node.</td></tr></table>	<i>nodeid</i>	Specifies the ID of the node whose qualification level is to be changed.	<i>new_qualif</i>	Specifies the new level of qualification assigned to the node.
<i>nodeid</i>	Specifies the ID of the node whose qualification level is to be changed.				
<i>new_qualif</i>	Specifies the new level of qualification assigned to the node.				
EXTENDED DESCRIPTION	<p>The <code>cmm_member_setqualif()</code> function can only be called from the master node. If an attempt is made to call this function from a node other than the master, a <code>CMM_EPERM</code> error is returned.</p> <p>The <i>nodeid</i> given to the <code>cmm_member_setqualif()</code> function must be that of a node with the <code>CMM_ELIGIBLE_MEMBER</code> attribute. Changing the qualification level of a node in the cluster that does not have this attribute generates a <code>CMM_EINVAL</code> returned status. Changing the qualification level of a node with the <code>CMM_OUT_OF_CLUSTER</code> state will be successful as it is impossible to know whether or not the node is eligible. If the <i>nodeid</i> given to this function as a parameter is that of the master node, a failover occurs and a <code>MASTER_DEMOTED</code> notification is sent. <code>CMM_ENOCLUSTER</code> is returned until a new master is elected.</p> <p>The <i>new_qualif</i> parameter given to the <code>cmm_member_setqualif()</code> function is one of the following parameters:</p> <table><tr><td><code>CMM_QUALIFIED_MEMBER</code></td><td>The current node can take any role.</td></tr></table>	<code>CMM_QUALIFIED_MEMBER</code>	The current node can take any role.		
<code>CMM_QUALIFIED_MEMBER</code>	The current node can take any role.				

RETURN  
VALUES

CMM_DISQUALIFIED_MEMBER	The current node cannot participate in a master or a vice-master election.
CMM_SYNCHRO_READY	The Cluster Reliable File System (CRFS) daemons are synchronized. The vice-master node can become master if necessary. Only meaningful for eligible nodes. A user application should not set this flags. CRFS is in charge of setting them and changing them may disrupt the cluster.
CMM_SYNCHRO_NEEDED	The CRFS daemons are not synchronized. The vice-master node can not become master if necessary. Only meaningful for eligible nodes. A user application should not set this flags. CRFS is in charge of setting them and changing them may disrupt the cluster.

The `cmm_member_setqualif` call is asynchronous; the call's action is not fully completed when the call returns. If a second `cmm_member_setqualif` call is made to disqualify the master while the first call is still executing but before the master is demoted, both calls return `CMM_OK`, even though the master is in the process of being demoted. The result is not affected by the second call.

After a `cmm_member_setqualif` call, for a short period of time the cluster has no master, until the vice-master becomes the new master. Any call to the Cluster Membership Manger (CMM) Application Programming Interface (API) function in this short period of time will return `CMM_ENOCLUSTER`.

Necessary notifications are sent according to the impact of this call, as described in the *Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide*.

The value of `sflag` on the indicated node is immediately updated when a change in the cluster state occurs, so a call to `cmm_member_getinfo(3CMM)` will reflect the change.

The `cmm_member_setqualif()` function returns one of the following errors:

CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.
CMM_ECONN	No <code>nhcmmmd</code> is currently accessible in the local node.



CMM_EINVAL	Invalid parameter. Either the <i>nodeid</i> is not that of a master-eligible node or <i>new_qualif</i> is incorrect.
CMM_ENOCLUSTER	The calling node is not yet in a cluster.
CMM_ENOTSUP	An unexpected service error occurred.
CMM_EPERM	Permission denied. The function was not called from the master node.
CMM_ETIMEOUT	The call timeout expired before the action was completed.
CMM_OK	Operation succeeds.

**SEE ALSO**

`Intro(3cmm)`, `cmm_member_getinfo(3cmm)`,  
`cmm_mastership_release(3cmm)`, `cmm_membership_remove(3cmm)`

(3cmm)



NAME	cmm_membership_include - rejoin the cluster after having left it												
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_membership_include();</pre>												
DESCRIPTION	<p>The <code>cmm_membership_include()</code> function causes the calling member to rejoin the cluster after it left the cluster as the result of a call to <code>cmm_membership_remove()</code> or because it rebooted with the option <code>CMM.StartUp.Join</code> set to <code>False</code>.</p> <p>Necessary notifications (<code>CMM_MEMBER_JOINED</code> or <code>VICEMASTER_ELECTED</code>) are sent according to the impact of this call. (See section 5.2 for a case-by-case description.)</p> <p>Note that to actually join the cluster, a node must be configured to be in the cluster.</p>												
RETURN VALUES	<p>The <code>cmm_membership_include()</code> function returns one of the following values:</p> <table><tr><td><code>CMM_OK</code></td><td>Operation succeeds or the node is already IN.</td></tr><tr><td><code>CMM_ENOCLUSTER</code></td><td>Calling node is not configured for any running cluster.</td></tr><tr><td><code>CMM_ETIMEDOUT</code></td><td>No response within the delay. The operation is not canceled, so it could be performed after returning from <code>cmm_membership_include()</code> with this error code.</td></tr><tr><td><code>CMM_ECONN</code></td><td>No CMM is currently and locally accessible.</td></tr><tr><td><code>CMM_ENOTSUP</code></td><td>Unexpected service error. Cluster might be in a critical state.</td></tr><tr><td><code>CMM_EBUSY</code></td><td>CMM temporarily out of resources. Retry later.</td></tr></table>	<code>CMM_OK</code>	Operation succeeds or the node is already IN.	<code>CMM_ENOCLUSTER</code>	Calling node is not configured for any running cluster.	<code>CMM_ETIMEDOUT</code>	No response within the delay. The operation is not canceled, so it could be performed after returning from <code>cmm_membership_include()</code> with this error code.	<code>CMM_ECONN</code>	No CMM is currently and locally accessible.	<code>CMM_ENOTSUP</code>	Unexpected service error. Cluster might be in a critical state.	<code>CMM_EBUSY</code>	CMM temporarily out of resources. Retry later.
<code>CMM_OK</code>	Operation succeeds or the node is already IN.												
<code>CMM_ENOCLUSTER</code>	Calling node is not configured for any running cluster.												
<code>CMM_ETIMEDOUT</code>	No response within the delay. The operation is not canceled, so it could be performed after returning from <code>cmm_membership_include()</code> with this error code.												
<code>CMM_ECONN</code>	No CMM is currently and locally accessible.												
<code>CMM_ENOTSUP</code>	Unexpected service error. Cluster might be in a critical state.												
<code>CMM_EBUSY</code>	CMM temporarily out of resources. Retry later.												
SEE ALSO	<code>Intro(3cmm)</code> , <a href="#">nhcmmnd(8)</a>												

(3cmm)



NAME	cmm_membership_remove - remove peer node
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_membership_remove( );</pre>
DESCRIPTION	<p>The <code>cmm_membership_remove()</code> function removes the node from which this function is called from the cluster. The node is still configured to be in the cluster, but its role has changed. The CMM API is always accessible for an <code>CMM_OUT_OF_CLUSTER</code> node.</p> <p>When the <code>nhcmm</code> daemon detects that a node is no longer part of the cluster, it informs other applications or services that are registered to receive notification. See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information on the notifications returned in different scenarios.</p> <p>To reintegrate a node into the cluster after the <code>cmm_membership_remove()</code> function has been called on the node, the function <code>cmm_membership_include()</code> can be used.</p> <p>Note that after the <code>cmm_membership_remove()</code> call, a node is still to be configured to be in the cluster, but it has the <code>CMM_OUT_OF_CLUSTER</code> role. Because it is still configured to be in the cluster, it can access cluster information:</p> <ul style="list-style-type: none"> <li>■ Functions that retrieve information on the cluster state can still be called by the node</li> <li>■ Functions that modify the cluster state can no longer be called by the node</li> </ul> <p>This is different from a node not being configured for <i>any</i> cluster. If the node is not configured for <i>any</i> cluster, the <code>CMM_ENOCLUSTER</code> value is returned.</p> <p>Any program running on the node can call <code>cmm_membership_remove()</code>. There is no authentication carried out of the program making the call. Calling this function from the master leads to a failover (or <code>CMM_ECANCELED</code> if no vice-master can take the mastership role).</p>

RETURN VALUES	The <code>cmm_membership_remove()</code> function returns one of the following values:	
	<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.
	<code>CMM_ECANCELED</code>	Operation canceled. The function was called on the master node but the vice-master node was not qualified to become master.
	<code>CMM_ECONN</code>	No <code>nhcmmnd</code> is accessible to the current node.
	<code>CMM_ENOCLUSTER</code>	The calling node is not yet in a cluster.
	<code>CMM_ENOTSUP</code>	An unexpected service error occurred. The cluster might be in a critical state.
	<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.
	<code>CMM_OK</code>	Operation succeeds.

**SEE ALSO** `Intro(3cmm)`, `nhcmmnd(8)`, `cmm_membership_include(3cmm)`

NAME	cmm_node_getid - retrieve ID of a node												
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_node_getid(cmm_nodeid_t * const me);</pre>												
DESCRIPTION	<p>The <code>cmm_node_getid()</code> function retrieves a <i>nodeid</i>. This value is accessible even if no master is currently elected. Using the <i>nodeid</i>, the node can retrieve information on its status in the cluster.</p> <p>The <i>me</i> parameter is a pointer to a <code>cmm_nodeid_t</code> structure. If an error occurs, <i>me</i> contains <code>CMM_INVALID_NODE_ID</code> and the returned value shows the cause of the error.</p> <p><b>Note</b> – This function is not related to cluster information and so it can be called from any peer node, even a node with the <code>CMM_OUT_OF_CLUSTER</code> state.</p>												
RETURN VALUES	<p>The <code>cmm_node_getid()</code> function returns one of the following values:</p> <table><tr><td><code>CMM_EBUSY</code></td><td>The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr><tr><td><code>CMM_ECONN</code></td><td>No <code>nhcmmnd(8)</code> is accessible on the current node.</td></tr><tr><td><code>CMM_EINVAL</code></td><td>Invalid parameter. <i>me</i> is a <code>NULL</code> pointer.</td></tr><tr><td><code>CMM_ENOTSUP</code></td><td>An unexpected service error occurred. The cluster might be in a critical state.</td></tr><tr><td><code>CMM_ETIMEOUT</code></td><td>The call timeout expired before the action was completed.</td></tr><tr><td><code>CMM_OK</code></td><td>Operation succeeds.</td></tr></table> <p>This call never returns <code>CMM_ENOCLUSTER</code>.</p>	<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.	<code>CMM_ECONN</code>	No <code>nhcmmnd(8)</code> is accessible on the current node.	<code>CMM_EINVAL</code>	Invalid parameter. <i>me</i> is a <code>NULL</code> pointer.	<code>CMM_ENOTSUP</code>	An unexpected service error occurred. The cluster might be in a critical state.	<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.	<code>CMM_OK</code>	Operation succeeds.
<code>CMM_EBUSY</code>	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.												
<code>CMM_ECONN</code>	No <code>nhcmmnd(8)</code> is accessible on the current node.												
<code>CMM_EINVAL</code>	Invalid parameter. <i>me</i> is a <code>NULL</code> pointer.												
<code>CMM_ENOTSUP</code>	An unexpected service error occurred. The cluster might be in a critical state.												
<code>CMM_ETIMEOUT</code>	The call timeout expired before the action was completed.												
<code>CMM_OK</code>	Operation succeeds.												
SEE ALSO	<code>Intro(3cmm)</code> , <code>nhcmmnd(8)</code>												

(3cmm)





<b>NAME</b>	<code>cmm_notify_dispatch</code> - dispatch cluster membership change messages								
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t  cmm_notify_dispatch( );</pre>								
<b>DESCRIPTION</b>	<p>The <code>cmm_notify_dispatch()</code> function processes a cluster membership change control message and invokes the appropriate callback function.</p> <p>The process uses the <code>select</code> or <code>poll</code> commands to detect messages arriving from the <code>nhcmmnd</code> daemon, with at least the file descriptor returned from the <code>cmm_notify_getfd()</code> function. When the file descriptor indicates that data must be read, the <code>cmm_notify__dispatch()</code> function is called, and the registered callback function is invoked from the same thread that calls the <code>cmm_notify_dispatch()</code> function.</p> <p>If an error occurs on this file descriptor within <code>poll()</code> or if the file descriptor is no longer valid, <code>CMM_EBADF</code> is returned by <code>cmm_notify_dispatch()</code>. Then <code>cmm_cmc_unregister(3cmm)</code> must be called and the whole registration must be performed - including calling the <code>cmm_cmc_register(3cmm)</code>, <code>cmm_cmc_filter(3cmm)</code> and <code>cmm_notify_getfd()</code> functions.</p> <p>Note that one call to <code>cmm_notify_dispatch()</code> can lead to as many calls to the callback as there are pending notifications. If within this callback, some functions are invoked concerning the state of the cluster (for instance, to get the number of nodes), the result of the function refers to the state of the cluster when the function was invoked. It does not refer to the state of the cluster when the notification was generated. In the meantime, some other modifications could have been applied to the cluster.</p>								
<b>RETURN VALUES</b>	<p>The <code>cmm_notify_dispatch()</code> function returns one of the following values:</p> <table> <tr> <td><code>CMM_EBADF</code></td><td>Bad file descriptor.</td></tr> <tr> <td><code>CMM_ENOENT</code></td><td>No callback is currently registered.</td></tr> <tr> <td><code>CMM_ENOTSUP</code></td><td>Unexpected service error. Cluster might be in a critical state.</td></tr> <tr> <td><code>CMM_OK</code></td><td>Operation succeeds.</td></tr> </table>	<code>CMM_EBADF</code>	Bad file descriptor.	<code>CMM_ENOENT</code>	No callback is currently registered.	<code>CMM_ENOTSUP</code>	Unexpected service error. Cluster might be in a critical state.	<code>CMM_OK</code>	Operation succeeds.
<code>CMM_EBADF</code>	Bad file descriptor.								
<code>CMM_ENOENT</code>	No callback is currently registered.								
<code>CMM_ENOTSUP</code>	Unexpected service error. Cluster might be in a critical state.								
<code>CMM_OK</code>	Operation succeeds.								
<b>SEE ALSO</b>	<p><code>Intro(3cmm)</code>, <code>select(3C)</code>, <code>poll(2)</code>, <code>cmm_cmc_filter(3cmm)</code>, <code>cmm_cmc_register(3cmm)</code>, <code>cmm_cmc_unregister(3cmm)</code>, <code>cmm_notify_getfd(3cmm)</code></p>								

(3cmm)



NAME	cmm_notify_getfd - receive cluster membership change messages						
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_notify_getfd(int * const fd);</pre>						
DESCRIPTION	<p>The <code>cmm_notify_getfd()</code> function stores a file descriptor, in <i>fd</i>. This descriptor detects the cluster membership change control messages sent by the <code>nhcmmnd</code> daemon. System services or applications that require cluster membership change notification delivery must use the <code>cmm_notify_getfd()</code> and the <code>cmm_notify_dispatch()</code> functions to receive and process messages from the <code>nhcmmnd</code> daemon.</p> <p>The <i>fd</i> parameter points to the location where the function stores the file descriptor used to receive messages from <code>nhcmmnd</code>.</p>						
EXTENDED DESCRIPTION	<p>Use the <code>select()</code> or <code>poll()</code> functions and the file descriptor returned from <code>cmm_notify_getfd()</code> to detect messages arriving from <code>nhcmmnd</code>. When the file descriptor indicates that data must be read, <code>cmm_notify_dispatch()</code> must be called. This triggers the associated callback registered through <code>cmm_cmc_register()</code>.</p> <p>If an error occurs on this file descriptor within <code>poll()</code> or if the file descriptor is no longer valid, <code>cmm_notify_dispatch()</code> returns a <code>CMM_EBADF</code> error. The thread that received this error must call the <code>cmm_cmc_unregister()</code> and the whole registration process must be performed, that is calling the <code>cmm_cmc_register()</code>, <code>cmm_cmc_filter()</code>, and <code>cmm_notify_getfd()</code> functions.</p> <p>If the <code>fork</code> command is called, the returned file descriptor is automatically closed in the created child process. The file descriptor is only valid within the parent.</p> <p>If an application calls the <code>cmm_notify_getfd()</code> function, it must use the returned file descriptor. The Cluster Membership Manager (CMM) library itself does not monitor events, so if the calling process does not call the <code>cmm_notify_dispatch()</code> function when an event occurs, the events accumulate without being handled.</p>						
RETURN VALUES	<p>The <code>cmm_notify_getfd()</code> function returns one of the following values:</p> <table><tr><td><code>CMM_EINVAL</code></td><td>Invalid argument.</td></tr><tr><td><code>CMM_ENOENT</code></td><td>No callback is currently registered.</td></tr><tr><td><code>CMM_OK</code></td><td>Operation succeeds</td></tr></table>	<code>CMM_EINVAL</code>	Invalid argument.	<code>CMM_ENOENT</code>	No callback is currently registered.	<code>CMM_OK</code>	Operation succeeds
<code>CMM_EINVAL</code>	Invalid argument.						
<code>CMM_ENOENT</code>	No callback is currently registered.						
<code>CMM_OK</code>	Operation succeeds						

(3cmm)

**SEE ALSO**

Intro(3cmm), select(3C), nhcmmnd(8), fork(2), poll(2),  
cmm\_cmc\_filter(3cmm), cmm\_cmc\_register(3cmm),  
cmm\_cmc\_unregister(3cmm), cmm\_notify\_dispatch(3cmm), select(3C),  
poll(2)

NAME	cmm_potential_getinfo - retrieve information about a peer node				
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_potential_getinfo (cmm_nodeid_t const nodeid, cmm_member_t * const member);  cmm_error_t cmm_member_getinfo (cmm_nodeid_t const nodeid, cmm_member_t * const member);</pre>				
DESCRIPTION	<p>The <code>cmm_potential_getinfo()</code> function retrieves the information contained in the <code>cmm_member_t</code> structure for a node identified by its <i>nodeid</i>. You can use <code>cmm_potential_getinfo()</code> to get into any peer node, even if it has the <code>CMM_OUT_OF_CLUSTER</code> state.</p> <p>See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information about the <code>cmm_member_t</code> structure.</p> <p>The <code>cmm_member_getinfo()</code> function retrieves the information in the <code>cmm_member_t</code> structure for a peer node.</p> <p>If the requested membership information is temporarily unavailable the operation is retried until it succeeds or a timeout occurs.</p> <p>In the case of a timeout, the <code>CMM_ETIMEDOUT</code> error is returned. If the <i>nodeid</i> specified is not in the cluster node table, a <code>CMM_ESRCH</code> error is returned.</p>				
PARAMETERS	<p>The <code>cmm_potential_getinfo()</code> and <code>cmm_member_getinfo()</code> functions take the following parameters:</p> <table><tr><td><i>member</i></td><td>Points to the <code>cmm_member_t</code> structure, which contains information about the node.</td></tr><tr><td><i>nodeid</i></td><td>Identifies the node on which information is requested.</td></tr></table>	<i>member</i>	Points to the <code>cmm_member_t</code> structure, which contains information about the node.	<i>nodeid</i>	Identifies the node on which information is requested.
<i>member</i>	Points to the <code>cmm_member_t</code> structure, which contains information about the node.				
<i>nodeid</i>	Identifies the node on which information is requested.				
RETURN VALUES	<p>The <code>cmm_potential_getinfo()</code> and <code>cmm_member_getinfo()</code> functions return one of the following values:</p> <table><tr><td><code>CMM_EAGAIN</code></td><td>The information might no longer be valid, as the node has been out of communication with the master node for a period of time.</td></tr><tr><td><code>CMM_EBUSY</code></td><td>The CMM API server is temporarily unable to respond to the requested operation. Wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr></table>	<code>CMM_EAGAIN</code>	The information might no longer be valid, as the node has been out of communication with the master node for a period of time.	<code>CMM_EBUSY</code>	The CMM API server is temporarily unable to respond to the requested operation. Wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.
<code>CMM_EAGAIN</code>	The information might no longer be valid, as the node has been out of communication with the master node for a period of time.				
<code>CMM_EBUSY</code>	The CMM API server is temporarily unable to respond to the requested operation. Wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.				

SEE ALSO

CMM_ECONN	The nhcmmnd daemon cannot be accessed on the current node.
CMM_EINVAL	Invalid argument such as the <i>member</i> is a NULL pointer or invalid <i>nodeid</i> .
CMM_ENOCLUSTER	The calling node is not part of any cluster.
CMM_ENOTSUP	Unexpected service error. The cluster might be in a critical state.
CMM_ESRCH	Returned by the <code>cmm_member_getinfo()</code> function if the node is in the local cluster node table but has the <code>CMM_OUT_OF_CLUSTER</code> role. Returned by both <code>cmm_potential_getinfo()</code> and <code>cmm_member_getinfo()</code> functions if the node is not in the local cluster nodes table.
CMM_ETIMEOUT	The call timeout expired before the action was completed.
CMM_OK	Operation succeeds.
Intro(3cmm)	

NAME	cmm_strerror - get error message string						
SYNOPSIS	<pre>cc [ flag... ] file... -lcgha_cmm -lrt #include &lt;cmm/cmm.h&gt; char *cmm_strerror(cmm_error_t errnum);</pre>						
DESCRIPTION	<p>The <code>cmm_strerror()</code> function maps the error number in <code>errnum</code> to an error message string, and returns a pointer to that string. The returned string must not be overwritten or freed by the calling process.</p> <p><code>errnum</code> is a string representing an error code.</p>						
RETURN VALUES	<p>The <code>cmm_strerror()</code> function returns one of the following values:</p> <table><tr><td>"No Error"</td><td><code>errnum</code> is equal to <code>CMM_OK</code>.</td></tr><tr><td>"Unknown Error"</td><td><code>errnum</code> does not correspond to any error code.</td></tr><tr><td>"A string equal to the error code"</td><td><code>errnum</code> is equal to the error code.</td></tr></table>	"No Error"	<code>errnum</code> is equal to <code>CMM_OK</code> .	"Unknown Error"	<code>errnum</code> does not correspond to any error code.	"A string equal to the error code"	<code>errnum</code> is equal to the error code.
"No Error"	<code>errnum</code> is equal to <code>CMM_OK</code> .						
"Unknown Error"	<code>errnum</code> does not correspond to any error code.						
"A string equal to the error code"	<code>errnum</code> is equal to the error code.						
SEE ALSO	<code>Intro(3cmm)</code>						

(3cmm)





<b>NAME</b>	cmm_vicemaster_getinfo - retrieve information about the master node or the vice-master node																		
<b>SYNOPSIS</b>	<pre>cc [ flag... ] file... -lcgha_cmm -lrt  #include &lt;cmm/cmm.h&gt;  cmm_error_t cmm_master_getinfo(cmm_member_t * const member);  cmm_error_t cmm_vicemaster_getinfo(cmm_member_t * const member);</pre>																		
<b>DESCRIPTION</b>	<p>The <code>cmm_master_getinfo()</code> function returns information about the current master node. The <code>cmm_vicemaster_getinfo()</code> function returns information about the current vice-master node. The information returned by these two functions has the same type and meaning as that returned by the <code>cmm_member_getinfo()</code> function.</p> <p>The <i>member</i> parameter is a pointer to a member structure where the function stores the member's information, such as its current state.</p> <p>See the <i>Netra High Availability Suite 3.0 1/08 Foundation Services CMM Programming Guide</i> for information on the <code>cmm_member_t</code> structure.</p>																		
<b>RETURN VALUES</b>	<p>The <code>cmm_master_getinfo()</code> and <code>cmm_vicemaster_getinfo()</code> functions return one of the following values:</p> <table> <tr> <td>CMM_EAGAIN</td><td>The information might be deprecated, because a node has been out of communication with the master node for a period of time.</td></tr> <tr> <td>CMM_EBUSY</td><td>The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.</td></tr> <tr> <td>CMM_ECONN</td><td>No nhcmmnd is currently accessible on the local node.</td></tr> <tr> <td>CMM_EINVAL</td><td>Invalid parameter. <i>member</i> is a NULL pointer.</td></tr> <tr> <td>CMM_ENOCLUSTER</td><td>The calling node is not yet in a cluster.</td></tr> <tr> <td>CMM_ENOTSUP</td><td>An unexpected service error occurred. The cluster might be in a critical state.</td></tr> <tr> <td>CMM_ESRCH</td><td>No such member. This return value is only applicable for the vice-master node.</td></tr> <tr> <td>CMM_ETIMEOUT</td><td>The call timeout expired before the action was completed.</td></tr> <tr> <td>CMM_OK</td><td>Operation succeeds.</td></tr> </table>	CMM_EAGAIN	The information might be deprecated, because a node has been out of communication with the master node for a period of time.	CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.	CMM_ECONN	No nhcmmnd is currently accessible on the local node.	CMM_EINVAL	Invalid parameter. <i>member</i> is a NULL pointer.	CMM_ENOCLUSTER	The calling node is not yet in a cluster.	CMM_ENOTSUP	An unexpected service error occurred. The cluster might be in a critical state.	CMM_ESRCH	No such member. This return value is only applicable for the vice-master node.	CMM_ETIMEOUT	The call timeout expired before the action was completed.	CMM_OK	Operation succeeds.
CMM_EAGAIN	The information might be deprecated, because a node has been out of communication with the master node for a period of time.																		
CMM_EBUSY	The CMM API server is temporarily out of resources to respond to the requested operation. The recommended action is to wait a short time and retry the operation. The length of the waiting must be decided by the user, depending on the application's characteristics.																		
CMM_ECONN	No nhcmmnd is currently accessible on the local node.																		
CMM_EINVAL	Invalid parameter. <i>member</i> is a NULL pointer.																		
CMM_ENOCLUSTER	The calling node is not yet in a cluster.																		
CMM_ENOTSUP	An unexpected service error occurred. The cluster might be in a critical state.																		
CMM_ESRCH	No such member. This return value is only applicable for the vice-master node.																		
CMM_ETIMEOUT	The call timeout expired before the action was completed.																		
CMM_OK	Operation succeeds.																		

(3cmm)

**SEE ALSO** | `Intro(3cmm), cmm_member_getinfo(3cmm)`

NAME	addon.conf - nhinstall configuration file to install additional patches and packages
SYNOPSIS	<b>addon.conf</b>
DESCRIPTION	<p>You can configure the <code>nhinstall</code> tool to install additional patches and packages by modifying the <code>addon.conf</code> file.</p> <p>It is not mandatory to configure this file. If this file is not configured or not present in the directory containing the configuration files, the <code>nhinstall</code> tool assumes there are no additional patches or packages to be installed. This file can be used to upgrade the Foundation Services at a later stage.</p> <p>The templates for the configuration files are contained in the <code>/opt/sun/nhas/templates/nhinstall</code> directory with <code>.template</code> extensions. Templates for the <code>addon.conf</code> file are specific to the hardware platform type. Copy the necessary <code>addon.conf</code> template files to a local directory on the installation server as follows:</p> <pre># mkdir config-file-directory # export NHOME=/opt/sun/nhas/templates/nhinstall # cp \$NHOME/addon.conf.*.template config-file-directory</pre> <p><b>Note</b> – All the configuration files must be in the same local directory on the installation server.</p> <p>The <code>addon.conf</code> file format is ASCII. Comment lines begin with the comment mark (<code>#</code>). Parameters consist of a keyword followed by an equals (<code>=</code>) sign followed by the parameter value, of the form:</p> <p><i>Keyword=Value</i></p> <p>Within the <i>Value</i>, you can use a slash at the end of a line to indicate that the <i>Value</i> is continued on the following line. You can also add comments within a <i>Value</i>. For example:</p> <pre>PACKAGE=MyPackage.x86_64.rpm \ # This is a package to add /MyDir - F LOCAL Y Y Y</pre> <p>Each additional patch or package to be installed must be specified in <code>addon.conf</code> by using the following parameters:</p> <pre>PACKAGE=reference dir sub_dir phase scope men diskless dataless</pre>

<i>reference</i>	The package name.						
<i>dir</i>	<p>The directory exported from the installation server and mounted on remote nodes.</p> <p>To install additional Foundation Services packages after the Foundation Services have been installed, specify the <i>dir</i> as NHAS and the <i>phase</i> as F. The additional packages must be in the directory specified for NHAS_PRODUCT_DIR in <code>env_installation.conf</code>.</p>						
<i>sub_dir</i>	<p>The <i>sub_dir</i> directory is a subdirectory of <i>dir</i> containing the packages. If there is no subdirectory and the package is located in the exported directory, specify "-" for the <i>sub_dir</i> parameter.</p> <p>If you define NHAS as the value for the <i>dir</i> parameter, the "-" takes one of the following values:</p> <ul style="list-style-type: none"> <li>■ NetraHASx/Packages for a package</li> </ul> <p>Where <i>x</i> depends on the Foundation Services version installed.</p>						
<i>phase</i>	<p>Indicates the phase when the patch or package must be installed. The phases are:</p> <table> <tr> <td>I</td><td>The package is installed during the installation of the operating system on the master-eligible nodes and dataless nodes.</td></tr> <tr> <td>S</td><td>Similar to I on Linux installation servers.</td></tr> <tr> <td>F</td><td>The package is installed after the Foundation Services are installed for MENs and dataless nodes.</td></tr> </table>	I	The package is installed during the installation of the operating system on the master-eligible nodes and dataless nodes.	S	Similar to I on Linux installation servers.	F	The package is installed after the Foundation Services are installed for MENs and dataless nodes.
I	The package is installed during the installation of the operating system on the master-eligible nodes and dataless nodes.						
S	Similar to I on Linux installation servers.						
F	The package is installed after the Foundation Services are installed for MENs and dataless nodes.						
<i>scope</i>	<p>Indicates where the package will be installed.</p> <table> <tr> <td>LOCAL</td><td>Install the package on the root partition of the nodes specified.</td></tr> <tr> <td>SHARED</td><td>Install the package in the shared package directory, <code>/SUNWcgha/local/export/services</code>. If you specify SHARED, the package or patch cannot be installed after the Solaris installation on MENs because the shared directory does not exist yet.</td></tr> </table> <p><b>Note</b> – The <code>USR_SPECIFIC</code>, <code>USR_SOLARIS</code>, and <code>CLONE_OPT</code> parameters are replaced by <code>LOCAL</code> if you are installing the software for a master-eligible node or a dataless node.</p>	LOCAL	Install the package on the root partition of the nodes specified.	SHARED	Install the package in the shared package directory, <code>/SUNWcgha/local/export/services</code> . If you specify SHARED, the package or patch cannot be installed after the Solaris installation on MENs because the shared directory does not exist yet.		
LOCAL	Install the package on the root partition of the nodes specified.						
SHARED	Install the package in the shared package directory, <code>/SUNWcgha/local/export/services</code> . If you specify SHARED, the package or patch cannot be installed after the Solaris installation on MENs because the shared directory does not exist yet.						

<i>reference</i>	The package name.						
<i>dir</i>	<p>The directory exported from the installation server and mounted on remote nodes.</p> <p>To install additional Foundation Services packages after the Foundation Services have been installed, specify the <i>dir</i> as NHAS and the <i>phase</i> as F. The additional packages must be in the directory specified for NHAS_PRODUCT_DIR in <code>env_installation.conf</code>.</p>						
<i>sub_dir</i>	<p>The <i>sub_dir</i> directory is a subdirectory of <i>dir</i> containing the packages. If there is no subdirectory and the package is located in the exported directory, specify "-" for the <i>sub_dir</i> parameter.</p> <p>If you define NHAS as the value for the <i>dir</i> parameter, the "-" takes one of the following values:</p> <ul style="list-style-type: none"><li>■ NetraHAS<i>x</i>/Packages for a package</li></ul> <p>Where <i>x</i> depends on the Foundation Services version installed.</p>						
<i>phase</i>	<p>Indicates the phase when the patch or package must be installed. The phases are:</p> <table><tr><td>I</td><td>The package is installed during the installation of the operating system on the master-eligible nodes and dataless nodes.</td></tr><tr><td>S</td><td>Similar to I on Linux installation servers.</td></tr><tr><td>F</td><td>The package is installed after the Foundation Services are installed for MENs and dataless nodes.</td></tr></table>	I	The package is installed during the installation of the operating system on the master-eligible nodes and dataless nodes.	S	Similar to I on Linux installation servers.	F	The package is installed after the Foundation Services are installed for MENs and dataless nodes.
I	The package is installed during the installation of the operating system on the master-eligible nodes and dataless nodes.						
S	Similar to I on Linux installation servers.						
F	The package is installed after the Foundation Services are installed for MENs and dataless nodes.						
<i>scope</i>	<p>Indicates where the package will be installed.</p> <table><tr><td>LOCAL</td><td>Install the package on the root partition of the nodes specified.</td></tr><tr><td>SHARED</td><td>Install the package in the shared package directory, <code>/SUNWcgha/local/export/services</code>. If you specify SHARED, the package or patch cannot be installed after the Solaris installation on MENs because the shared directory does not exist yet.</td></tr></table> <p><b>Note</b> – The <code>USR_SPECIFIC</code>, <code>USR_SOLARIS</code>, and <code>CLONE_OPT</code> parameters are replaced by <code>LOCAL</code> if you are installing the software for a master-eligible node or a dataless node.</p>	LOCAL	Install the package on the root partition of the nodes specified.	SHARED	Install the package in the shared package directory, <code>/SUNWcgha/local/export/services</code> . If you specify SHARED, the package or patch cannot be installed after the Solaris installation on MENs because the shared directory does not exist yet.		
LOCAL	Install the package on the root partition of the nodes specified.						
SHARED	Install the package in the shared package directory, <code>/SUNWcgha/local/export/services</code> . If you specify SHARED, the package or patch cannot be installed after the Solaris installation on MENs because the shared directory does not exist yet.						

<i>men</i>	Indicates if a package is to be installed on the MENs.  Options are Y or N.  This parameter is ignored if <i>scope</i> is set to SHARED.
<i>diskless</i>	Indicates if a package is to be installed for a diskless node.  Options are Y or N.  This parameter is ignored on Linux because diskless nodes are not supported for Linux in this release.
<i>dataless</i>	Indicates if a package is to be installed for a dataless node.  Options are Y or N.  This parameter is ignored if <i>scope</i> is set to SHARED.
<i>method</i>	Indicates the method used for patch installation for diskless nodes. The <i>method</i> parameter is optional on the Solaris OS and is ignored on Linux because diskless nodes are not supported for Linux in this release.

**EXAMPLES** This section provides examples of how to use the `addon.conf` file.

**EXAMPLE 1** Sample `addon.conf` File

```
# A package located on a user's directory installed on the root file system
# only on master-eligible nodes after NHAS Foundation Services installation
PACKAGE=MyPkgName /export/myDistrib - F LOCAL Y N N

# A package located on a user's directory and installed
# on master-eligible and dataless nodes after NHAS FS installation.
PACKAGE=MyPkgName /export/myDistrib - F LOCAL Y N Y
PACKAGE=MyPkgName /export/myDistrib - F LOCAL Y N Y
```

**SEE ALSO** `env_installation.conf(5)`, `cluster_definition.conf(5)`, `nhinstall(8)`

NAME	cluster_definition.conf - nhinstall configuration file to define the cluster
SYNOPSIS	<b>cluster_definition.conf</b>
DESCRIPTION	<p>Configure the <code>cluster_definition.conf</code> file to define the cluster nodes for the <code>nhinstall</code> tool.</p> <p>The templates for the configuration files are contained in the <code>/opt/sun/nhas/templates/nhinstall</code> directory with <code>.template</code> extensions. Copy the configuration files to a local directory on the installation server as follows:</p> <pre># <b>mkdir</b> config-file-directory # <b>export</b> NHOME=/opt/sun/nhas/templates/nhinstall # <b>cd</b> config-file-directory # <b>cp</b> \$NHOME/cluster_definition.conf.template cluster_definition.conf</pre> <p><b>Note</b> – All the configuration files must be in the same local directory on the installation server.</p> <p>The <code>cluster_definition.conf</code> file format is ASCII. Comment lines begin with the comment mark (<code>#</code>). Parameters consist of a keyword followed by an equals (<code>=</code>) sign followed by the parameter value, of the form:</p> <p><i>Keyword=Value</i></p> <p>Within the <i>Value</i>, you can use a slash at the end of a line to indicate that the <i>Value</i> is continued on the following line. You can also add comments within a <i>Value</i>. For example:</p>





## CLUSTER\_NETWORK

Define the class of IP addresses for your cluster network. The cluster network can have IP addresses of any class. The parameter has the following format:

```
CLUSTER_NETWORK=netmask nic0-subnet nic1-subnet
cgtp-subnet
```

- *netmask* is the mask common to all subnets configured in the `/etc/netmasks` file, for example, `255.255.0.0`.
- *nic0-subnet* is the subnet of the first network interface, *NIC0*. This subnet is configured in the `/etc/netmasks` file, for example, `172.15.0.0`.
- *nic1-subnet* is the subnet of the second network interface, *NIC1*. This subnet is configured in the `/etc/netmasks` file, for example, `172.16.0.0`.
- *cgtp-subnet* is the subnet of the virtual network interface, *cgtp0*. This subnet is configured in the `/etc/netmasks` file, for example, `172.17.0.0`.

By default, class C IP addresses are used as follows:

```
CLUSTER_NETWORK=255.255.255.0 10.clusterid.1.0
10.clusterid.2.0 10.clusterid.3.0
```

Where *clusterid* is the value of the `CLUSTER_ID` parameter.

## DATA\_MGT\_POLICY

Define how the cluster behaves when the vice-master node starts up while the master node is down. Options are:

## INTEGRITY

The vice-master waits for the old master to rejoin the cluster before it takes the master role. This ensures that the cluster uses the most up-to-date data.

## AVAILABILITY

The vice-master does not wait for the old master to rejoin the cluster before it takes the master role. Data written to the master while the vice-master is down is lost.

## ADAPTABILITY

The vice-master checks the disk synchronization state. If the state is not synchronized, that is the state returned by `nhcmmstat` is `synchro:NEEDED`, the vice-master waits for the old master to rejoin the cluster. If the state is synchronized, that is the state returned by `nhcmmstat` is `synchro:READY`, the vice-master is elected as the new master.

The default value for `DATA_MGT_POLICY` is `INTEGRITY`.

`DATA_MGT_POLICY=INTEGRITY`

## DATALESS

Define each dataless node.

There is an entry for each dataless node and each entry has the following format:

```
DATALESS=nodeid MAC0 - {name|-} {NIC0|-} {NIC1|-}
public-name public-IP public-NIC
```

- *nodeid*

The ID of the node used to define IP addresses for the node. This option is mandatory.

- *MAC0*

The Ethernet address of the first network interface of the node. This option is mandatory. This address is required to boot the dataless node from the installation server.

- -

The Ethernet address of the second network interface of the node. This option is mandatory. The *MAC1* address is ignored and you must specify a hyphen (-).

- *name*

Name of the node. By default, the names are assigned as follows:

`NMEN-Cclusterid-Nnodeid`

Do not use underscores (" \_ ") when naming a node. For more information, see `hosts(5)`.

- *NIC0*

Name of the first network interface. By default the first value defined for the `NMEN_INTERFACES` parameter is used for *NIC0*. To use the default value, you must specify a hyphen (-).

- *NIC1*

Name of the first network interface. By default the first value defined for the `NMEN_INTERFACES` parameter is used for *NIC1*. To use the default value, you must specify a hyphen (-).

- *public-name*

Name of the node on the public network different from the name defined with the *name* parameter. If `PUBLIC_NETWORK` is not defined, the *public-name* is ignored.

- *public-ip*

IP address of the node on the public network. If `PUBLIC_NETWORK` is not defined, the *public-ip* is ignored.

- *public-nic*

Network interface for the node supporting the public network. This can be either a physical network interface or an alias. If `PUBLIC_NETWORK` is not defined, the *public-nic* is ignored.

#### Example 1:

```
MEN=10 00:03:ba:f1:78:38 - node10
MEN=20 00:03:ba:f1:78:26 - node20
DATALESS=30 00:03:ba:f1:7b:b0 - node30
DATALESS=40 00:03:ba:f1:7b:21 - node40
```

**Example 2:**

```

MEN=10 00:03:ba:f1:78:38 - node10 - - FSNode1
192.168.12.5 eth2
MEN=20 00:03:ba:f1:78:26 - node20 - - FSNode2
192.168.12.6 eth2
DATALESS=30 00:03:ba:f1:7b:b0 - node30
DATALESS=40 00:03:ba:f1:7b:21 - node40

```

In these examples there are four entries, one entry for each of the master-eligible nodes and two entry for two dataless nodes. The `nhinstall` tool first installs the product on the first master-eligible node defined, then on the second master-eligible node, and then on the two dataless nodes.

To add dataless nodes to a cluster that is already running, add the definitions for the new nodes by using the `DATALESS` parameter and run the `nhinstall` command with the `add` option. For information about the `add` option of the `nhinstall` command, see the `nhinstall(8)` man page.

**DEFAULT\_ROUTER\_IP**

Defines the IP address of the default router for the public network. The `DEFAULT_ROUTER_IP` is set to an IP address.

```
DEFAULT_ROUTER_IP=IP address
```

The default is the public IP address of the installation server.

## DIRECT\_LINK

You can prevent the occurrence of two master nodes in one cluster by connecting the serial ports of the two master-eligible nodes and defining the `DIRECT_LINK` parameter. By default, this parameter is not configured.

The `DIRECT_LINK` parameter has the following format:

```
DIRECT_LINK=MEN1-serial-device  MEN2-serial-
device  speed  [heartbeat-in-seconds]
```

- *MEN1-serial-device* is the serial port on the first master-eligible node to use to connect to the second master-eligible node, for example, `/dev/ttyS1`
- *MEN2-serial-device* is the serial port on the second master-eligible node to use to connect to the second master-eligible node, for example, `/dev/ttyS1`
- *speed* is the serial line speed. Valid values for *speed* are 38400, 57600, 76800, and 115200.
- *heartbeat-in-seconds* is the frequency of the heartbeat checking the link between the two master-eligible nodes. The default value for heartbeat is 20 seconds.

## EXPORTED

The directory to be created and exported on the master-eligible node. There is an entry for every directory to be exported. Each entry has the following format:

```
EXPORTED=name
```

*name* is the directory to be exported. However, do not specify the following directories:

- `/SUNWcgha/local/export` because the `nhinstall` tool automatically creates this directory

**Note** – Exported directories must be on a replicated partition because these directories must be accessible regardless of the node that is master. Exporting a non-replicated directory results in errors during a switchover.

## EXTERNAL\_BONDING\_MASTER\_ADDRESS

Define an external IP address for the master node that is monitored using bonding (redundant link). The parameter has the following format:

```
EXTERNAL_BONDING_MASTER_ADDRESS=<hostname>
<netmask>\
<network> <master IP address> <bonding device>
\
<prim.
interface>
[ <secondary interface> ] [ <IPv6 hostname> \
\
<master IPv6 address> ] <ARP | MII>
\
<failure detection time> [ <list of target nodes> ]
```

- *hostname* is the host name associated with the master IP address. This name is added in /etc/hosts.
- *netmask* is the netmask of the subnetwork.
- *network* is the subnetwork.
- *master IP address* is the IP address that is set up when the node becomes the master node
- *bonding device* is the name of the bonding device to be created.
- *primary interface* is the first network interface to be part of the group.
- *secondary interface* is the second network interface to be part of the group. Use of this parameter is optional. If the parameter is not specified, the group will contain only one interface.
- *IPv6 hostname* is the host name associated with the master IPv6 address. This name is added in /etc/hosts.
- *master IPv6 address* is the IPv6 address that is set up when the node becomes the master node.

For example:

```
EXTERNAL_BONDING_MASTER_ADDRESS=men103-eam \
255.255.255.0 10.103.2.0 10.103.2.251 bond0 eth1 \
- men103ipv6-eam fec0::209:3dff:fe10:bf93 MII 100
```

## EXTERNAL\_MASTER\_ADDRESS

Define an external IP address for the master node. The parameter has the following format:

```
EXTERNAL_MASTER_ADDRESS=<hostname> <netmask> \  
<network> <master IP address> <interface>  
[ <IPv6 hostname> <master IPv6 address> ]
```

- *hostname* is the host name associated with the master IP address. This name is added in `/etc/hosts`.
- *netmask* is the netmask of the subnetwork.
- *network* is the subnetwork.
- *master IP address* is the IP address that is set up when the node becomes the master node
- *interface* is the network interface where the master IP address is configured.
- *IPv6 hostname* is the host name associated with the master IPv6 address. This name is added in `/etc/inet/ipnodes`.
- *master IPv6 address* is the IPv6 address that is set up when the node becomes the master node.

This address is managed according to the node mastership, but it is not monitored.

For more information, see the EXTERNAL\_BONDING\_ MASTER\_ADDRESS feature used for monitoring.

## HARDWARE

This directive must be used to specify the hardware type of your nodes as follows:

```
HARDWARE=nodeid hardware
```

- *nodeid*
- Corresponding to the id of the node. (See MEN, DISKLESS, and DATALESS.)
- *hardware*  
Can be `netra_cp3020` for x64 nodes, or `netra_cp3060` or `netra_t2000` for sun4v nodes. For Linux, this must be `netra_cp3020`.

**INSTALL\_NSM**

Install the Node State Manager. Options are YES and NO. The default is NO.

Set the `INSTALL_NSM` parameter to YES to enable NSM for uses other than external access.

**INSTALL\_SAFCLM**

Install the SA Forum/CLM package on all nodes. Options are YES and NO. The default is NO.

Set the `INSTALL_SAFCLM` parameter to YES to install the SA Forum/CLM API.

Set the `INSTALL_SAFCLM` parameter to NO to install the Foundation Services without installing SA Forum/CLM.

**LOGICAL\_SLICE\_SUPPORT**

Install the volume management feature (LVM2 on Linux). Options are YES and NO. The default is NO.

If `LOGICAL_SLICE_SUPPORT` is set to YES, the volume management feature is configured for managing replicated partitions and their associated bitmap partitions.

**MASTER\_ID**

Define the node id assigned to the master node. This allows you to create a master floating address. The default node id for the master is 1.



**MASTER\_LOSS\_DETECTION**

Define if the absence of a master node in the cluster must be detected by dataless nodes. Options are YES and NO. The default is YES.

If MASTER\_LOSS\_DETECTION is set to YES, the absence of a master node in the cluster is detected by the dataless nodes and if such an absence occurs, the dataless nodes are rebooted. If

MASTER\_LOSS\_DETECTION is NO, the absence of a master node in the cluster is not detected by dataless nodes.

**MASTER\_LOSS\_TIMEOUT**

Defines the amount of time the vice-master node will wait before taking over when it detects a stale cluster situation.

The value is expressed in seconds, with a minimum of 38 seconds.

The default value is 190 seconds.

**MEN**

Define each master-eligible node. It is mandatory to define this keyword.

When you define the MEN keyword, create an entry for each master-eligible node using the following format:

```
MEN=nodeid MAC0 - {name|-} {NIC0|-} {NIC1|-} \
public-name public-IP public-NIC
```

- *nodeid*

- The ID of the node used to define IP addresses for the node. This option is mandatory.

- *MAC0*

The Ethernet address of the first network interface of the node. This option is mandatory. This address is required to boot the master-eligible node from the installation server.

- -

The Ethernet address of the second network interface of the node is ignored and you must specify a hyphen (-).

- *name*

Name of the node. By default, the names are assigned as follows:

`MEN-Cclusterid-Nnodeid`

- *NIC0*

Name of the first network interface. By default the first value defined for the `MEN_INTERFACES` parameter is used for *NIC0*. To use the default value, you must specify a hyphen (-).

- *NIC1*

Name of the second network interface. By default the first value defined for the `MEN_INTERFACES` parameter is used for *NIC1*. To use the default value, you must specify a hyphen (-).

- *public-name*

Name of the node on the public network different from the name defined with the *name* parameter. If `PUBLIC_NETWORK` is not defined, the *public-name* is ignored.

- *public-ip*

IP address of the node on the public network. If `PUBLIC_NETWORK` is not defined, the *public-ip* is ignored.

- *public-nic*

Network interface for the node supporting the public network. This can be either a physical network interface or an alias. If `PUBLIC_NETWORK` is not defined, the *public-nic* is ignored.

**Example 1:**

```
MEN=10 00:03:ba:f1:78:38 - node10
MEN=20 00:03:ba:f1:78:26 - node20
```

**Example 2:**

```
MEN=10 00:03:ba:f1:78:38 - node10 - - FSNode1
192.168.12.5 eth2
MEN=20 00:03:ba:f1:78:26 - node20 - - FSNode2
192.168.12.6 eth2
```

There are two entries, one for each of the master-eligible nodes. The `nhinstall` tool first installs the product on the first master-eligible node defined and then on the second master-eligible node.

**MEN\_INTERFACES**

The network interfaces used for Carrier Grade Transfer Protocol (CGTP) on master-eligible nodes. The parameter has the following format:

```
MEN_INTERFACES=nic0 nic1
```

*nic0* is the name of the first network interface.

*nic1* is the name of the second network interface.

It is mandatory to define this parameter.

**MOUNTED**

The mount point on the master-eligible node of the directory that is mounted on the master-eligible nodes and the dataless nodes. There is an entry for each mount point, and each entry has the following format:

```
MOUNTED=mounting-point remote-dir
```

- *mounting-point*
- The mount point on each node.
- *remote-dir*

The directory name on the master-eligible node.

The following directories are automatically created by the `nhinstall` tool. Do not define them.

- The `/SUNWcgha/remote` directory, which is the mount point for the *men\_name*: `/SUNWcgha/local/export/data` directory.
- The `/SUNWcgha/services` directory, which is the mount point for the *men\_name*: `/SUNWcgha/local/export/services/ha_v1/opt`.
- The `/SUNWcgha/swdb` directory, which is the mount point for the *men\_name*: `/SUNWcgha/local/export/services` directory.

Where *men\_name* is the host name of the master-eligible nodes.

NFS\_USER\_DIR\_NOAC

Define the NFS `noac` option for remote mounted directories. The `noac` option suppresses data and attributes caching.

If you choose `YES`, the `noac` option is configured when mounting remote directories. In this case, all data is retrieved from the master node disk. Data and attribute caching is suppressed.

Alternatively, if you choose `NO`, the `noac` option is not configured. In this case, data is cached on the local node.

Use the `noac` option if the impact on performance is acceptable.

The default is `YES`, for example:

```
NFS_USER_DIR_NOAC=YES
```

NIC0\_POSTFIX, NIC1\_POSTFIX, CGTP\_POSTFIX

Defines a suffix string for each host name. These definitions can be set to change the suffix for each host name related to a specific network interface. By default the suffixes are:

- `NIC0_POSTFIX=""` Therefore, by default, the host name for the `NIC0` interface has no suffix.
- `NIC1_POSTFIX="-nic1"`
- `CGTP_POSTFIX="-cgtp"`

If you specify all the suffixes, as in the following example, the `nhinstall` tool creates two host names in the `/etc/hosts` file—one host name with the suffix and one host name without the suffix with respect to the `NIC0` interface.

For example, if the suffixes are:

```
NIC0_POSTFIX="-mynic0"
NIC1_POSTFIX="-mynic1"
CGTP_POSTFIX="-mycctp"
```

The resulting `/etc/hosts` file will include the following type of entry for each node:

```
10.250.1.10 netraMEN1 netraMEN1-mynic0
10.250.1.20 netraMEN2 netraMEN2-mynic0
10.250.1.30 netraDISKLESS1 netraDISKLESS1-mynic0
```

**Note** – Do not use underscores (“\_”) within a suffix. For more information, see `hosts(5)`.

#### NMEN\_INTERFACES

The network interfaces used for CGTP on diskless and dataless nodes. The parameter has the following format:

```
NMEN_INTERFACES=nic0 nic1
```

*nic0* is the name of the first network interface.

*nic1* is the name of the second network interface.

It is mandatory to define this parameter if dataless nodes are defined.

#### PASSWORD

The password for the superuser set for all nodes in the cluster. Set this password even if you manually install the Solaris OS on the master-eligible nodes because this password is required by the `nhinstall` tool to execute the commands that create the diskless environment.

By default, the password is `sunrules`.

## PUBLIC\_NETWORK

Define the public network IP addresses and netmasks for the cluster. The parameter has the following format:

PUBLIC\_NETWORK=netmask subnet

- *netmask* is the mask for the subnet. For example:  
255.255.255.0
- *subnet* is the public subnet configured in the  
/etc/netmasks file. For example: 192.168.0.0

**Note** – The cluster installation can be done either through this public network or through the private network:

- The installation server will be configured to perform the installation using the public network if the *SERVER\_IP* parameter is defined in the *env\_installation.conf* file (*SERVER\_IP* is public IP address for the installation server, part of the same subnet as the one described in *PUBLIC\_NETWORK*).
- The installation server will be configured to perform the installation from the private network if the *SERVER\_IP* parameter is not defined in the *env\_installation.conf* file.

## RESTRICT\_RHOSTS

Restrict the master-eligible nodes from connecting to each other remotely. Options are YES and NO. The default is NO.

If *RESTRICT\_RHOSTS* is set to YES, only the installation server can connect remotely to the master-eligible nodes with *rsh* and *rcp* commands. The master-eligible nodes cannot connect to each other remotely. Therefore, you will not be able to run *nhadm(8)* commands such as *synccheck* and *syncgen* on the master-eligible nodes.

If *RESTRICT\_RHOSTS* is set to NO, the master-eligible nodes can connect to each other remotely with the *rsh* and *rcp* commands. The installation server can also connect to the master-eligible nodes remotely.

## SERIALIZE\_SYNC

Determine how disk synchronization is performed. Values are YES and NO. The default value is NO.

Synchronization is necessary following a switchover or the vice-master booting, or when you request a full replication. In these circumstances, if SERIALIZE\_SYNC is set to NO, Reliable NFS starts the synchronization of all slices at the same time. If SERIALIZE\_SYNC is set to YES, slices are synchronized one slice at a time. This reduces the network and disk overhead but increases the time it takes for the vice-master to synchronize with the master. During this time, the vice-master is not eligible to take on the role of master.

## SHARE\_PROTECTION

Define the type of access control used to share directories exported from the master node. Options are PER\_NODE and PER\_SUBNETWORK. The default is PER\_NODE.

- PER\_NODE: This option increases the protection of each exported directory, but increases the switch-over and fail-over time. Each node is explicitly named for the common directories.
- PER\_SUBNETWORK: This option reduces the number of shared commands, which saves time during a switch-over or a fail-over with an acceptable level of protection. Only authorized subnetworks are listed for all exported directories.

## SLICE

Defines the disk partitioning. There is an entry for every partition on the disk, and each entry has the following format:

SLICE=name size mount-point bitmap option [scope]

- *name*

The name of the disk partition. For example, sda1.

- *size*

The size of the partition in Mbytes. For the last slice, the size can be set to free.

- *mount-point*

The name of the mount point of the partition. If the partition is used as a bitmap by DRBD, the mount point must be set to unnamed.

- *bitmap*

The name of the DRBD metadisk. If the partition is not replicated, specify a hyphen (-).

- *option*

Specify a hyphen (-) on Linux.

- *[scope]*

The scope of the definition of the slice. The scope is a comma separated list including values for `MEN`, `DATALESS`, and `nodeid` where the values listed define the nodes to which the slice definition applies. By specifying a type of node such as `MEN` or `DATALESS` you apply the slice definition to that type of node. By specifying the `nodeid` of nodes you can apply the slice definition to a particular set of nodes.

**Note** – If the replicated slices on the master node and the vice-master node have different names, you must use logical slice support. This ensures that the LVM2 is installed automatically and the anomaly between the replicated slice names is hidden.

**Note** – The restrictions due to partitions scheme on Linux apply here. To overcome the limit of four primary partitions, a primary partition is declared as extended and is partitioned into sub-partitions. The `nhinstall` tool will also consider that all partitions declared after the swap partition are extended partitions.



```
SLICE=sda1 4096/ - - MEN
SLICE=sda2 1024 swap - - MEN
SLICE=sda5 2048 /SUNWcgha/local sda7 - MEN
SLICE=sda 4096 /export sda7 - MEN
SLICE=sda7 256 unnamed - - MEN
SLICE=sda8 free /user1 - - MEN

SLICE=sda1 4096/ - - DATALESS
SLICE=sda2 1024 swap - - DATALESS
SLICE=sda5 2048 /user - - DATALESS
SLICE=sda 4096 /user1 - - DATALESS
SLICE=sda7 1024 /user2 - - DATALESS
SLICE=sda8 128 /user3 - - DATALESS
```

The following disk partitions are mandatory:

- The root partition, /
- The /SUNWcgha/local partition

SYNC\_FLAG

Delays disk synchronization at startup. Options are YES and NO. The default is YES.

If SYNC\_FLAG is set to NO, you delay the start of disk synchronization until you use the nhenablesync command. For more information, see the nhenablesync(8) man page.

USE\_CGTP

Install the CGTP on the cluster. Options are YES and NO. The default is YES.

If USE\_CGTP is set to NO, the CGTP packages and patches are not installed. Only the first network interface is considered in the definition of MEN and DATALESS. The CGTP\_POSTFIX, if specified, is ignored. In this case, you configure a single network link and your cluster network is not redundant.

EXAMPLES

**EXAMPLE 1** Example of a Standard Configuration for a Cluster With Two Master-Eligible Nodes and One Dataless Node

```
#
# This file enables you to define the cluster
# environment.
#
#-----
CLUSTER_ID=123
PUBLIC_NETWORK=255.255.255.0 10.17.252.0
SHARE_PROTECTION=PER_SUBNETWORK
```

```
USE_CGTP=YES
INSTALL_SAFCLM=YES
INSTALL_NSM=YES

MEN_INTERFACES=eth2 eth3
NMEN_INTERFACES=eth2 eth3

HARDWARE=10 netra_cp3020
HARDWARE=20 netra_cp3020
HARDWARE=30 netra_cp3020

MEN=10 00:03:ba:f1:76:38 - men123-1 - - andretti-1
10.17.252.217 eth2
MEN=20 00:03:ba:f1:78:26 - men123-2 - - andretti-2 2
10.17.252.218 eth
DATALESS=30 00:03:ba:f1:7b:b0 - men123-4 - - andretti-3
10.17.252.219 eth2
SLICE=sda1 4096 / - - MEN
SLICE=sda2 1024 swap - - MEN
SLICE=sda5 2048 /SUNWcgha/local sda7 - MEN
SLICE=sda6 4096 /export sda7 - MEN
SLICE=sda7 256 unnamed - - MEN
SLICE=sda8 free /user1 - - MEN

SLICE=sda1 4096 / - - DATALESS
SLICE=sda2 1024 swap - - DATALESS
SLICE=sda5 2048 /user - - DATALESS
SLICE=sda6 4096 /user1 - - DATALESS
SLICE=sda7 1024 /uzer2 - - DATALESS
SLICE=sda8 128 /user3 - - DATALESS
```

**SEE ALSO** nhinstall(8), nhadm(8), nhenablesync(8), addon.conf(5),  
env\_installation.conf(5), hosts(5)

NAME	cluster_nodes_table - central cluster management file																
SYNOPSIS	/etc/opt/sun/nhas/cluster_nodes_table																
DESCRIPTION	<p>The cluster_nodes_table contains a definition for each node in the cluster. The cluster_nodes_table is located on each master-eligible node. This file stores the membership and configuration information for all peer nodes in a cluster including potential future nodes. A node must have an entry in the cluster_nodes_table to be part of a cluster.</p> <p>A template of the cluster_nodes_table file is available in /etc/opt/sun/nhas/cluster_nodes_table.template.</p> <p>By default, the cluster_nodes_table is located in the /etc/opt/sun/nhas/ directory on each master-eligible node. You can change the path to the cluster node table by editing the parameter CMM.LocalConfig.Dir parameter in the nhfs.conf file. For more information on nhfs.conf, see the nhfs.conf(5) man page.</p> <p>The following is an example of a cluster node table with three peer nodes:</p> <p>VERSION 2</p> <table><tr><td>#NodeId</td><td>Domain_id</td><td>Name</td><td>Attributes</td></tr><tr><td>10</td><td>250</td><td>netraMEN1-cgtp0</td><td>-</td></tr><tr><td>20</td><td>250</td><td>netraMEN2-cgtp0</td><td>-</td></tr><tr><td>30</td><td>250</td><td>netraDISKLESS1-cgtp0</td><td>-</td></tr></table> <p>NodeId                      This is the unique node ID within the cluster. This number is the decimal equivalent of the host part of the node's IP address.</p>	#NodeId	Domain_id	Name	Attributes	10	250	netraMEN1-cgtp0	-	20	250	netraMEN2-cgtp0	-	30	250	netraDISKLESS1-cgtp0	-
#NodeId	Domain_id	Name	Attributes														
10	250	netraMEN1-cgtp0	-														
20	250	netraMEN2-cgtp0	-														
30	250	netraDISKLESS1-cgtp0	-														

Domain_id	This is the unique cluster ID within the cluster domain and must be the same for each peer node. This Domain_id must be the same as the one defined in the nhfs.conf file. For more information see the nhfs.conf(5) man page.
Name	This is the node name. The name must be the same as the one in the nhfs.conf file on the node. See the nhfs.conf(5) man page.
Attributes	Each node can be assigned the following attributes:  <b>Note</b> – Do not alter the role of a node by modifying the attribute column of cluster_nodes_table file.

D	DisqualifiedD corresponds to the CMM_DISQUALIFIED_MEMBER attribute of the CMM API. If a node is flagged as disqualified, it cannot be assigned a master or vice-master role. This attribute is only applied to a master-eligible node.
S	Synchronization needed. If a node has the S flag, the disks of the master node and vice-master nodes are not synchronized. This is a read-only flag and must not be changed manually. This attribute applies only to master-eligible nodes.
-	Not DisqualifiedIf a node has this attribute, the node is not disqualified and is in the cluster.

To check the role of a node, use the nhcmmstat command, as described in the nhcmmstat(8) man page.

**EXTENDED  
DESCRIPTION**

If you want to add a new node to a cluster you must either verify that the node already has an entry in the cluster node table or create an entry for this node in the table. You must only modify this file during initial cluster configuration if you install the software manually on the cluster or if you are adding or removing a node. For more information, see the cmm\_config\_reload(3CMM) man page.

When editing the cluster node table file, make sure that:

1. The NodeId for each node in the cluster\_nodes\_table is unique. If this is not the case, use nhcmmstat to determine the correct NodeId of the peer nodes and modify the cluster node table.
2. The NodeId for each node in the cluster\_nodes\_table has a value *n* such that  $3 < n < 255$ . If this is not the case, modify the cluster node table.

3. The master-eligible nodes have the attribute “-” in `cluster_nodes_table`. If this is not the case, correct this error in the file.

All other updates are performed automatically by the `nhcmmmd` daemon. The `nhcmmmd` daemon on the master node uses the cluster node table to know which nodes are in the cluster and what attributes they are assigned.

The `cluster_nodes_table` is read and changed automatically by the `nhcmmmd` daemon as follows:

1. When the master node is elected:
  - a. The master node reads the `cluster_nodes_table` and stores it in memory.
  - b. The master node broadcasts this view of the `cluster_nodes_table` to peer nodes in the cluster.
  - c. Each master-eligible node receives this view and stores this view of the `cluster_nodes_table`.
2. When the cluster is running:
  - a. After the cluster is up and running, the `cluster_nodes_table` is only read when a `cmm_config_reload()` is called or when the following command is executed:
 

```
# nhcmmstat -c reload
```
  - b. The master node reads and stores the new view of the `cluster_nodes_table` in memory.
  - c. The master node broadcasts the new view to peer nodes in the cluster.
  - d. The vice-master node receives this view and stores this view of the `cluster_nodes_table`.
3. The `cluster_nodes_table` is modified when an attribute of a node has been changed through the API, for example, `cmm_member_setqualif()`.

The preceding methods of creating and managing the `cluster_nodes_table` file, ensures that the `cluster_nodes_table` is *persistent*, that is, if the cluster goes down for any reason, the same `cluster_nodes_table` file is restored when the cluster comes back up, no matter which master-eligible nodes is elected master.

## WARNINGS

Changes should not be made to the cluster node table without careful consideration because the cluster node table maintains the central view of the cluster membership and node status. Altering the cluster node table can result in a cluster that is not highly available.

## SEE ALSO

`nhcmmmd(8)`, `nhadm(8)`, `nhfs.conf(5)`, `nhcmmstat(8)`,  
`cmm_config_reload(3CMM)`



NAME	env_installation.conf - nhinstall configuration file defining the installation environment
SYNOPSIS	<b>env_installation.conf</b>
DESCRIPTION	<p>Configure the env_installation.conf file to define the installation environment for the nhinstall tool.</p> <p>The templates for the configuration files are contained in the /opt/sun/nhas/templates/nhinstall directory with .template extensions. Copy the configuration files to a local directory on the installation server as follows:</p> <pre># mkdir config-file-directory # export NHOME=/opt/sun/nhas/templates/nhinstall # cd config-file-directory # cp \$NHOME/env_installation.conf.template env_installation.conf</pre> <p><b>Note</b> – All the configuration files must be in the same local directory on the installation server.</p> <p>The env_installation.conf file format is ASCII. Comment lines begin with the comment mark (#). Parameters consist of a keyword followed by an equals (=) sign followed by the parameter value, of the form:</p> <p><i>Keyword=Value</i></p> <p>The following <i>Keyword</i> and <i>Value</i> parameters are supported:</p>

EXTENDED  
DESCRIPTION

Modify variables in the `env_installation.conf` file as follows:

SERVER_INTERFACE	<p>The network interface of the installation server used to access the cluster when installing the product, for example:</p> <pre>SERVER_INTERFACE=hme1</pre>
SERVER_NODE	<p>The node ID of the installation server within the cluster. This must be a unique value between 3 and 254, for example:</p> <pre>SERVER_NODE=253</pre>
SERVER_IP	<p>The IP address of the installation server. This IP address is required only when:</p> <ul style="list-style-type: none"><li>■ A public network is used, that is, the <i>PUBLIC_NETWORK</i> variable is defined in <code>cluster_definition.conf</code>.</li><li>■ The installation server has to be configured to use this public network to install the cluster nodes.</li></ul> <p>If you do not define the <i>SERVER_IP</i> parameter, the installation server will use the private network to install the cluster nodes, and an IP address will be automatically created for the installation server based on the cluster ID (<i>CLUSTER_ID</i> parameter defined in <code>cluster_definition.conf</code>), and the node ID (<i>SERVER_NODE</i> parameter defined in <code>env_installation.conf</code>).</p> <p>Example of <i>SERVER_IP</i> definition:</p> <pre>SERVER_IP=192.168.12.50</pre> <p><b>Note</b> – If you plan to use the public network to install cluster nodes, make sure that the IP address specified in the <i>SERVER_IP</i> parameter is on the same subnet as the one specified in the <i>PUBLIC_NETWORK</i> parameter.</p>
AUTO_REBOOT	<p>When launched, the <code>nhinstall</code> tool automatically reboots the nodes as required during the installation process.</p> <p>Options are YES and NO. The default is YES.</p>



OS_INSTALL	<p>Install the operating system. Options are ALL, NONE, DISKLESS_ONLY, and DISKLESS_DATALESS_ONLY. The default is ALL.</p> <p>If you choose NONE, the operating system is not installed on the cluster nodes. In this case, make sure of the following before launching the <code>nhinstall</code> tool:</p> <ul style="list-style-type: none"> <li>■ The operating system is already installed on the nodes.</li> <li>■ Java Development Kit (JDK) is already installed on the cluster nodes.</li> </ul> <p>If you choose DISKLESS_DATALESS_ONLY, the operating system (Solaris or Linux) is not installed on the master-eligible nodes. You must make sure that the operating system is already installed on the nodes before running the <code>nhinstall</code> tool.</p>
NHAS_PRODUCT_DIR	<p>The directory containing the Foundation Services distribution, for example:</p> <p><code>NHAS_PRODUCT_DIR=/cdrom</code></p>
WORKING_DIR	<p>The directory on the installation server where temporary files are created during the installation process. This directory must be writable and shared. The progress indicator used for installation recovery is also stored in this directory. Do not specify the <code>/tmp</code> directory, which is deleted in the event of a reboot.</p> <p><code>WORKING_DIR=/export/nhttp</code></p>
MVISTA_TARGET_DIR	The location of the MontaVista Target distribution.
MVISTA_LSP_DIR	The location of the MontaVista Linux Support Package distribution.
DATALESS_MVISTA_TARGET_DIR	The location of the MontaVista Target distribution for dataless nodes, if this distribution is not the same as the distribution installed on the master-eligible nodes in the cluster. By default, the distribution installed on the master-eligible nodes is used for dataless nodes.

DATALESS_MVISTA_LSP_DIR	The location of the MontaVista Linux Support Package distribution for dataless nodes, if this distribution is not the same as the distribution installed on the master-eligible nodes in the cluster. By default, the distribution installed on the master-eligible nodes is used for dataless nodes.
WINDRIVER_IMAGES_DIR	Location of the Wind River CGL distribution.
WINDRIVER_ROOTNFS_DIR	Directory where a root NFS file system will be created for each type of platform that has to be installed with Wind River CGL. This root NFS file system will be mounted by the cluster nodes during the process of Wind River installation on the node's disks.
DATALESS_WINDRIVER_IMAGES_DIR	Location of the Wind River CGL distribution for dataless nodes, if this distribution is not the same as the distribution installed on the master-eligible nodes in the cluster. By default, the distribution installed on the master-eligible nodes is used for dataless nodes.
DATALESS_WINDRIVER_ROOTNFS_DIR	Directory where a root NFS file system will be created for each type of platform that has to be installed with Wind River CGL on the dataless nodes, if the distribution is not the the same as the distribution installed on the master-eligible nodes in the cluster. By default, the distribution installed on the master eligible nodes is used for dataless nodes. This root NFS file system will be mounted by the dataless nodes during the process of Wind River installation on the nodes disks.

**EXAMPLES****EXAMPLE 1** Sample env\_installation.conf File for a MontaVista Cluster

```

#
# Installation server network interface used to access the cluster.
#
SERVER_INTERFACE=eth1

#
# The node id attributed to the installation server within the cluster.
#
SERVER_NODE=253

#
# Install the operating system on the master-eligible nodes.
# Options are ALL, NONE, or DISKLESS_DATALESS_ONLY. The default is ALL.
OS_INSTALL=ALL

#
# IMPORTANT:
# All directories mentioned below will be exported (via the share command)
# Please check that they can be exported. The directories must not be the
# child of a directory already exported.
#

#
# Location of the Netra HA Suite Distribution
#
NHAS_PRODUCT_DIR=/export/nhas-dist

#
# Define the working directory where:
# . the progress indicator file is stored.
# . temporary files are stored.
# This directory must be writable.
#
WORKING_DIR=/export/nhtmp

#
# Location of the MontaVista target distribution
MVISTA_TARGET_DIR=/export/mvista/target

# Location of the MontaVista LSP distribution
MVISTA_TARGET_DIR=/export/mvista/lsp

```

**EXAMPLE 2** Sample env\_installation.conf File for Heterogeneous Cluster With Master-Eligible Nodes Running Solaris and Dataless Nodes Running MV

## Linux

```

#
# Installation server network interface used to access the cluster.
#
SERVER_INTERFACE=hme1

#
# The node id attributed to the installation server within the cluster.
#
SERVER_NODE=253

#
# Install the operating system on the master-eligible nodes.
# Options are ALL, NONE, or DISKLESS_ONLY. The default is ALL.
OS_INSTALL=ALL

#
# IMPORTANT:
# All directories mentioned below will be exported (via the share command)
# Please check that they can be exported. The directories must not be the
# child of a directory already exported.
#

#
# Location of the Netra HA Suite Distribution
#
NHAS_PRODUCT_DIR=/export/nhas-dist

#
# Define the working directory where:
# . the progress indicator file is stored.
# . temporart files are stored.
# This directory must be writable.
#
WORKING_DIR=/export/nhtmp

#
# Location of the Solaris distribution on the installation server.
# To be used on master-eligible nodes
#
SOLARIS_DIR=/export/solaris-dist

# Location of the MontaVista target distribution.
# To be used on dataless nodes
DATALESS_MVISTA_TARGET_DIR=/export/mvista/target

# Location of the MontaVista LSP distribution.
# To be used on dataless nodes
DATALESS_MVISTA_TARGET_DIR=/export/mvista/lsp

```

**SEE ALSO** `nhinstall(8)`, `cluster_definition.conf(5)`, `addon.conf(5)`

<b>NAME</b>	<code>nhadmsync.conf</code> - list of nonreplicated files and the differences between them
<b>SYNOPSIS</b>	<code>/SUNWcgha/remote/etc/nhadmsync.conf</code>
<b>DESCRIPTION</b>	<p>The <code>nhadmsync.conf</code> file is the configuration file for the <code>nhadm synccheck</code> and <code>nhadm syncgen</code> commands. The <code>nhadm synccheck</code> command compares nonreplicated files on the master and the vice-master nodes, printing any differences to the console. You can accept the differences using the <code>nhadm syncgen</code> command.</p> <p>To use these commands, both master-eligible nodes must have remote access to the other master-eligible node. For details on how to enable this, see the <code>nhadm(8)</code> man page.</p>
<b>EXTENDED DESCRIPTION</b>	<p>In the <code>nhadmsync.conf</code>, you must specify the non shared files that you want to compare. The default location of this file is <code>/SUNWcgha/remote/etc/nhadmsync.conf</code>. The name and location of this file can be changed at any time and is specified in the <code>-y   -syncfile</code> option, when using the <code>nhadm synccheck</code> command.</p> <p>You can create multiple versions of the <code>nhadmsync.conf</code> file. This enables you to have lists that are specific to a feature or a group of features, as described in the <code>nhadm(8)</code> man page. The <code>nhadm synccheck</code> configuration files must have write permissions if you want to use the <code>nhadm syncgen</code> command.</p> <p>To use the <code>nhadmsync.conf</code> file, copy the template file <code>/opt/sun/nhas/templates/nhadm/nhadmsync.conf.template</code> to <code>/SUNWcgha/remote/etc/nhadmsync.conf</code>.</p> <p>Add the names of the files to be compared, to the <code>nhadmsync.conf</code> file. Make sure that the filenames you add have the following criteria:</p> <ul style="list-style-type: none"> <li>■ The files exist on both master-eligible nodes</li> <li>■ The files are not replicated on a shared file system</li> </ul> <p>The syntax of your entries in this file must be the following:</p> <pre> NODEID=node1 node2 FILE=filename1 =BEGIN ... =END FILE=filename2 =BEGIN ... =END </pre>

NODEID	<p><i>node1</i> and <i>node2</i> are the node IDs of the master and vice-master nodes, respectively. If these are not present, the default logical IP addresses of the master and vice-master nodes are used.</p> <p>When <code>nhadm syncgen</code> is executed, the NODEID parameter is generated with the actual node IDs of both nodes to ensure the comparison is always made in the same order. This is because the <code>diff -b</code> command is dependent on the order of the files.</p> <p>If the NODEID parameter is present, it must be the first line of the <code>nhadmsync.conf</code> file and can only be preceded by a blank line.</p>
FILE	The name of the file to be tested.
=BEGIN...=END	This contains the result of the <code>diff -b</code> command for the file specified by the preceding FILE parameter.

**EXAMPLES**

**EXAMPLE 1** Example of information added by `syncgen` after the file comparisons

If you defined the `nhadmsync.conf` file as follows:

```
NODEID=10 20
FILE=/etc/ethers
FILE=/etc/hosts
FILE=/etc/netmasks
```

After the `nhadm syncgen` command is executed, the `nhadmsync.conf` file might contain the following information:

```
NODEID=10 20
FILE=/etc/ethers
FILE=/etc/hosts
=BEGIN
5c5,6
  10.250.1.10  MEN-C250-N10  loghost
---
> 10.250.1.20  MEN-C250-N20  loghost
> 10.250.1.10 MEN-C250-N10
8d8
```

```
10.250.1.20 MEN-C250-N20
=END
FILE=/etc/netmasks
```

The differences printed to the `nhadmsync.conf` file are the differences that would be found by running `diff -b` on the files listed in `nhadmsync.conf`. For more information on the `diff` command, see the `diff(1)` man page.

**SEE ALSO**

`nhadm(8)`, `diff(1)`





NAME	nhfs.conf - Foundation Services configuration file
SYNOPSIS	<code>/etc/opt/sun/nhas/nhfs.conf</code>
DESCRIPTION	<p>The <code>nhfs.conf</code> file contains configuration information for the Foundation Services such as the Cluster Membership Manager, Reliable NFS, and the Node State Manager. This file also provides cluster addressing and interface configuration information.</p> <p>Configure this file on each node if you plan to install the Foundation Services manually. To manually configure the parameters in the <code>nhfs.conf</code> file, uncomment the parameters, that is, delete the comment mark (#) at the beginning of the line, and modify the value of the parameter. A description of each parameter is provided in the following sections.</p> <p>When you have manually installed the Foundation Services packages, copy the template file <code>/etc/opt/sun/nhas/nhfs.conf.template</code> to the default location <code>/etc/opt//sun/nhas</code> as <code>nhfs.conf</code> on each peer node. For each file, make the necessary modifications in a text editor.</p> <p>Do not re-edit the <code>nhfs.conf</code> file when the cluster is running.</p> <p>The <code>nhfs.conf</code> file format is ASCII. Parameters consist of a keyword followed by an equals (=) sign followed by the parameter value, of the form:</p> <p style="margin-left: 40px;"><i>Keyword=Value</i></p> <p>The following <i>Keyword</i> and <i>Value</i> parameters are supported.</p> <p>Cluster.DataManagementPolicy</p> <p style="margin-left: 40px;">When IP-based replication is used, define how the cluster behaves when the vice-master node starts up while the master node is down. You can select one of three data management policies. Values are Integrity, Availability, and Adaptability. The default value is Integrity. The only value that applies for shared-disk configurations is Availability. The format for this property is:</p> <p style="margin-left: 40px;"><code>Cluster.DataManagementPolicy=Integrity   Availability   Adaptability</code></p>
COMMON PARAMETERS	

### Integrity

If there is a vice-master but no master in the cluster, the vice-master waits for the old master to rejoin the cluster before it takes the master role. This ensures that the cluster has the most up-to-date data. Choosing this value might reduce the availability of the cluster, but it prioritizes data integrity.

### Availability

For either configuration, if there is a vice-master in the cluster but no master node, the vice-master does not wait for the old master to rejoin the cluster before taking on the master role. The previous master is still off-line. For configurations using IP-based replication, any data that is written to the new master while there is no vice-master will be lost. When the old master node comes back on line as the vice-master node, a full synchronization occurs between the two master-eligible nodes.

### Adaptability

If there is a vice-master in the cluster but no master, the vice-master checks the disk synchronization state. If the state is not synchronized, that is the state returned by `nhcmmstat -c vice` is `synchro:NEEDED`, the vice-master waits for the master to come up. This is equivalent to the `Integrity` data management policy. If the state is synchronized, the state returned by `nhcmmstat -c vice` is `synchro:READY`, the vice-master is elected the new master. This is equivalent to the `Availability` data management policy.

### `Cluster.Master.ID`

Specify the host part of the master node's floating IP address in decimal form. The floating address triplet for the node with the master role is calculated from the local network interface addresses, the netmask, and the value of this parameter. These addresses are IPv4 address.

### Integrity

If there is a vice-master but no master in the cluster, the vice-master waits for the old master to rejoin the cluster before it takes the master role. This ensures that the cluster has the most up-to-date data. Choosing this value might reduce the availability of the cluster, but it prioritizes data integrity.

### Availability

For either configuration, if there is a vice-master in the cluster but no master node, the vice-master does not wait for the old master to rejoin the cluster before taking on the master role. The previous master is still off-line. For configurations using IP-based replication, any data that is written to the new master while there is no vice-master will be lost. When the old master node comes back on line as the vice-master node, a full synchronization occurs between the two master-eligible nodes.

### Adaptability

If there is a vice-master in the cluster but no master, the vice-master checks the disk synchronization state. If the state is not synchronized, that is the state returned by `nhcmmstat -c vice` is `synchro:NEEDED`, the vice-master waits for the master to come up. This is equivalent to the Integrity data management policy. If the state is synchronized, the state returned by `nhcmmstat -c vice` is `synchro:READY`, the vice-master is elected the new master. This is equivalent to the Availability data management policy.

### Cluster.Master.ID

Specify the host part of the master node's floating IP address in decimal form. The floating address triplet for the node with the master role is calculated from the local network interface addresses, the netmask, and the value of this parameter. These addresses are IPv4 address.

	<p>The default value is 1.  <code>Cluster.Master.ID=1</code></p> <p>If the <code>Cluster.Master.ID</code> is 1 and the cluster network is set up as follows:</p> <ul style="list-style-type: none"> <li>■ Netmask: 255.255.255.0</li> <li>■ <code>NIC0</code> subnet: 192.200.168.0</li> <li>■ <code>NIC1</code> subnet: 10.250.2.0</li> <li>■ <code>cgtp0</code> subnet: 192.200.175.0</li> </ul> <p>The floating address triplet is as follows:</p> <ul style="list-style-type: none"> <li>■ <code>NIC0</code> floating address: 192.200.168.1</li> <li>■ <code>NIC1</code> floating address: 10.250.2.1</li> <li>■ <code>cgtp0</code> floating address: 192.200.175.1</li> </ul> <p>This address is calculated similarly if you have IP addresses of any other class.</p>
<code>Node.Domainid</code>	<p>Specify the <i>domainid</i> of the cluster. You must modify this parameter. There is no default value.  <code>Node.Domainid=250</code></p>
<code>Node.NodeId</code>	<p>This parameter specifies the node ID of the current node. There is no default value.</p>
<code>Node.NIC0</code>	<p>The name of the first network interface, <i>NIC0</i>, used for CGTP. The default value is <code>hme0</code>.  <code>Node.NIC0=hme0</code></p> <p>This parameter could be a logical network interface, for example, <code>hme0:2</code>.</p> <p>If you have not installed the CGTP patches and packages and you want to configure a single network link for your cluster, configure this parameter.</p>

`Node.NIC1`

The name of the second network interface, *NIC1*, used for CGTP. The default value is `hme1`.

`Node.NIC1=hme1`

This parameter could be a logical network interface, for example, `hme1:2`.

If you have not installed the CGTP patches and packages and you want to configure a single network link for your cluster, do not configure this parameter.

`Node.NICCGTP`

The virtual interface used by CGTP. The default value is `cgtp0`.

`Node.NICCGTP=cgtp0`

If you have not installed the CGTP patches and packages and you want to configure a single network link for your cluster, do not configure this parameter.

`Node.UseCGTP`

Specify whether the CGTP is to be used or not. Values are `True` or `False`. The default value is `True`.

`Node.UseCGTP=True`

If you have not installed the CGTP patches and packages and you want to configure a single network link for your cluster, set this parameter to `False`.

`Node.RNFS.Installed`

Specify whether Reliable NFS is installed on a node. Values are `True` or `False`. The default value is `False`.

`Node.RNFS.Installed=True`

Both master-eligible nodes must have the same value for this parameter.

`Node.StartupMessagesOnConsole`

Specify whether or not service boot messages should be displayed in the console. Values are `True` (display) or `False` (do not display). The default value is `False` on the Solaris OS and `True` on Linux.

`Node.RNFS.Installed=True`

`Node.RNFS.Installed=False`

**DIRECT LINK  
PARAMETERS**`Node.Type`

Specify the type of node. Defining this parameter is mandatory. When you create the `nhfs.conf` file on each peer node, specify the type of node in this parameter. The `Node.Type` parameter can have one of the following values:

- `Diskfull`—A master-eligible node
- `Dataless`—A dataless node

There is no default value. For more information on types of nodes, see the *Netra High Availability Suite Foundation Services Overview*.

To prevent a split brain situation, you can connect the serial ports of the master-eligible nodes and configure the following parameters. Split brain is a situation where the network fails and two master nodes are elected in the same cluster. Configure the following parameters if you have connected the serial ports of the master-eligible nodes.

`Node.Direct-Link.Serial.Device`

Specify the serial device, that is, the system's serial ports. There is no default value.

```
Node.Direct-Link.serial.Device=/dev/term/b
```

`Node.Direct-Link.Serial.Speed`

Specify the serial line speed. Valid values are 38400, 57600, 76800, or 115200. The higher the value, the better the link.

There is no default value.

```
Node.Direct-Link.serial.Speed=115200
```

`Cluster.Direct-Link.Backend`

## CMM PARAMETERS

This parameter enables the direct link. If this parameter is not present in the `nhfs.conf` file, the direct link will not be used even if you have connected the serial ports of the master-eligible nodes.

The only value accepted by this parameter is `serial`:

```
Cluster.Direct-Link.Backend=serial
```

```
Cluster.Direct-Link.Heartbeat
```

Specify the number of seconds between two checks to detect a failure. Therefore, if one master-eligible node receives no heartbeat during the specified period of seconds, the node is alerted that there may be a problem.

There is no default value. For example, specify an interval of 20 seconds as follows:

```
Cluster.Direct-Link.Heartbeat=20
```

```
CMM.IsEligible
```

Specify whether the node is master eligible. Values are `True` or `False`. The default value is `False`.

```
CMM.IsEligible=True
```

```
CMM.LocalConfig.Dir
```

Specify the directory where the configuration file, `cluster_nodes_table`, is located on each master-eligible node. There is no default value.

```
CMM.LocalConfig.Dir=/etc/opt/SUNWcgha
```

```
CMM.MasterLoss.Detection
```

Determine whether nodes must reboot in the absence of a master node in the cluster. Values are `True` and `False`. The default is `True`.

	<p>If <code>CMM.MasterLoss.Detection</code> is set to <code>True</code>, the nodes are rebooted when they detect that there is no master node in the cluster for a period determined by <code>CMM.MasterLoss.Timeout</code> plus 8 seconds. If <code>CMM.MasterLoss.Detection</code> is set to <code>False</code>, nodes do not reboot if there is no master node in the cluster. However, the nodes cannot access directories that would be exported by the master node if such a node existed. You must ensure that nodes can adapt to this situation.</p>
<code>CMM.MasterLoss.Timeout</code>	<p>Used to specify the amount of time the vice-master node will wait before taking over when it detects a stale cluster situation. The value is expressed in seconds with a minimum value of 38 seconds. The default value is 190.</p>
<code>CMM.Miniconfig.Dir</code>	<p>Specify the directory where the configuration file, <code>target.conf</code>, is located on each master-eligible node. There is no default value.</p> <p><code>CMM.Miniconfig.Dir=/etc/opt/SUNWcgha</code></p>
<code>CMM.Miniconfig.File</code>	<p>Specify the name of the local configuration file, <code>target.conf</code>. There is no default value.</p> <p><code>CMM.Miniconfig.File=target.conf</code></p>
<code>CMM.StartUp.Join</code>	<p>Determine if nodes will automatically try to join the cluster at startup.</p> <p>Possible values are <code>True</code> and <code>False</code>. The default value is <code>True</code>.</p>
<code>PMDADM.nhcmmmd.ActionScript</code>	<p>For information, refer to <code>PMDADM.&lt;nametag&gt;.ActionScript</code> in the PMD parameters section.</p> <p>The default value for the CMM daemon is <code>/etc/opt/sun/nhas/init.d/default_critical_actionscript.sh</code></p>



PMDADM.nhcmmd.ActionScriptDelay

For information, refer to  
PMDADM.<nametag>.ActionScriptDelay in the PMD parameters section.

PMDADM.nhcmmd.Criticality

For information, refer to  
PMDADM.<nametag>.Criticality in the PMD parameters section.

The default value for the CMM daemon is TRUE.

PMDADM.nhprobed.ActionScript

For information, refer to  
PMDADM.<nametag>.ActionScript in the PMD parameters section.

The default value for the Probe daemon is  
/etc/opt/sun/nhas/init.d/  
default\_critical\_actionscript.sh

PMDADM.nhprobed.ActionScriptDelay

For information, refer to  
PMDADM.<nametag>.ActionScriptDelay in the PMD parameters section.

PMDADM.nhprobed.Criticality

For information, refer to  
PMDADM.<nametag>.Criticality in the PMD parameters section.

The default value for the Probe daemon is TRUE.

For more information on the CMM parameters, see the nhcmmd(8) man page.

## RELIABLE NFS PARAMETERS (GENERAL)

The following properties are common to all replication methods:

`RNFS.StatdAlternatePath`

Specify the alternate stated repository. For information about `statd`, see the `statd(8)` man page.

You must modify this parameter. There is no default value.

`RNFS.NFSAlternatePath=directory-path`

where *directory-path* is the path to the stated directory. This directory must be on a replicated disk partition.

`RNFS.NFSAlternatePath=/SUNWcgha/local`

**Note** – As of the 3.0 11/06 release of the Netra HA Suite product, this property has been deprecated. It is still supported, however, the property `RNFS.NFSAlternatePath` should now be used instead.

`RNFS.Share`

Describes the file systems to be shared. You must modify this parameter.

- This parameter uses the same syntax as the `shareexportfs` command. For more information, see the `shareexportfs(8)` man page.
- Each `shareexportfs` command must be on its own line. The first string of an `RNFS.Share` line must be either `shareexportfs` or `/usr/sbin/shareexportfs`.
- Each `RNFS.Share` line must contain one `shareexportfs` command line per partition to be exported by NFS.
- Each `RNFS.Share.x` represents a partition to be shared and *x* is an integer (0, 1, 2) that distinguishes each partition to be replicated.

To ensure that the CMM behaves correctly, you must:

- Grant superuser access for all the master-eligible nodes on the exported file system where the `cluster_nodes_table` file resides.
- Use the addresses for the CGTP interface for the master-eligible nodes. If CGTP is not installed, use the *NIC0* addresses.

There is no default value. In the following example, `cgtp6` and `cgtp7` represent the static CGTP addresses of the master-eligible nodes.

```
RNFS.Share.0=exportfs -i -o \
rw,no_root_squash,anonuid=0,anongid=0 \
cgtp6:/SUNWcggha/local/export \
cgtp7:/SUNWcggha/local/export \
master-cgtp:/SUNWcggha/local/export
```

**Note** – This property can be modified without fully shutting down the cluster; changes are applied after a successive clean reboot of both MEN nodes.

#### RNFS.Mode

Define the way RNFS replication is handled. The only valid choice for this parameter currently is: `RNFS.Mode =DRBD`.

DRBD

DRBD is used as the IP-based replication mechanism.

#### PMDADM.nhcrfsd.ActionScript

For information, refer to `PMDADM.<nametag>.ActionScript` in the PMD parameters section.

The default value for the RNFS server daemon is:  
`/etc/opt/sun/nhas/init.d/  
default_critical_actionscript.sh`

#### PMDADM.nhcrfsd.ActionScriptDelay

For information, refer to `PMDADM.<nametag>.ActionScriptDelay` in the PMD parameters section.

`PMDADM.nhcrfsd.Criticality`

For information, refer to  
`PMDADM.<nametag>.Criticality` in the PMD  
parameters section.

The default value for the RNFS server daemon is  
TRUE.

`PMDADM.nhcrfsd.MaxRetries`

For information, refer to  
`PMDADM.<nametag>.MaxRetries` in the PMD  
parameters section.

The default value for the RNFS server daemon is 2.

`PMDADM.nhcrfsd.MinPeriod`

For information, refer to  
`PMDADM.<nametag>.MinPeriod` in the PMD  
parameters section.

The default value for the RNFS server daemon is  
60.

`PMDADM.nhcrfscIntd.Criticality`

For information, refer to  
`PMDADM.<nametag>.Criticality` in the PMD  
parameters section.

The default value for the RNFS client daemon is  
FALSE.

`PMDADM.nhcrfscIntd.MaxRetries`

For information, refer to  
`PMDADM.<nametag>.MaxRetries` in the PMD  
parameters section.

The default value for the RNFS client daemon is 3.

`PMDADM.nhcrfscIntd.MinPeriod`

For information, refer to  
`PMDADM.<nametag>.MinPeriod` in the PMD  
parameters section.

The default value for the RNFS client daemon is 60.

`PMDADM.nfs.server.ActionScript`

For information, refer to  
`PMDADM.<nametag>.ActionScript` in the PMD  
parameters section.

The default value for the NFS server daemon is:  
`/etc/opt/sun/nhas/init.d/  
default_critical_actionscript.sh`

`PMDADM.nfs.server.ActionScriptDelay`

For information, refer to  
`PMDADM.<nametag>.ActionScriptDelay` in the  
PMD parameters section.

`PMDADM.nfs.server.Criticality`

For information, refer to  
`PMDADM.<nametag>.Criticality` in the PMD  
parameters section.

The default value for the NFS server daemon is  
TRUE.

`PMDADM.nfs.server.MaxRetries`

For information, refer to  
`PMDADM.<nametag>.MaxRetries` in the PMD  
parameters section.

The default value for the NFS server daemon is 2.

`PMDADM.nfs.server.MinPeriod`

For information, refer to  
`PMDADM.<nametag>.MinPeriod` in the PMD  
parameters section.

The default value for the NFS server daemon is 60.

`PMDADM.nfs.client.ActionScript`

For information, refer to  
`PMDADM.<nametag>.ActionScript` in the PMD  
parameters section.

The default value for the NFS client daemon is:  
`/etc/opt/sun/nhas/init.d/  
default_critical_actionscript.sh`

`PMDADM.nfs.client.ActionScriptDelay`

For information, refer to  
`PMDADM.<nametag>.ActionScriptDelay` in the  
 PMD parameters section.

`PMDADM.nfs.client.Criticality`

For information, refer to  
`PMDADM.<nametag>.Criticality` in the PMD  
 parameters section.

The default value for the NFS client daemon is  
 TRUE.

`PMDADM.nfs.client.MaxRetries`

For information, refer to  
`PMDADM.<nametag>.MaxRetries` in the PMD  
 parameters section.

The default value for the NFS client daemon is 2.

`PMDADM.nfs.client.MinPeriod`

For information, refer to  
`PMDADM.<nametag>.MinPeriod` in the PMD  
 parameters section.

The default value for the NFS client daemon is 60.

For more information on the Reliable NFS parameters, see the `nhcrfsd(8)` man  
 page.

**Note** – Some parameters in this file require a single value while others require  
 multiple values.

RELIABLE NFS  
PARAMETERS  
(DRBD  
SPECIFIC)

The following properties apply to only DRBD mode (IP-based replication):

RNFS.Slice

Define the disk partitions that must be managed by Reliable NFS. You must modify this parameter. The format for this parameter is:

RNFS.Slice.x=drbd-resource-id drbd- \ device-name mount-flag

- *x* is an integer (0, 1, 2) that distinguishes each partition to be replicated.
- *drbd-resource-id* is the DRBD resource id as specified in the `/etc/drbd.conf` file. For more information, see the `drbd.conf(5)` man page.
- *drbd-device-name* is the name of the DRBD device.
- *mount-flag* is NoFS, Mandatory, or BestEffort:

NoFS	Do not mount the partition.
Mandatory	Mandatory mount. If the partition is not mountable, stop with error.
BestEffort	Best effort mount. Try to mount the partition. If this is not possible, log an error and continue.

There is no default value.

`RNFS.Slice.2=drbd_r0 /dev/drbd0` Mandatory

**Note** – This property can be modified without fully shutting down the cluster; changes are applied after a successive clean reboot of both MEN nodes.

#### `RNFS.CheckReplicatedSlices`

Check the status of replicated slices by continuously scanning them. This property is disabled by default. To enable this property, set `RNFS.CheckReplicatedSlices` to `True`:  
`RNFS.CheckReplicatedSlices=True`

This check is only performed when the cluster is in the `synchro:READY` state. To determine the synchronization state of the cluster, run the `nhcmmstat` command. If you enable this property and the cluster is not in the `synchro:READY` state the following action occurs:

- An error message is displayed on the master node.
- The cluster is put in the `synchro:NEEDED` state to prevent a switchover occurring. If a switchover occurs, the master might not have access to the most up to date data.

#### `RNFS.CheckReplicatedSlicesInterval`

Set the time between two successive reads of the replicated slices. Values are a number of milliseconds. The default value is 10 milliseconds. The

`RNFS.CheckReplicatedSlicesInterval` property is ignored if the `RNFS.CheckReplicatedSlices` property is not set to `True`.

`RNFS.CheckReplicatedSlicesInterval=time` in milliseconds



**RNFS.EnableSync**

Determine whether disk synchronization is started automatically at startup. Values are `True` and `False`. The default is `True`.

If `RNFS.EnableSync` is set to `True`, disk synchronization is triggered at startup.

If `RNFS.EnableSync` is set to `False`, you delay the start of disk synchronization until you use the `nhenablesync` command. For more information on this command, see the `nhenablesync(8)` man page.

`RNFS.EnableSync=False`

**RNFS.SyncType**

Specify the method used for synchronization between slices on master-eligible nodes. Values are `FS` or `RAW`. The default value is `FS`.

If `RNFS.SyncType` is set to `FS`, this feature is enabled. The time taken for a full slice synchronization is reduced because only the blocks used by the slice's file-system are replicated. If `RNFS.SyncType` is set to `RAW`, this property is disabled.

Both master-eligible nodes must have the same value for the `RNFS.SyncType` parameter. If you change the value of the `RNFS.SyncType` property, reboot the master-eligible nodes one at a time. For information on rebooting cluster nodes, see the *Netra High Availability Suite 3.0 1/08 Foundation Services Cluster Administration Guide*. The `RNFS.SyncType` property is only valid for master-eligible nodes.

`RNFS.SerializeSync`

Determine how disk synchronization is performed. Values are `TRUE` and `FALSE`. The default value is `FALSE`.

Synchronization is necessary following a switchover or the vice-master booting, or when you request a full replication. In these circumstances, if `RNFS.SerializeSync` is set to `FALSE`, Reliable NFS starts the synchronization of all slices at the same time. If `RNFS.SerializeSync` is set to `TRUE`, slices are synchronized one slice at a time. This reduces the network and disk overhead but increases the time it takes for the vice-master to synchronize with the master. During this time, the vice-master is not eligible to take on the role of master.

`RNFS.NFSAlternatePath=/SUNWcgha/local`

`RNFS.Mode=DRBD`

`RNFS.Slice.2=drbd_r0 /dev/drbd0 Mandatory`

`RNFS.Slice.3=drbd_r1 /dev/drbd1 Mandatory`

`RNFS.Share.0=exportfs -i -o`

`rw,no_root_squash,anonuid=0,anongid=0 \`

`cgtp1:/SUNWcgha/local/export cgtp2:/SUNWcgha/local/export`

`master-cgtp:/SUNWcgha/local/export`

`RNFS.SyncType=FS`

`RNFS.CheckReplicatedSlices=False`

## NODE MANAGEMENT AGENT PARAMETERS

`RNFS.EnableSync=True`

`RNFS.SerializeSync=False`

Configure these parameters to specify the external floating address assigned to the master node. The following parameters must be configured when you use the Node Management Agent (NMA).

`PMDADM.nma.ActionScript`

For information, refer to `PMDADM.<nametag>.ActionScript` in the PMD parameters section.

The default value for NMA is:  
`/etc/opt/sun/nhas/init.d/default_noncritical_actionscript.sh`

`PMDADM.nma.ActionScriptDelay`

For information, refer to `PMDADM.<nametag>.ActionScriptDelay` in the PMD parameters section.

`PMDADM.nma.Criticality`

For information, refer to `PMDADM.<nametag>.Criticality` in the PMD parameters section.

The default value for NMA is `FALSE`.

`PMDADM.nma.MaxRetries`

For information, refer to `PMDADM.<nametag>.MaxRetries` in the PMD parameters section.

The default value for NMA is 10.

`PMDADM.nma.MinPeriod`

For information, refer to `PMDADM.<nametag>.MinPeriod` in the PMD parameters section.

The default value for NMA is 60.

**NODE STATE  
MANAGER  
PARAMETERS**

The following parameters must be configured when you use the Node State Manager.

NSM.Exec.MasterDir	<p>The directory containing the scripts for the master node. There is no default value.</p> <p>NSM.Exec.MasterDir=/opt/sun/actions/master</p>
NSM.Exec.ViceMasterDir	<p>The directory containing the scripts for the vice-master node. There is no default value.</p> <p>NSM.Exec.ViceMasterDir=/opt/sun/actions/vicemaster</p>
NSM.Log.MasterDir	<p>The log file directory for the master node. There is no default value.</p> <p>NSM.Log.MasterDir=/var/run/sun/actions/master</p>
NSM.Log.ViceMasterDir	<p>The log file directory for the vice-master node. There is no default value.</p> <p>NSM.Log.ViceMasterDir=/var/run/sun/actions/vicemaster</p>
PMDADM.nhnsmd.ActionScript	<p>For information, refer to PMDADM.&lt;nametag&gt;.ActionScript in the PMD parameters section.</p> <p>The default value for the NSM daemon is /etc/opt/sun/nhas/init.d/default_critical_actionscript.sh</p>
PMDADM.nhnsmd.ActionScriptDelay	<p>For information, refer to PMDADM.&lt;nametag&gt;.ActionScriptDelay in the PMD parameters section.</p>
PMDADM.nhnsmd.Criticality	<p>For information, refer to PMDADM.&lt;nametag&gt;.Criticality in the PMD parameters section.</p> <p>The default value for the NSM daemon is TRUE.</p>

## PROCESS MONITOR DAEMON PARAMETERS

`PMDADM.nhnsmd.MaxRetries`

For information, refer to `PMDADM.<nametag>.MaxRetries` in the PMD parameters section.

The default value for the NSM daemon is 2.

`PMDADM.nhnsmd.MinPeriod`

For information, refer to `PMDADM.<nametag>.MinPeriod` in the PMD parameters section.

The default value for the NSM daemon is 60.

Configure these parameters to specify the external floating address assigned to the master node. These parameters are apply to the entire node. The following parameters must be configured when you use the Process Monitor Daemon (PMD).

`PMDADM.<nametag>.ActionScript`

Specify which action script has to be executed in case `<nametag>` fails. A valid value is a path to an executable or a command line. The default value is a Null string, meaning that no action script is executed. For example:

```
PMDADM.nhcmmd.ActionScript=/etc/opt/
sun/nhas/init.d/
default_critical_actionscript.sh
```

`PMDADM.<nametag>.ActionScriptDelay`

Specify the maximum number of seconds for which the action script may run. When this timeout is reached, the PMD stops current execution of the action script and considers it as "having failed." For example:

```
PMDADM.nhcrfsd.ActionScriptDelay
=5
```

**EXTERNAL  
ADDRESS  
MANAGER  
PARAMETERS**

<code>PMDADM.&lt;nametag&gt;.DaemonStartupDelay</code>	<p>Specify the maximum number of seconds for which the process startup script may run. When this timeout is reached, the PMD stops current execution of the process startup script and considers it as "having failed." For example: <code>PMDADM.nhcrfsd.DaemonStartupDelay=15</code></p>
<code>PMDADM.&lt;nametag&gt;.Criticality</code>	<p>Specify whether the PMD <code>nametag</code>, <code>&lt;nametag&gt;</code>, is critical to the node. Values are <code>TRUE</code> or <code>FALSE</code>. The default value is <code>FALSE</code>. If the daemon under <code>&lt;nametag&gt;</code> is critical and fails, PMD will reboot the node after having executed the action script (if specified), and that action script returned a non-zero value.</p>
<code>PMDADM.&lt;nametag&gt;.MaxRetries</code>	<p>Specify the number of daemon restarts the PMD should attempt for <code>&lt;nametag&gt;</code> before executing the action script, if specified. Valid values are <code>-1</code> (for infinite retries), or a positive number. The default value is <code>0</code>, meaning that PMD will not attempt to restart the daemon.</p>
<code>PMDADM.&lt;nametag&gt;.MinPeriod</code>	<p>Specify the number of minutes for which the PMD should count the <code>&lt;nametag&gt;</code> retries. When PMD has counted retries for the specified number of minutes, the current retry count is reset to zero. Valid values are <code>0</code> (for infinite counting) or a positive number. The default value is <code>0</code>.</p>
	<p>Configure these parameters to specify the external floating address assigned to the master node. These parameters are apply to the entire node. The following parameters must be configured when you use the External Address Manager (EAM).</p>

`EAM.SyncWithRNFS`

Activate synchronization between EAM and RNFS. Activating the synchronization is mandatory if nonpeer nodes acting as NFS clients are accessing the Reliable NFS service, otherwise, NFS clients could experience I/O errors. The format of this parameter is:

`EAM.SyncWithRNFS = True|False`

The default value is `FALSE`

`True`

Synchronization between EAM and RNFS is active, meaning that when a node leaves the Master role, RNFS will wait for EAM to complete before proceeding. When a node re-enters the Master role, EAM will wait for RNFS to complete before proceeding.

`False`

Synchronization between EAM and RNFS is not active, meaning that EAM and RNFS treat CMM notifications concurrently. This improves failover performance by about 500 milliseconds.

`Node.External.FloatingAddress.x`

A list of the external floating IP addresses (IPv4 or IPv6) and slave interfaces. All those addresses will be present on the master node. There is no default value.

`Node.External.FloatingAddress.0=192.168.0.100 eth1`

`Node.External.FloatingAddress.1=fe80:a00:20f7:fa9c:f401 eth1`

`Node.External.Monitor.Group.x`

The list of bonding interfaces to monitor. When one of the monitored groups fails (when all of its interfaces fail), a switchover is triggered. There is no default value.

`Node.External.Monitor.Group.0=bond0`

`EAM.AutoGARP.Period`

Send periodically a gratuitous ARP packet announcing the MAC address associated to the EAM-managed external floating IP addresses. The time period between sends is expressed in seconds, and a value of 0 or less will deactivate the feature. (Gratuitous ARP packets will still be sent whenever an address is set up on any node). The default value is 0 (deactivated).

`PMDADM.nheamd.ActionScript`

For information, refer to `PMDADM.<nametag>.ActionScript` in the PMD parameters section.

The default value for the EAM daemon is `/etc/opt/sun/nhas/init.d/default_critical_actionscript.sh`

`PMDADM.nheamd.ActionScriptDelay`

For information, refer to `PMDADM.<nametag>.ActionScriptDelay` in the PMD parameters section.

`PMDADM.nheamd.Criticality`

For information, refer to `PMDADM.<nametag>.Criticality` in the PMD parameters section.

The default value for the EAM daemon is `TRUE`.



PMDADM.nheamd.MaxRetries

For information, refer to  
PMDADM.<nametag>.MaxRetries in the  
PMD parameters section.

The default value for the EAM daemon is  
2.

PMDADM.nheamd.MinPeriod

For information, refer to  
PMDADM.<nametag>.MinPeriod in the  
PMD parameters section.

The default value for the EAM daemon is  
60.

## PROBE PARAMETERS

Probe.DetectionDelay

The delay in milliseconds before considering a node as down  
when not receiving any heartbeat from that node.

Default value is set to 300 ms. USE WITH CARE.

Probe.HeartbeatNumber

The number of heartbeats to send in the detection delay period.

Default is 3, meaning that if Probe.DetectionDelay is set to  
300 ms, then one heartbeat will be sent every 100 ms (3  
heartbeats are sent in a 300-ms period). USE WITH CARE.

## EXAMPLES

### EXAMPLE 1 Example of the nhfs.conf File on a Master-Eligible Node

```
### Common part
Node.DomainId=69
Node.NIC0=eth0:1
Node.NIC1=eth1
Node.NICCGTP=cgtp0
Node.UseCGTP=True
Node.type=Diskfull
Node.External.FloatingAddress.0=192.168.0.100
Node.External.FloatingAddress.1=10.25.17.1
Node.External.FloatingAddress.2=fe80::a00:20f7:fa9c:f401
Node.External.Monitor.Group.0=group_1
Node.External.Monitor.Group.1=group_2

### Cluster part
Cluster.Master.Id=1

### CMM part
```

```

CMM.IsEligible=True
CMM.LocalConfig.Dir=/etc/opt/sun/nhas

### RNFS part
RNFS.NFSAlternatePath=/SUNWcgha/local
RNFS.Mode=DRBD
RNFS.Slice.2=drbd_r0 /dev/drbd0 Mandatory
RNFS.Slice.3=drbd_r1 /dev/drbd1 Mandatory
RNFS.Share.0=exportfs -i -o rw,no_root_squash,anonuid=0,anongid=0 \
men1-cgtp:/SUNWcgha/local/export men2-cgtp:/SUNWcgha/local/export \
master-cgtp:/SUNWcgha/local/export
RNFS.SyncType=FS
RNFS.CheckReplicatedSlices=False
RNFS.EnableSync=True
RNFS.SerializeSync=False

### NSM Part
NSM.Exec.MasterDir=/opt/sun/actions/master
NSM.Exec.ViceMasterDir=/opt/sun/actions/vicemaster

NSM.Log.MasterDir=/var/run/sun/actions/master
NSM.Log.ViceMasterDir=/var/run/sun/actions/vicemaster

```

**EXAMPLE 2** Example of the `nhfs.conf` File on a Dataless Node

```

Node.NICCGTP=cgtp0
Node.UseCGTP=True
Node.NIC1=eth1
Node.NIC0=eth0
Node.DomainId=250
Node.Type=Dataless

CMM.IsEligible=False
CMM.LocalConfig.Dir=/etc/opt//sun/nhas
CMM.CurrentNodeName=netraDATALESS1

```

**SEE ALSO**

`init(8)`, `nhcrfsd(8)`, `nhcmmd(8)`, `share(8)`, `nhenablesync(8)`, `nhpmd(8)`, `nhnsmd(8)`, `cluster_nodes_table(5)`

NAME	target.conf - local configuration file
SYNOPSIS	<b>/etc/opt/SUNWcgha/target.conf</b>
DESCRIPTION	<p>The local configuration file, <code>target.conf</code>, contains the cluster ID, attributes, and election roles for each node in the cluster. For example, the <code>target.conf</code> file will specify whether the node is a master-eligible node. The <code>target.conf</code> file is located on each node on the cluster and contains a description of the node on which it is located.</p>
WARNINGS	<p>Modify this file only when you are manually fixing a problem on the cluster as described in the <code>cluster_nodes_table(5)</code> man page.</p>
EXTENDED DESCRIPTION	<p>The following is an example of a <code>target.conf</code> file:</p> <pre>VERSION : 2           # Version number domain_id: 128        # Cluster ID attributes: -         # Local nodes attributes election: 5           # Election round number role: MASTER          # Role</pre> <p>For an explanation of the fields in the example, see the <code>cluster_nodes_table(5)</code> man page.</p>
SEE ALSO	<code>nhcmmmd(8)</code> , <code>cluster_nodes_table(5)</code> , <code>nhadm(8)</code> , <code>nhfs.conf(5)</code>



<b>NAME</b>	cgtp - CGTP virtual device driver
<b>SYNOPSIS</b>	<b>modprobe cgtp</b>
<b>DESCRIPTION</b>	<p>The Carrier Grade Transport Protocol (CGTP) is used for reliable transport between nodes over dual redundant network links.</p> <p>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.</p> <p>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.</p> <p>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.</p> <p>If one link fails, the data packet sent on the other network path is still able to reach the destination address. However, until the failed network link is repaired, the system is not highly available.</p> <p>CGTP is implemented by kernel module cgtp.ko. It comes preinstalled with the Netra HA Suite Foundation Services or it can be installed standalone. CGTP filtering on the reciever side is implemented by the kernel module ipt_cgtp.ko. To use CGTP, these kernel modules must be loaded. Then, a couple of Ethernet interfaces must be assigned to cgtp0 using <code>/opt/sun/sbin/cgtpool</code>. CGTP filtering is turned on using <code>/sbin/iptables</code>.</p> <p>CGTP never actually sends or receives data. Instead, the CGTP configuration must be done at the same time as the configuration of no more than two physical interfaces of any type that are responsible for redundancy. When the cgtp0 virtual physical interface and the two physical interfaces are properly configured, data will be sent and received through the redundant interfaces. At the emitter, cgtp is used to specify the source address of the CGTP IP packets. Packets are then transmitted on the physical interfaces. At the receiver, incoming CGTP packets are received by redundant interfaces, the duplicates are filtered out at the IP level, and the remaining packets are presented to upper applications (after an optional reassembly).</p> <p>For example:</p>

1. Load the CGTP kernel module:

```
# modprobe cgtp
```

2. Load CGTP filtering kernel module:

```
# modprobe ipt_cgtp
```

3. Assign Ethernet interfaces to the CGTP virtual interface:

```
# /opt/sun/sbin/cgtpool cgtp0 eth0 eth1
```

4. Configure and start CGTP filtering (assuming 10.191.3.0 is CGTP network address):

```
# /sbin/iptables -A PREROUTING -t mangle -m cgtp --ipsrc 10.191.3.0 -j DROP
```

5. Configure the CGTP IP address:

```
# ifconfig cgtp0 10.191.3.10 netmask 255.255.255.0 broadcast 10.191.3.0 up
```

For detailed instructions on how to manually install and configure CGTP on standalone nodes, see the *Netra High Availability Suite 3.0 1/08 Foundation Services Standalone CGTP Guide*.

**Note** – Ensure that return path filtering is turned off, otherwise CGTP will not function properly (echo 0 > /proc/sys/net/ipv4/conf/all/rp\_filter).

## FILES

/proc/net/cgtp0/gateway - The gateway table.

This table contains the IP addresses of the gateways that should be used to resolve the MAC addresses of the CGTP destination that are not directly resolvable (for example, when installing a heterogeneous cluster with both Linux and Solaris nodes). The replicated CGTP traffic is then sent to the MAC addresses of these gateways.

To add an entry to the redundant gateways table, use the echo command as follows:

```
# echo "add <dest IP addr> <gateway IP addr> <slave interface>" \
    /proc/net/cgtp0/gateway
```

To remove an entry from the redundant gateways table, use the echo command as follows:

```
# echo "del <dest IP addr> <gateway IP addr> <slave interface>" \
    /proc/net/cgtp0/gateway
```

To change the gateway IP address of an existing entry in the redundant gateways table, use the echo

```
# echo "change <dest IP addr> <new gateway IP addr> <slave interface>" \
    /proc/net/cgtp0/gateway
```

For example, if a CGTP IP address 11.0.3.15 can only be reached through two gateways 11.0.1.15 (through the eth0 interface, connected to the first redundant network) and 11.0.2.15 (through eth1, connected to the second network), then the following entries should be added to the redundant gateways table:

```
# echo "add 11.0.3.15 11.0.1.15 eth0" >/proc/net/cgtp0/gateway
# echo "add 11.0.3.15 11.0.2.15 eth1" >/proc/net/cgtp0/gateway
```

And the result appears as:

```
# cat /proc/net/cgtp0/gateway
DeviceDest. IP addrGateway IP addr
-----
eth011.0.3.1511.0.1.15
eth111.0.3.1511.0.2.15
```

**SEE ALSO** [iptables\(8\)](#)

