HPC

BAS5 for Xeon

High Availability Guide

REFERENCE
86 A2 21FA 00

HPC

BAS5 for Xeon

High Availability Guide

Software

October 2008

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

REFERENCE
86 A2 21FA 00

Trademarks and Acknowledgements

The information in this document is subject to change without notice. Bull will not be liable for errors contained herein, or for incidental or consequential damages in connection with the use of this material.

Preface

Scope and Objectives

High Availability is used to ensure the continuous running of the cluster when there is a system component failure or a downtime event. This guide describes the configuration and management of the different forms of High Availability that apply to Bull HPC clusters.

Important

Refer to the Software Release Bulletin (SRB), delivered with your system, for details of the High Availability functions and features that apply to your Bull Advanced Server (BAS) delivery.

Please pay particular attention to any restrictions which may be in place for your delivery.

Intended Readers

This guide is for Administrators of Bull HPC systems.

Structure

This manual is organised as follows:

Chapter 1. *Cluster High Availability – an Introduction*
Details the main principles of High Availability and some general concepts that apply to Bull clusters.

Chapter 2. *Management Node High Availability*
Explains how to implement High Availability for the Management Node using Cluster Suite and additional HA scripts.

Chapter 3. *Configuring High Availability for the Management Node*
Describes how to install the Primary and Secondary Management Nodes for High Availability.

Chapter 4. *Managing High Availability on the Management Node*
Describes the Cluster Suite commands used to manage High Availability on the Management Node.

Chapter 5. *Lustre File System High Availability for I/O Nodes*
Describes the hardware architecture for Lustre High Availability and various failure modes which may apply.

Chapter 6. *Configuring High Availability for Lustre*
Explains how to implement High Availability for I/O Nodes which use the **Lustre** file system.

Chapter 7. *Lustre and Management Node High Availability*
Describes how to configure the Lustre services which are managed from a Highly Available Management Node

Chapter 8. *NFSv3 High Availability for I/O Nodes*
Describes the hardware architecture for NFSv3High Availability and the various failure modes which may apply.

Chapter 9. *Configuring High Availability for NFSv3*
Explains how to implement High Availability for I/O Nodes which use the **NFSv3** file system.

Chapter 10. *High Availability for Resources and Jobs*
Explains how to configure High Availability for the **PBS Professional**, **LSF** and **RMS** resource managers.

Bibliography

Refer to the manuals included on the documentation CD delivered with you system OR download the latest manuals for your Bull Advanced Server **(BAS)** release, and for your cluster hardware, from: http://support.bull.com/

The Bull *BAS5 for Xeon Documentation* CD-ROM (86 A2 91EW) includes the following manuals:

- Bull *HPC BAS5 for Xeon Installation and Configuration Guide* (86 A2 87EW).
- Bull *HPC BAS5 for Xeon Administrator's Guide* (86 A2 88EW).
- Bull *HPC BAS5 for Xeon User's Guide* (86 A2 89EW).
- Bull *HPC BAS5 for Xeon Maintenance Guide* (86 A2 90EW).
- Bull *HPC BAS5 for Xeon Application Tuning Guide* (86 A2 16FA).
- Bull *HPC BAS5 for Xeon High Availability Guide* (86 A2 21FA).

The following document is delivered separately:

- The *Software Release Bulletin* (SRB) (86 A2 71 EJ)

Important

The Software Release Bulletin contains the latest information for your BAS delivery. This should be read first. Contact your support representative for more information.

In addition, refer to the following:

- Bull *Voltaire Switches Documentation CD* (86 A2 79ET)
- *NovaScale Master* documentation

For clusters which use the **PBS Professional** Batch Manager:

- PBS Professional 9.2 *Administrator's Guide (*on the *PBS Professional CD-ROM)*
- PBS Professional 9.2 *User's Guide* (on the *PBS Professional CD-ROM*)

Highlighting

- Commands entered by the user are in a frame in '`Courier`' font, as shown below:

```
mkdir /var/lib/newdir
```

- System messages displayed on the screen are in '`Courier New`' font between 2 dotted lines, as shown below.

```
Enter the number for the path :
```

- Values to be entered in by the user are in '`Courier New`', for example:

  `COM1`

- Commands, files, directories and other items whose names are predefined by the system are in '**Bold**', as shown below:

  The **/etc/sysconfig/**dump file.

- The use of *Italics* identifies publications, chapters, sections, figures, and tables that are referenced.

- < > identifies parameters to be supplied by the user, for example:

  `<node_name>`

WARNING
A Warning notice indicates an action that could cause damage to a program, device, system, or data.

Table of Contents

List of Figures

Chapter 1. Cluster High Availability - an Introduction

Important

Different types of High Availability are supported for different Bull Advanced Server (BAS) system releases.

See the System Release Bulletin for the BAS version delivered with your cluster for details of the High Availability domains and features that apply to your system.

By High Availability we mean a protocol, and its associated execution, which ensures operational continuity when there is a downtime event for a particular part of the cluster. High Availability ensures that a cluster is protected against any potential single points of failure (SPOF) that may exist. A continuously available cluster system is characterized as one with minimum downtime in any given year. The best High Availability rate is known as 99.999% availability.

Highly available clusters operate by having redundant nodes to take over a service, if and when a system component fails. Hardware and software faults are detected instantly, and the application is immediately switched to the redundant node with no lost of processing time, and with no direct intervention from the Administrator. This process is known as failover.

The implementation and configuration of High Availability focuses on system backup, failover processing, data storage, and access routines for the various components of the cluster. This manual describes the various mechanisms that are available for Bull HPC clusters. A prime requirement for these mechanisms is to ensure that there is no data corruption when failover occurs.

1.1 High Availability Domains

The architectural requirements, the software configuration operations, and the management of High Availability is described in this manual for the different elements of Bull HPC clusters to which High Availability may apply.
The following domains are described:

1.1.1 High Availability for the Management Node

High Availability for the Management Node uses an **active/passive** node configuration. That is to say there exists a fully redundant secondary node which only comes into operation when there is a failure on the primary node.

- See *Chapter 2* for a description of possible Management Node failures, High Availability architectural requirements, and the Management Node HA scripts that are provided.
- See *Chapter 3* for details on how to configure High Availability on the Management Node.
- See *Chapter 4* for details on managing High Availability on the Management Node.

1.1.2 High Availability for the File System

Lustre

Lustre uses an **active/active** node configuration. This means that all the I/O nodes are active and if a node fails the other nodes will pick up its load.

- See *Chapter 5* for a description of Lustre failover mechanisms, the hardware architecture required, and an analysis of the different possible failures that are covered by High Availability for Lustre.

- See *Chapter 6* for details on how to configure High Availability for Lustre.

- See *Chapter 7* for details on managing High Availability for Lustre including the additional configuration steps that are necessary for clusters which also include High Availability for the Management Node.

NFSv3

NFSv3 uses an **active/passive** node configuration for the NFS nodes.

- See *Chapter 8* for description of the hardware architecture required, and the different possible failures that are covered by High Availability for NFSv3.

- See *Chapter 9* for details on how to configure and monitor High Availability for NFSv3.

1.1.3 High Availability for Resources and Jobs

- See *Chapter 10* for description of High Availability for the different resource and job managers that exist for Bull HPC clusters, including **PBS Professional**, **LSF** and **Quadrics RMS**.

Chapter 2. Management Node High Availability

Important

Management Node High Availability is not supported for all Bull Advanced Server (BAS) system releases.

See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

This chapter explains how to implement Management Node High Availability using **Cluster Suite** and specific HA scripts. The following topics are described:

- 2.1 *Introduction*
- 2.2 *Architecture*
- 2.3 *Understanding High Availability Scripts for the Management Node*
- 2.4 *Understanding Channel Bonding (Optional)*

2.1 Introduction

One of the most fragile points of the cluster is the Management Node. All management services for a cluster running on the Management Node will become unavailable if the Management Node goes down. The management of a compute cluster job will run directly from the Management Node. Therefore, if the Management Node goes down it will no longer be possible to submit any compute jobs, making all the nodes unusable.

In order to avoid such a lost of functionality, it is necessary to physically double up the Management Nodes, and to put a recovery mechanism into place for the cluster management services on the backup node.

When a problem occurs on one of the Management Nodes, the management applications running on that node rock over to the second one. High Availability of the cluster management is based on an architecture which includes two Management Nodes and shared disk space.

Important

When there is a switch of Management Node, the High Availability of the application program running on the cluster is not guaranteed. The High Availability for the application program can be ensured only if the application program includes a means of continuing a task(s) which was/were in progress when the sudden interruption, and subsequent switch, occurred.

For the Management Nodes that are made Highly Available, the node that is actually running the critical services is designated as the **Primary Node**. The node which is ready to take the place of the Primary Node should the Primary Node go down is designated as the **Back-up Node**.

The Primary node sends signals (**heart-beats**) through the network. If a software or hardware breakdown occurs, the High Availability software takes the actions necessary to maintain the availability of the services, and the integrity of the data, whilst rocking (failover) over to the Back-up node in a transparent way.

When the backup Back-up node takes charge, the High Availability software ensures that the first Primary node "does not awake" suddenly, forcing the backup Back-up node to shutdown as a result of a power switch. When the backup Back-up node is certain that the failing Primary node is down, it will take charge of the services from this node.

It is possible that the node which has failed may retake charge of its functions after a reboot, or an intervention of an operator. This High Availability feature makes it possible for an operator to rock the services to the backup Back-up node (service of relocating) in order to intervene at the time of the maintenance actions.

The High Availability management tool is **Cluster Suite.** A heart-beat mechanism detects if the Management Nodes are running. If the Primary node is found to be unattainable then a **fencing** mechanism is activated, this powers off the failing node. Then the **Back-up node** becomes the **Primary node**.

Within an HPC Cluster the different constituent elements, for example Compute Nodes, I/O nodes, network equipment, and so on, have to address a **virtual Management Node.** This virtual Management Node is in reality the Primary node. This virtualization of the Management Node is implemented by the mechanisms described within this section.

The elements that allow the administrator to ensure that High Availability is in place on the Management Nodes are described in this chapter.

Note The concept of High Availability applies to a range of Management Node functionalities. This chapter is only concerned with protecting critical services using High Availability.

2.1.1 Possible Management Node failures

Management Node kernel panic/hang

When a node hangs or encounters a panic, it does not send its heartbeat messages within the authorized period. This silence is detected by the High Availability peer node which fences the silent node, and takes over the cluster services when the fencing is completed.

Node Hardware Failure or Power Down

The node does not send its heartbeat messages in the authorized period. This silence is detected by the High Availability peer node, which fences the silent node and takes over the cluster services when the fencing is completed.

2.2 Architecture

2.2.1 Hardware

Management Node services must be rapidly made operational again, when a Management Node goes down. For this to happen, the hardware elements must be doubled up. In order to avoid data synchronization problems between the 2 Management Nodes, a rack of disks is shared between them. This rack is designed to be tolerant of breakdowns and ruptures.

At the Ethernet connection level, there are two network interfaces, which are seen as single interface. The electric power supply of the two Management Nodes may be controlled remotely using the Ethernet network.

Typical hardware structures are shown below.

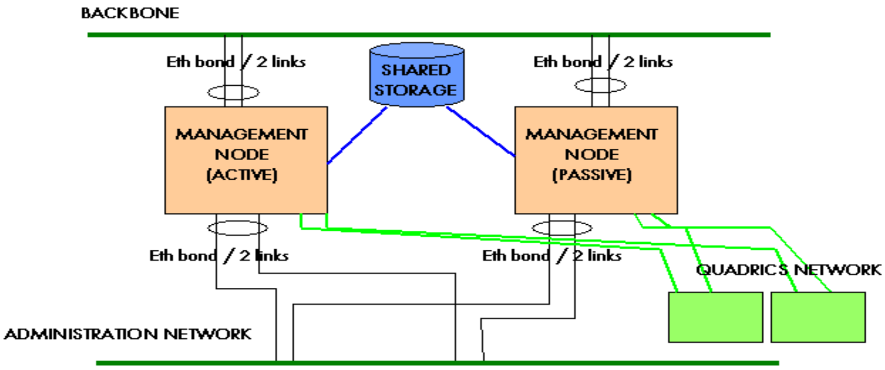


Figure 2-1. High Availability Management Node configuration

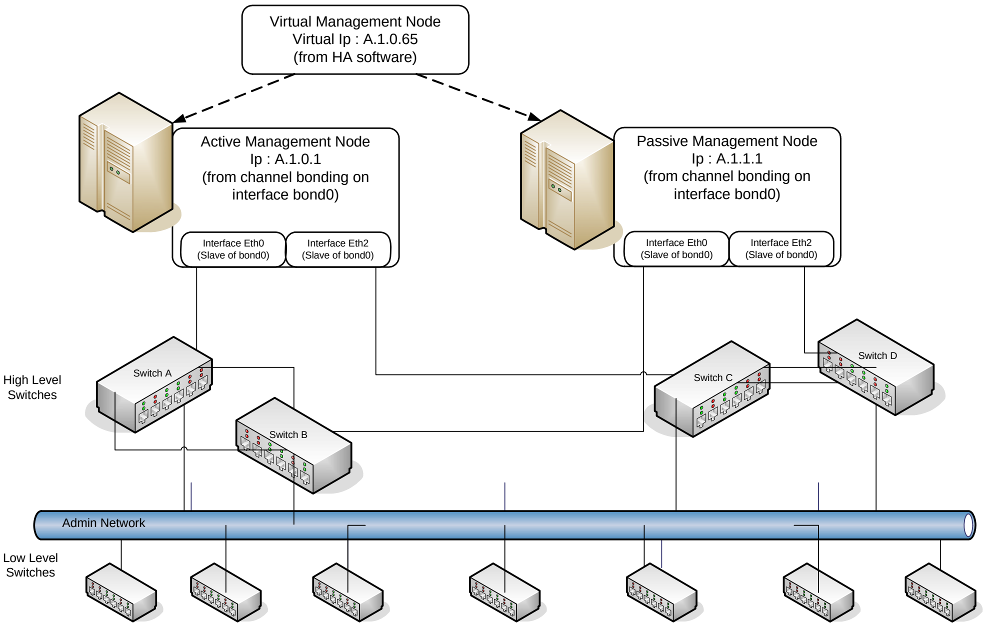


Figure 2-2. High Availability management network configuration

Each service will run on only one of the Management Nodes, which are contained within the High Availability system, at a time.

In order to distinguish the Management Nodes, one is called the Primary node and the other is called the Secondary node. The one on which the services are running will be in active mode, and the other one (this may also be known as the backup node) will be in passive mode, ready to step in if the active node breaks down.

The primary node is the node that appears first alphabetically.

The primary node may also be defined as follows:

- The only Management Node available on the cluster that makes it possible to meet all the management functional needs.
- The node on which modifications are made, these are then propagated or replicated on the other Management Nodes.

2.2.2 Software

High Availability is implemented using Red Hat **Cluster Suite**. Therefore, Cluster Suite has to be configured, and the system adapted, so that it runs correctly.

A Linux distribution has to be installed on all the Management Nodes, as well as all the applications needed for smooth operation of the cluster.

In addition, scripts have been developed to enable **Cluster Suite** to implement High Availability on the Management Nodes. These scripts manage the High Availability of some services, by for example starting, stopping or changing the state of a service.

Important

Some services include the High Availability functionality directly, for example SLURM, others allow a synchronization mechanism to run. These services are not covered by this section of the manual.

For each service that is to be made highly available it will be necessary to take an inventory of:

- Which files and directories have to be present on both the active and passive nodes (**/etc/hosts**, **/etc/sshd**, etc.) and that can be modified for configuration purpose, although, this is seldom necessary.
- Which files and directories may be subject to change and so difficult to synchronize. This data should be placed on the shared discs that are common to both Management Nodes (**/etc/httpd**, etc.)
- Which files and directories do not change (**/usr/sbin/httpd**, etc.) and so do not need any special administration.

Example of how to configure High Availability for a HTTP server

Prerequisites:

- Two nodes with separate IP addresses.
- Both nodes are exactly the same in functional terms, both from a software and system point of view. There may be differences for the MAC addresses and other hardware references.
- Each node has a *power on/power off* mechanism.
- A virtual IP address exists.
- All the HTTP server files are in the **/var/www** directory.
- The HTTP service is managed by **/etc/init.d/httpd**.
- The **/dev/sdb** device (visible from both nodes) corresponds to a storage rack shared between both nodes, and the partition **/dev/sdb1** is free and large enough to contain **/var/www**.

Implementing High Availability for the HTTP service consists of:

- Stopping the http service on both nodes and removing it from the boot start up.
- Initializing the **/dev/sdb1** partition, and then recopying the contents of **/var/www** to the **/dev/sdb1** partition.

To implement High Availability on the Management Nodes of a cluster, a list of all the services to be made highly available should be drawn up. As will be seen later, it has been decided that **Cluster Suite** will manage only one service, **haservices** which manages all the High Availability elements and services.

See The Red Hat **Cluster Suite** documentation for detailed information.

2.2.3 Network

The applications running on the compute nodes have to communicate with the management applications. Most of the time client applications communicate with a host which is identified by its IP address. However, when there is a failover (the existing active node stops being the active one and the passive one becomes active) the new active node must be as accessible as the previous one. Therefore, the IP address has to follow the active node. Nevertheless, both Management Nodes must have a proper IP address so that they can be reachable separately.

Accordingly, each Management Node has an IP address to interface with the management network (for example, 10.0.0.1/24 for the primary node, and 10.0.0.2/24 for the secondary node). In effect, when the High Availability service is active on the primary node the interface with the Management Node is a virtual IP address (for example, 10.0.0.250/24), virtual in the sense that it does not identify the primary node but the node which is active. This is more a functional resource than a physical one.

When the Active Node is the Primary Node

- The management interface with the primary node has two IP addresses (10.0.0.1/24 and 10.0.0.250/24).
- The management interface with the secondary node has one IP address (10.0.0.2/24).
- All cluster components can connect with the primary node using its real IP address (10.0.0.1/24) and its virtual IP address (10.0.0.250/24).
- The secondary node can only be accessed by using its real IP address (10.0.0.2/24).
- The active node is accessed using the virtual IP address (10.0.0.250/24).

When the Active Node is the Secondary Node

- The management interface with the primary node has one IP address (10.0.0.1/24).
- The management interface with the secondary node has two IP addresses (10.0.0.2/24 and 10.0.0.250/24).
- The primary node can only be accessed by using its real IP address (10.0.0.1/24).
- All cluster components can connect with the secondary node using its real IP address (10.0.0.2/24) and its virtual IP address (10.0.0.250/24).
- The active node is accessed using the virtual IP address (10.0.0.250/24).

2.3 Understanding High Availability Scripts for the Management Node

The **/etc/HA** directory contains all the tools that enable High Availability to be implemented and which are also used to manage the critical services which have been made highly available. It also contains the configuration files for these High Availability tools.

When Cluster Suite is configured, the **/etc/HA/bin/haservices** script is defined as the main service. This script is the entrance point for implementing High Availability for the cluster management services.

All the High Availability scripts used on the Management Node are in the **/etc/HA/bin** directory. The list of scripts is as follows:

cs Starts all the Cluster Suite services.

dbmConfig Updates files according to the ClusterDB status.

haadminnodes Lists the Management Nodes contained in the database.

hacisco Removes the ARP tables for the switches included in the database.

hacron De-activates/activates the crons specified in the **/etc/cron.d** file.

hadown Simulates a script which always returns the same result.

hafake Simulates a service.

haip Handles (Manages) the virtual IP addresses associated with the active node.

halinks Manages the dynamic links for the directories which have to be on a shared storage system.

hamount Manages the mount points of the shared storage system.

hapidcleaner Removes the **.pid** file which may prevent some services from starting.

hapostfix Manages the postfix mailing service.

harpms Checks that the list of installed RPMs is identical on each Management Node.

haservices Main script which controls all the HA services.

hasynchro Manages the synchronization of files between the Management Nodes.

hasyslogng Manages **syslog-ng**.

hatstfs Tests the writing capability for different systems of shared files.

haunwantedfiles Manages the services that must not be launched when a node is started.

haupdatedb Updates the database according to the active node.

Some of these scripts are configurable using the files in the **/etc/HA** folder.

2.3.1 /etc/HA/bin/haservices script

This script has to be used within Cluster Suite.

According to the parameter (**start**, **status** or **stop**), a range of commands, specified in the **/etc/HA/start**, **/etc/HA/stop** or **/etc/HA/status** data files will be executed.

Note Three different data files are necessary because the commands to start, stop or report the status are not the same, or are not executed in the same order.

Each file consists of a series of lines that include:

- Lines beginning with a hash symbol (#). These are comments.
- Empty lines. These are ignored.
- Lines that indicate a command and consist of the full path without any options or spaces.

Note Some of the example data files delivered in the BAS5 for Xeon rpm will be different from the examples shown in the next three sections.

2.3.2 /etc/HA/start file

This file contains the list of the commands to be carried out when the **haservices** script is launched is run with the **start** parameter. The commands are carried out in the order listed in the **start** file. When the first error occurs, **haservices** displays an error message.

/etc/HA/start example file

```
# What to do at starting time and in which order ?
# Programs to call with 'start' parameter

/etc/HA/bin/haunwantedfiles
/etc/HA/bin/hamount
/etc/HA/bin/halinks
/etc/HA/bin/hapidcleaner

/etc/init.d/postgresql
/etc/HA/bin/haip
# the following line mostly starts nagios
/etc/HA/bin/dbmconfig

/etc/HA/bin/hasyslogng

# for managing Cisco switches
/etc/HA/bin/hacisco
```

```
/etc/init.d/snmptrapd
/etc/init.d/gmond
/etc/init.d/gmetad
/etc/init.d/dhcpd
/etc/init.d/ldap
/etc/init.d/httpd
/etc/init.d/conman
/etc/init.d/nfs
/etc/HA/bin/hasystemimager

# At this step you should customize according to :
# When using RMS Quadrics uncomment the next two lines
#/etc/init.d/msqld
#/etc/HA/bin/harms


# At this step you should customize according to :
# When using LSF/SLURM uncomment the next line
#/etc/init.d/lsf

# If you are using cyrus/imap uncomment the next two lines
#/etc/init.d/saslauthd
#/etc/init.d/cyrus-imapd
/etc/init.d/postfix

/etc/HA/bin/hacron

#for management FDA storage
/etc/HA/bin/hafda
```

2.3.3 /etc/HA/status file

This file must be empty because there is not yet monitoring of status.

2.3.4 /etc/HA/stop file

This file contains the list of the commands to be carried out when the **haservices** script is launched with the **stop** parameter. The commands are carried out in the order given in the **stop** file. When the first error occurs, **haservices** displays an error message.

The starting order of the commands is important. In the **stop** file the commands are usually executed in reverse order to that of the start file.

/etc/HA/stop example file

```
# What to do at stoping time and in which order ?
# Programs to call with 'stop' parameter

/etc/HA/bin/halsf

/etc/HA/bin/harms
/etc/init.d/msqld
/etc/init.d/qsnet

#/etc/HA/bin/hapostfix
/etc/init.d/saslauthd
/etc/init.d/cyrus-imapd

/etc/init.d/systemimager
/etc/init.d/nfs
```

```
/etc/init.d/conman
/etc/init.d/httpd
/etc/init.d/ldap
/etc/init.d/dhcpd
/etc/init.d/gmetad
/etc/init.d/gmond
/etc/init.d/nagios
/etc/init.d/snmptrapd

#/etc/init.d/krb5kdc
#/etc/init.d/kadmin

/etc/init.d/postgresql

/etc/HA/bin/hasyslogng
/etc/HA/bin/hamount
/etc/HA/bin/haip
```

2.3.5 hacron script

In the case of a call with the **start** option, for each file listed in the **/etc/HA/cron** directory, this script will move the corresponding file from the **/etc/cron.d** directory to the **/etc/cron.d/SLEEP** directory in order to deactivate the cron.

In the case of a call with the **stop** option, for each file listed in **/etc/HA/cron**, this script will move the corresponding file from the **/etc/cron.d/SLEE**P directory to the **/etc/cron.d** directory in order to re-activate this cron.

The **/etc/HA/cron** file ignores all empty lines; all comment lines starting with # are ignored. All other lines are considered as the names of files that are usually included in **/etc/cron.d**.

2.3.6 haip script

This script manages the virtual IP addresses.

In the case of a call with the **start** argument, for each relevant line in the **/etc/HA/ip** data file, the script will add the IP address specified to the interface indicated. Also, an **ARP** update is sent out in order to notify all hardware components linked to this local network that the virtual IP address is now associated with the network interface specified.

In the case of a call with the **stop** argument, for each relevant line in the **/etc/HA/ip** directory, the script withdraws the specified virtual IP address from the network interface indicated.

In the case of a call with the **status** argument, for each relevant line in the **/etc/HA/ip** directory, the script checks that the specified virtual IP address is associated with the network interface indicated. When the first error occurs, an integer different from zero is returned.

The **/etc/HA/ip** file can handle empty lines; all comment lines starting with # are ignored. All other lines are considered as specifying a network interface and a virtual IP address in the following format:

- tabulation
- name of the interface, for example eth0, eth2, bond0
- tabulation
- IP address:
  - 4 integers separated with a dot, for example : 10.0.0.50
  - a slash
  - an integer (netmask).

Example of 2 virtual addresses for 2 network interfaces

```
bond0  10.0.0.50/8
eth1 192.168.255.1/16
```

2.3.7 halinks script

This script makes it possible to manage the symbolic links for the directories that have to be included in the shared storage system. The corresponding data file is **/etc/HA/links**. Any blank line or one starting with # is ignored. Any other line is considered as specifying the complete path of a directory.

In the case of a call with the **start** or **status** argument, the script checks that each path specified in the **/etc/HA/links** file is a symbolic link. For the first error met, an error code different from zero is returned.

In the case of a call with the **repair** argument, the script checks that each path specified in the **/etc/HA/links** file is a symbolic link. If it is not the case, it renames the directory by suffixing the name with **.orig-<date>**. Then, the link is created pointing towards what should be on the shared storage system. For example, for a path starting with **var**, the link points towards a similar path in which **var** is replaced with **varha** (for example **/var/lib/pgsql** replaced with **/varha/lib/pgsql**). If the shared filesystem is mounted and the shared directory indicated by the symbolic link does not exist, then the script copies the original directory into the directory indicated by the symbolic link.

Important

On the primary Management Node of the cluster, it is advisable to launch this script after the shared file systems have been mounted with the /etc/HA/bin/hamount start command. To avoid any corruption problems, do not perform this operation while Cluster Suite is running.
On the secondary Management Node, it is advisable not to have mounted the shared filesystems and not to have Cluster Suite running.

In the case of a call with the **undo** argument, the script checks that each path specified in the **/etc/HA/links** file corresponds to a symbolic link. If it does, it deletes the link, and then renames it as the most recent directory **<link_name>.orig-<date>** in **<link_name>**. **To avoid any corruption problems, do not perform this operation while Cluster Suite is running.**

/etc/HA/links file example:

```
# directory placed on shared disk

/etc/httpd
/etc/nagios
/etc/openldap
/etc/snmp
/etc/systemimager

/usr/lib/clustmngt/rms/etc
/var/log/postgres

/usr/lib/nagios
/usr/share/nagios
/usr/lib/rms/msql
/var/lib/ganglia
/var/lib/ldap
/var/lib/pgsql
/var/lib/systemimager
/var/lib/imap
/var/log/conman
/var/log/httpd

/var/log/systemimager
/var/spool/nagios
/var/spool/imap
/var/www
/var/rms

/usr/share/lsf
```

2.3.8 hamount script

This script manages the mount/un-mount operations for filesystems, usually, those which are included in shared storage systems.

Important

When Cluster Suite is running, do not run this script manually on any of the Management Nodes of the cluster in order to avoid data corruption.

The data file of this script is **/etc/HA/mnttab**. A valid line contains the following:

- A device name, beginning with **/dev/**:
  `/dev/sda, /dev/sda1, /dev/hdb, /dev/hdb3, /dev/sdm …`
- One or more spaces.
- A directory path indicating the mount point for the specified device.

Note Do not use space or tab characters in the device and path names. The following regular expressions are valid -**^(/dev/.*?)\s+(.*?)\s*$** -:

Examples of valid lines:

```
/dev/sdj1        /etcha
/dev/sdj2        /usrha
```

```
/dev/sdj3        /tftpboot
/dev/sdj4        /homeha
/dev/sdp1        /varha
```

In the case of a call with the **start** argument, for each specified mount point, the script unmounts and mounts the filesystem.

In the case of a call with the **stop** argument, for each specified mount point, the script unmounts the filesystem.

In the case of a call with the **status** argument, for each mount point, the script checks that the filesystem is correctly mounted.

In order that the administrator properly understands the consequences of a call to **hamount** with the **stop** argument, the filesystem unmounting operations are indicated below:

- If the filesystem is easily unmounted, then the operation is successful.
- If not, the script searches for all the processes which are included in the tree structure, and sends a SIGSTOP to them.
- If necessary, the above operation will be repeated several times.
- If this does not work then a SIGKILL is sent to all processes included in the tree structure.
- If necessary the above operation will be repeated several times.
- Finally, as a last measure, a forced unmount is run (lazy umount).

Note A forced **unmount** may not be stable.

2.3.9 hapidcleaner script

This script is used to clean the **pid** files which could disturb the launching of services.

Note A **pid** file is a file containing in the first line an integer referring to a process.

The data file is **/etc/HA/pid2clean**. All blank lines or those starting with # are ignored. All other lines are considered as specifying the complete path for a pid file.

In the case of a call with the **start** argument, for each path specified in the **/etc/HA/pid2clean** file, the script checks if the corresponding process is in the course of being executed. If not, then the script deletes the file.

/etc/HA/pid2clean example file

```
# One filename per line. Files to remove at CS startup
/var/lib/pgsql/data/postmaster.pid
```

2.3.10 hasynchro script

Note This script can only be run from the active node and has to be launched manually as it will not start automatically.

This script enables the management of files that can only be synchronized manually. These files are rarely modified.

The corresponding data file is **/etc/HA/synchro**. All blank lines or those starting with # are ignored. All other lines are considered as specifying the complete path for a file.

In the case of a call with the **start** or **status** arguments, the script connects to the passive node checks each file specified in the **/etc/HA/synchro** file and returns a status different from zero when the first different file is encountered.

In the case of a call with the **repair** argument, the script connects to the passive nodes, and for each file different from that of the active node, it replaces the passive node file with the active node version.

/etc/HA/synchro file example:

```
# Which files or to be synchronized ?
# One fullpath per line

/etc/HA/ip
/etc/HA/links
/etc/HA/mnttab
/etc/HA/pid2clean
/etc/HA/start
/etc/HA/status
/etc/HA/stop
/etc/HA/synchro


/etc/conman.conf
/etc/conman-tpl.conf
#/etc/dhcpd.conf
/etc/dhcpd-tpl.conf
/etc/exports
/etc/genders
/etc/gmetad.conf
/etc/gmond.conf
#/etc/hosts
/etc/hosts-tpl
/etc/resolv.conf
/etc/ssh/ssh_config
/etc/ssh/ssh_host_dsa_key
/etc/ssh/ssh_host_dsa_key.pub
/etc/ssh/ssh_host_key
/etc/ssh/ssh_host_key.pub
/etc/ssh/ssh_host_rsa_key
/etc/ssh/ssh_host_rsa_key.pub
/etc/storageadmin/ddn_admin.conf
/etc/storageadmin/nec_admin.conf
/etc/storageadmin/storframework.conf
/etc/sudoers
/etc/syslog-ng/syslog-ng.conf
```

2.3.11 hasyslogng script

The logs generated by the **hasyslogng** script running on the active node are saved in the **/varha/log** folder.

It should be noted that when the Management Node starts and is not in active mode **syslog-ng** saves its logs in the **/var/log** folder. This means that the Administrator has to be aware of the existence of these different folders (**/varha/log** and **/var/log**) and of the Management Node mode in order to identify the location of the logs.

Further, some of the logs may be saved on the common storage sub systems using the symbolic link mechanism which is put into place using the **halinks** command. The decision to save the logs on the storage sub system is made on a case by case basis. Again the Administrator has to keep in mind that the **syslog-ng** logs may be stored in different places according to settings made on the Management Node and its mode (active or passive).

2.3.12 haunwantedfiles script

This script manages any problems for the services which are executed when the Management Node starts, and which are intended to be managed by Cluster Suite. For example, if it is decided that the **httpd** server should be highly available, then its data files must be on the shared storage system. In order to avoid data corruption problems on the shared storage system the httpd server should only run on the active node, and the service should not be launched when the Management Node starts.

This script is based on the configuration files of **/etc/HA/bin/haservices** (/etc/HA/start, /etc/HA/stop and /etc/HA/status).

In the case of a call with the **start** or **status** parameters, the **haunwanted** script checks that no existing links in the system start directories points towards a service managed by **haservices**.

In the case of a call with the **report** parameter, for each service managed by **haservices,** the script will check in each start directory **(/etc/rc*.d)** for the presence of a link pointing to the service. If a link exists, then the script creates the **HA_UNWANTED** directory in the **/etc/rc*.d** directory where the link may be found and places the link in this directory, and then runs the service with the **stop** argument.

In the case of a call with the **undo** parameter, all the links in the **/etc/rc*.d/HA_UNWANTED** directory are moved up one level in the tree structure.

Note The **haunwantedfiles** script works in verbose mode. In the event of a call with the **stop** parameter, several messages similar to "`service xx stop FAILED`" are issued. This is normal. An OK status should be returned when the execution is finished. If the script is run for a second time, the output should be "OK" immediately.

2.4 Understanding Channel Bonding (Optional)

Implementing High Availability on a Management Node and the use or not of Channel Bonding are completely independent. It is the responsibility of the cluster administrator to decide whether or not to use Channel Bonding, assuming that the cluster equipment allows this.

Note	If Channel Bonding is to be used it must be setup before installing High Availability on the Management Nodes. See *HPC Installation and Configuration Guide* for details.

In short, Channel Bonding makes it possible, when a machine is equipped with several Ethernet interfaces, to divide up the work load or to implement High Availability on the different Ethernet interfaces.

As an example, consider a node equipped with two Ethernet interfaces (**eth0** and **eth1**) both of which are connected to the same local network. Either one or the other of these two interfaces can use the same IP address without disturbing the operation of the equipment connected to this local network.

See	For more information about Channel bonding,
see: http://linux-ip.net/html/ether-bonding.html

Chapter 3. Configuring High Availability for the Management Node

Important

Management Node High Availability is not supported for all Bull Advanced Server (BAS) system releases.

See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

3.1 Installing the Primary and Secondary Management Nodes

Important

The IP addresses used will depend on the address plan for the system. Those used in this section are examples only.

In this section, we assume what follows:

- The Primary Management Node is called `ns0` and the Secondary Management Node is called `ns13`.
- The `ns0` IP address is `10.1.0.1`
- The `ns13` IP address is `10.1.0.13`
- The alias IP address is `10.1.0.65`

3.1.1 Prerequisites

- Check that the following HA specific RPMs are installed:

  HighAvailability-x.x-x.noarch.rpm (installed from the Cluster Management CD).

  These RPMs install configuration files in the /etc/HA folder.

- To check the list of the Management Nodes, enter:

```
/etc/HA/bin/haadminnodes
```

According to our example, this command will display:

```
ns0
ns13
```

- The same RPMs must be installed on both `ns0` and `ns13`. To check the installed RPMs on both nodes, run the following command on ns0:

```
/etc/HA/bin/harpms fullstatus
```

- Check that the BMC is reachable for both Management Nodes.

  Note: this is not applicable to the Bull NovaScale Management Nodes managed by a PAP (Platform Administration Processor).

- The ClusterDB has been configured on `ns0` as described before in this chapter.

- The network must be configured on both nodes. Do not define an alias on `ns13`.

- The database (**/var/lib/pgsql**) must **NOT** be a mount point, neither on `ns0` nor `ns13`. It must be present locally on `ns0` and started.

- Change the status of the Secondary Management Node:

```
dbmNode set --name ns13 --status managed
dbmConfig configure --service syshosts
```

- Using the **phpPgAdmin** web interface for the ClusterDB change the value of **CLUSTER.actif-ha** (TABLE.field) field to *true*. A virtual IP address for the backbone must be defined and the value for the **CLUSTER.node_backbone_ipaddr** updated in the **ClusterDB** with **phpPgAdmin**.

- The shared storage system for the Management Nodes must be fully configured and shared LUNs which include sufficient space must have been created. The shared storage systems must not be used, particularly not by the shared LUNs.

- All the directories that will be shared should not be mount points for those which will be managed by **halinks**. All other directories, for example **/tftpboot**, which are mount points, will be managed manually.

3.1.2 Configuring Channel Bonding

The Channel bonding mechanism makes it possible for a system to treat two physical interfaces in a logical bond with only one IP address. It allows the kernel to provide a single logical interface for two physical links connected to two distinct switches, with only one bond being used at the same time. The name of the interface can be indicated by the user, it is usually similar to "`bond0`".

Note Channel Bonding is **NOT** a mandatory feature for High Availability.

Important

The following tasks must be performed on EACH Management Node.

Before beginning the configuration operation, stop the interfaces that will be used, using the **ifdown** command (in the example below these interfaces are named `eth0` and `eth2`).

```
ifdown eth0
ifdown eth2
```

1. Channel Bonding

   In the **/etc/sysconfig/network-scripts** directory of each Management Node, define the following files:

ifcfg-bond0 file

```
DEVICE=bond0
ONBOOT=yes
TYPE=Bonding
BOOTPROTO=none
NETMASK=255.255.0.0
IPADDR= IP address of the node on eth0 interface
(10.1.0.1 for ns0 and 10.1.0.13 for ns13)
USERCTL=no
PEERDNS=no
IPV6INIT=no
```

`NOZEROCONF=yes`	To override the default address provided by Red Hat
`MACADDR`	Optional, however it should not be used.
`NETMASK` and `IPADDR`	Depends on the network configuration of the Management Node.

ifcfg-eth0 file

```
DEVICE=eth0
ONBOOT=yes
TYPE=Ethernet
SLAVE=yes
MASTER=bond0
ISALIAS=no
BOOTPROTO=none
```

`NOZEROCONF=yes`	To override the default address provided by Red Hat

ifcfg-eth2 file

```
DEVICE=eth2
ONBOOT=yes
TYPE=Ethernet
SLAVE=yes
MASTER=bond0
ISALIAS=no
BOOTPROTO= none
```

`NOZEROCONF=yes`	To override the default address provided by Red Hat

The **ifcfg-eth0:1** file must be deleted because the IP alias is managed by the HA software. This file exists only on the Primary Management Node.

`HWADDR` is optional, however it should not be used.

`IPADDR` and `NETMASK` are not used here.

2. Gateway and Route

   Rename the **route-eth0** file in **route-bond0** if it is present.

   You may then find something similar to the following in the **route-eth0** file.

   ```
   10.0.0.0./8 via 10.1.255.254
   ```

Note Replace `10.0.0.0` with the IP address for your network and the second address (`10.1.255.254`) with the address of the switch.

3. modprobe.conf

   In the /etc/modprobe.conf file define the 'bonding' module.

   Assuming that eth0 is the first interface set as "slave" of the channel bonding, add the following lines for the eth0 interface:

```
alias bond0 bonding
options bonding mode=1 downdelay=1000 updelay=1000 miimon=1000
primary=eth0
```

4. Restart the Network

   Run:

```
service network restart
```

3.1.3 Configuring Services

Configuring NTP

As a result of its configuration capabilities NTP is a service which does not require High Availability management with **Cluster Suite**.

Assuming that `10.1.0.1` is the IP address of the primary Management Node and `10.1.0.13` is the IP address of secondary Management Node, edit the /etc/ntp.conf file to configure NTP as follows, if you are using a local clock:

1. On the primary Management Node:

```
restrict default nomodify notrap noquery
restrict 10.1.0.13
restrict 10.0.0.0 mask 255.0.0.0 notrap nomodify
restrict 127.0.0.1
peer 10.1.0.13
server  127.127.1.0     # local clock
fudge   127.127.1.0 stratum 10
driftfile /var/lib/ntp/drift
broadcastdelay  0.008
keys            /etc/ntp/keys
tinker panic 0
tinker stepout 0
```

2. On the secondary Management Node:

```
restrict default nomodify notrap noquery
restrict 10.1.0.1
restrict 10.0.0.0 mask 255.0.0.0 notrap nomodify
restrict 127.0.0.1
peer 10.1.0.1
server  127.127.1.0     # local clock
fudge   127.127.1.0 stratum 10
driftfile /var/lib/ntp/drift
broadcastdelay  0.008
keys            /etc/ntp/keys
tinker panic 0
tinker stepout 0
```

3. Restart the service on both nodes:

```
service ntpd restart
```

Configuring syslog-ng

To avoid socket creation problems during machine boot, it is recommended that a listening socket is created with the address 0.0.0.0, rather than using other virtual or real addresses. A virtual address should be chosen for the sockets that will be used to send logs.

On each Management Node edit the file /etc/syslog-ng/syslog-ng.conf.

1. Search for all the lines which contain the `SUBSTITUTE` string.
2. For each line that is found, check and if necessary modify the lines which follow as indicated below:
   - For each `source` block which contains an IP address set the value 0.0.0.0 instead of the IP alias.
   - For each `destination` block which contains an IP address set the value 127.0.0.1. However, using an alias IP address may be useful if the administrator would like all passive Management Nodes to send their logs to the active Management Node.

3.1.4 Implementing the Primary Node

The steps to implement High Availability on the primary node are as follows:

3.1.4.1 Prepare the Partitions

1. **Create the partitions on the external storage.**

Note The actual disk names may be different than shown in the example below, depending on the number of disks in the system.

For example on the /dev/sde, /dev/sdf, /dev/sdg, /dev/sdh, /dev/sdi devices, run the **mkfs.ext3** command as shown below:

```
mkfs.ext3 /dev/sdf
```

2. **Declare the file systems. One external storage LUN per file system.**

Mount point	File System (example)	Size		
		Mini	Maxi	Recommended
/etcha	/dev/sdf	500 MB	1 GB	1 GB
/usrha	/dev/sdg	5 GB	15 GB	15 GB
/varha	/dev/sdh	20 GB	50% of the remaining = 112 to 121 GB	70 GB
/homeha	/dev/sdi	10 GB	50% of the remaining = 112 to 121 GB	50 GB
/tftpboot	/dev/sde	2 GB	10 GB	10 GB

Edit the **/etc/HA/mnttab file** and add the following lines, according to the above table:

```
/dev/sdf        /etcha
/dev/sdg        /usrha
/dev/sdh        /varha
/dev/sde        /tftpboot
/dev/sdi        /homeha
```

Important

The size of the file systems must fit the space available. See the recommended sizes in the table.

3. Mount these new file systems.

   a. Save the current **tftpboot**:

```
mv /tftpboot /tftpboot.orig
```

   b. Create the mount directories:

```
mkdir /etcha /usrha /varha /homeha /tftpboot
```

   c. Mount the file systems:

```
/etc/HA/bin/hamount start
```

   d. Retrieve the tftpboot data.

```
cp -af  /tftpboot.orig/*  /tftpboot/.
```

   e. To check that the mount operations have been correctly performed run:

```
mount
```

   or:

```
df -h
```

   and check that the **/etcha /usrha /varha /tftpboot /homeha** partitions are mounted.

3.1.4.2 Check Configuration Files

The **/etc/HA** directory contains several configuration files that should be checked to ensure they fit the configuration of your Management Nodes. You should particularly check the following files, which should be customized according to the Resource Manager and the mount points of the shared storage system:

/etc/HA/start
/etc/HA/stop
/etc/HA/status
/etc/HA/synchro
/etc/HA/mnttab
/etc/HA/ip
/etc/HA/links
/etc/HA/cron

Run the command below on both Management Nodes to deactivate the crons:

```
/etc/HA/bin/hacron stop
```

Define the alias IP address

Edit the **/etc/HA/ip** file and add the following line (example):

```
bond0      10.1.0.65/16
bond1 176.18.118.65/24
```

or if bonding is not used

```
eth0     10.1.0.65/16
eth1     176.18.118.65/24
```

3.1.4.3 Configuring Cluster Suite

Run the command below to generate the **/etc/cluster/cluster.conf** file.

```
cs-config-admin
```

3.1.4.4 Control the Services Started at Boot Time

Cluster Suite software stops and restarts some system services. It is mandatory that these services are not started at boot time. The **haunwantedfiles** script ensures this task: it stops the daemons and prevents them from starting at next boot. Run:

```
/etc/HA/bin/haunwantedfiles repair
```

3.1.4.5 Set-up the Links to Shared Disks

To set up the links to shared disks, run:

```
/etc/HA/bin/halinks repair
```

3.1.5 Starting High Availability on the Primary Node

1. Start Cluster Suite by running this command:

```
cs -c start
```

2. Then start the services, running the following command, where `HA_NSM` is the name defined in the **cluster.conf** file.:

```
clusvcadm -e HA_NSM
```

3. Check that the launch is successful, by running the command:

```
clustat
```

and by checking the **/var/log/ha.log** file.

3.1.6 Implementing the Secondary Node

The secondary node(s) must have the same packages installed on them as on the primary node. The different services must be configured except those which are managed by High Availability.

The steps to implement High Availability on the secondary node are as follows:

3.1.6.1 Retrieve /etc/hosts

Retrieve the /etc/hosts file from the primary Management Node and copy it to the secondary Management Node.

3.1.6.2 Synchronize Files

From the primary Management Node, synchronize the files that cannot be put on the shared storage system:

```
/etc/HA/bin/hasynchro repair
```

This command has to be run twice. The first time the command synchronizes the files. The second time the command should return an OK status.

3.1.6.3 Inhibit Daemons for the Launch

From the secondary Management Node, inhibit the daemons controlled by the Cluster Suite by removing the links of /etc/rc.d and by stopping the daemons if they are running.

```
/etc/HA/bin/haunwantedfiles repair
```

3.1.6.4 Set-up Dynamic Links

From the secondary Management Node, set-up the links for the shared directories:

```
/etc/HA/bin/halinks repair
```

Important

Run this command a second time and check that the output is "OK".

3.1.6.5 Prepare the Mount Points for the Shared Partitions

From the secondary Management Node, create directories as follows:

```
mkdir /etcha /homeha /usrha /varha
```

3.1.6.6 Start Cluster Suite

Run:

```
cs -c start
```

Check that Cluster Suite has started correctly and that the primary Management Node is the active node which is running:

```
clustat
```

The output should show that `ns0` and `ns13` are defined as 'member name' and that `ns0` is the 'owner', which means that it is the active node as in the example below.

```
Member Status: Quorate, Group Member

  Member Name                              State      ID
  ------ ----                              -----      --
  ns0                                    Online     0x0000000000000001
  ns13                                   Online     0x0000000000000002

  Service Name          Owner (Last)                  State
  ------- ----          ----- ------                  -----
  HA_NSM                ns0                         started
```

Chapter 4. Managing High Availability on the Management Node

Important

Management Node High Availability is not supported for all Bull Advanced Server (BAS) system releases.

See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

4.1 Cluster Suite

Cluster Suite is the tool used to manage High Availability for the Management Node.

The following commands and service can be used to manage **Cluster Suite**:

cs To start / stop the Cluster Suite daemons

cs-config-admin A command for the automatic configuration of High Availability on the Management Nodes

clustat To obtain online information regarding the Cluster Suite status

clusvcadm To administer Cluster Suite using a command line interface

'HA System Status' service This Nagios service displays the state of the Management Nodes which are running with High Availability. It is used within **NovaScale Master – HPC Edition**

4.1.1 /etc/HA/bin/cs

The **Cluster Suite** daemons are launched using the **cs** command:

```
cs -c start
```

The **Cluster Suite** daemons may be stopped by using the command:

```
cs -c stop
```

Important

Using this command may mean that it is not possible to relaunch the Cluster Suite daemons without rebooting the node.

4.1.2 cs-config-admin

This command, below, configures High Availability on the Management Nodes automatically:

```
/usr/sbin/cs-config-admin
```

There is no need to use the **cs -config-admin** graphical interface.

The command requires:

1. Cluster Database access to the Primary Management Node.

2. A Secondary Management Node must be connected to the Administration network.

Assuming the Primary Management Node can access the ClusterDB, run the **cs -config-admin** command without including any additional parameters.

If the command returns the error below;

```
No da_iocell_component preloaded for management nodes, give HA node
name as parameter please.
```

Then either update the Cluster DataBase with details of the management iocell, or run the command again using the Secondary node name as the parameter, as shown below;

```
cs-config-admin <node_ha_name>
```

The command must have been run successfully before the High Availability mechanism is launched.

4.1.3 clustat

This command is used on the Management Nodes where **Cluster Suite** has been started. It is used to display the status for Cluster Suite, see the example below:

```
[root@nova0 ~]# clustat
```

This will give output similar to that shown below:

```
Member Status: Quorate, Group Member
  Member Name                        State     ID
  ------ ----                        -----     --
  tiger0                             Online    0x0000000000000001
  tiger7                             Online    0x0000000000000002

  Service Name         Owner (Last)                State
  ------- ----         ----- ------                -----
  HA_NSM               tiger0                      started
```

4.1.4 clusvcadm

This command is used on the Management Nodes where **Cluster Suite** has been started. The most useful options are:

clusvcadm -d <service name>	Disable a service
clusvcadm -e <service name>	Enable a service
clusvcadm -e <service name> -m <member>	Enable a service on a member node
clusvcadm -r <service name> -m <member>	Relocate an active service to a specific member node

Refer to the man page for this command for more information.

```
man clusvcadm
```

4.2 Patching a Node / Updating an Application

4.2.1 Introduction

Important

It is not possible to give a generic procedure to apply a patch or to de-install/install an application or a package on Management Nodes, while High Availability is active and operational. It is the responsibility of the administrator to find the best solution. The administrator needs to understand the High Availability mechanism as well as any particular concerns for the application or for the data which needs to be updated.

Below is a list of difficulties which may arise when applying a patch or de-installing/installing a package or an application:

- The impossibility to de-install due to files or directories which cannot be accessed, following the High Availability operation to link them with data on the shared storage system.

- The operations of removing links and replacement by directories can lead to the situation where data remains on the shared storage system. Therefore as the **halinks repair** command does not remove data, the data on the shared storage system will not be updated, and this will result in incompatibilities at reboot, and even data corruption.

- When the patch is run on the active node, a **migration** of data on the shared storage system may result. When another node becomes the active node then this new active node does not include the patch and it will not be able to start some services or will corrupt some data. Furthermore, if a passive node is forcefully patched by manually mounting the file systems, the patch operation may fail and/or corrupt data.

- If the patch is applied to the local data, but not distributed to the data of the shared storage system, then there is a risk of data corruption and/or impossibility of starting the application.

There is no procedure for updating applications on a High Availability Management Node. An example follows for Lustre, it may be necessary to include additional manipulations at the time of a real intervention.

4.2.2 Example of Updating Lustre

In this example, **Lustre** is updated with a patch, using the command:

```
rpm -Uvh <package pathname>
```

The Lustre shared directories are:

/etc/lustre
/var/lib/ldap

Notes

- Updating **Lustre** packages locks the whole cluster, as all the management services will be stopped.
- The update must be performed on all Management Nodes, but using different procedures for the Primary and Secondary Nodes, as described below.
- Free space on the local disks and on the shared storage system is used to perform the update.

Updating Lustre on the Primary Node

The steps to update **Lustre** on the Primary Node are:

1. Stop the services managed by **Cluster Suite**. For example, to stop the **HA_NSM** service, run:

```
clusvcadm -d HA_NSM
```

2. Return the symbolic links to their original state:

```
/etc/HA/bin/halinks undo
```

3. Remove the shared directories:

```
rm -rf /etc/lustre
rm -rf /var/lib/ldap
```

4. Restore the starting scripts to their original state:

```
etc/HA/bin/haunwantedfiles undo
```

5. Mount the shared storage system:

```
/etc/HA/bin/hamount start
```

6. Copy the shared directories from the shared storage system to the local disks:

```
cp -af /etcha/lustre /etc/lustre
cp -af /varha/lib/ldap /var/lib/ldap
```

7. Remove the directories from the shared storage system:

```
rm -rf /etcha/lustre /etc/lustre
rm -rf /varha/lib/ldap
```

8. Apply the patch by running the command:

```
rpm -Uvh <package pathname>
```

9. Restore the symbolic links by running the command:

```
/etc/HA/bin/halinks repair
```

10. De-activate the start of the services managed by **Cluster Suite** by running the command:

```
/etc/HA/bin/haunwatedfiles repair
```

11. It is better to stop **Cluster Suite** on the Secondary Nodes, if these have not been patched, before restarting the service on the Primary Node. On each Secondary Node, run the command:

```
cs -c stop
```

12. Restart the service managed by **Cluster Suite:**

```
clusvcadm -e HA_NSM
```

Updating Lustre on the Secondary Node

The steps to update Lustre on the secondary node are as follows:

1. Stop **Cluster Suite**:

```
cs -c stop
```

2. Return the symbolic links to their original state:

```
/etc/HA/bin/halinks undo
```

3. Restore the starting scripts to their original state by running the command:

```
/etc/HA/bin/haunwantedfiles undo
```

4. Apply the patch:

```
rpm -Uvh <package pathname>
```

5. Restore the symbolic links:

```
/etc/HA/bin/halinks repair
```

6. De-activate the launch of the services managed by Cluster Suite:

```
/etc/HA/bin/haunwatedfiles repair
```

7. Restart Cluster Suite:

```
cs -c start
```

If **Cluster Suite** has not started correctly, reboot the node and then restart **Cluster Suite** once the node has booted, and is reachable.

Chapter 5. Lustre File System High Availability for the I/O Nodes

Important

Lustre File System High Availability for I/O nodes is not supported for all Bull Advanced Server (BAS) system releases.

See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

This chapter explains how to implement High Availability for I/O Nodes and **Lustre** file system and only applies to clusters which have installed the **Lustre** file system from the **XLustre** CDROM.

5.1 Introduction to the Lustre File System

Lustre uses object based disks for storage. Metadata servers are used for storing file system metadata. This design provides a substantially more efficient division of labor between computing and storage resources. Replicated, failover metadata Servers (**MDSs**) maintain a transactional record of high-level files and file system changes. Distributed Object Storage Targets (**OSTs**) are responsible for actual file system I/O operations and for interfacing with storage devices. This division of labor, and of responsibility, leads to a truly scalable file system and more reliable recoverability from failures by providing a combination of the advantages of journaling and distributed file systems. **Lustre** supports strong file and metadata locking semantics to maintain total coherency of the file systems even when there is concurrent access. File locking is distributed across the storage targets (**OSTs**) that constitute the file system, with each OST handling locks for the objects that it stores.

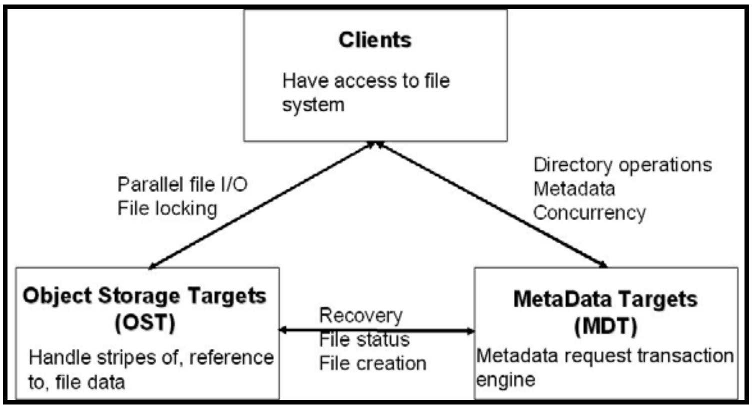

Clients
Have access to file system
Parallel file I/O
File locking
Directory operations
Metadata
Concurrency
Object Storage Targets (OST)
Handle stripes of, reference to, file data
Recovery
File status
File creation
MetaData Targets (MDT)
Metadata request transaction engine

Figure 5-1. Lustre interactions

See Lustre: A Scalable, High-Performance File System Cluster File Systems, Inc. http://www.lustre.org/docs/whitepaper.pdf

5.2 Lustre Failover Mechanism

Lustre supports the notion of failover pairs. Two nodes which are connected to shared storage can function as a failover pair, in which one node is the active provider of the service (OST or MDT), and the second node is the passive secondary server.

The Lustre services are declared on both nodes with the same name. The MDT is configured with a list of servers (OSSs) for clients to pass through in order to connect to the OSTs. The Lustre servers must have distinct network addresses.

The failover mechanism of the Lustre system is based on the capacity to enable client reconnection when the OSTs and MDTs are moved to other nodes.

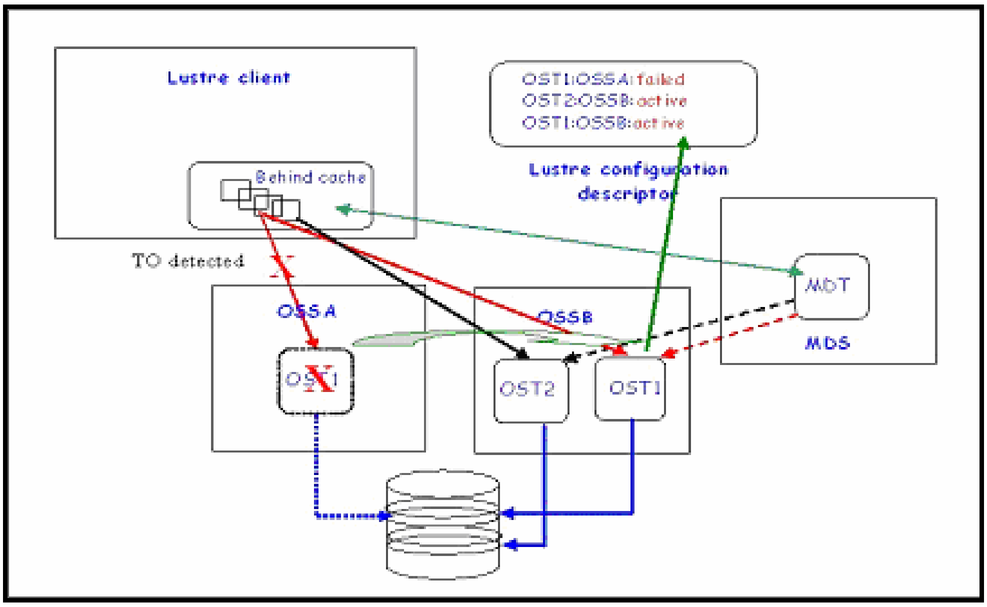


Figure 5-2. OST takeover and client recovery

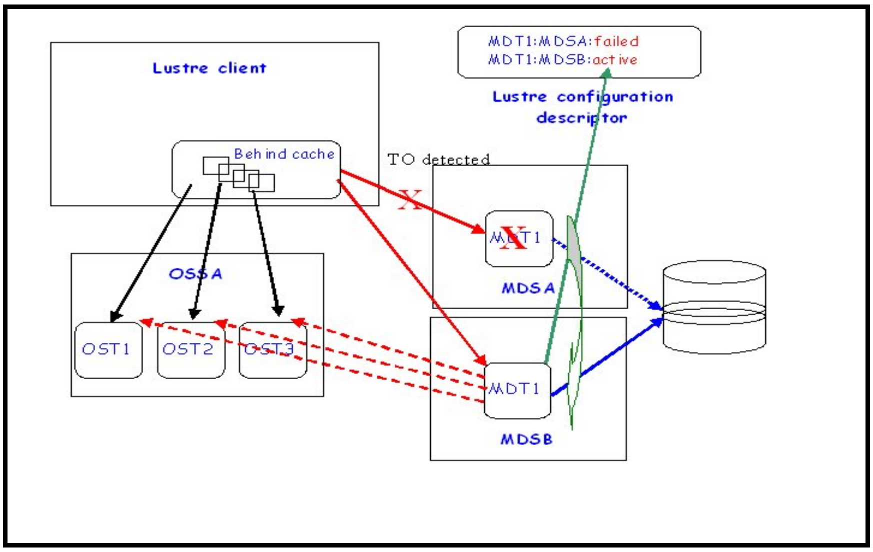


Figure 5-3. MDT takeover and client recovery

Lustre targets work in a synchronous mode: a client request is executed on the storage device and acknowledgement provided for the client only after it has been acknowledged by the device. In the case of a failure, the storage devices will ensure that committed data is preserved. Uncommitted data, meaning data not acknowledged, is kept by the clients in their "reserve cache" and can be resent when the system recovers.

When a request to a target (MDT or OST) is not acknowledged in time, the Lustre client suspects a failure and starts the recovery procedure as follows:

- Checks routing information in the Lustre configuration descriptor
- Reconnects to the migrated target
- Lost transactions are replayed
- Locks are re-acquired
- Partially completed multi-node operations are restarted.

On the server side, the target failover mechanism is similar to the High Availability service migration: the service is "stopped" on one node, and then restarted on the rescue node which has access to the same storage devices.

Transactions requested on metadata local to the targets (ext3 log files) and those which are global for the Lustre system (MDT) are logged in log files. These files are replayed when there is a service migration.

Lustre does not prevent simultaneous access. This means that an external mechanism must ensure that the shared storage will only be accessed by one node at a time. This can be done by powering off the failed node.

5.3 Hardware Architecture

Bull High-Availability management of the Lustre system relies on a specific hardware architecture.

The I/O nodes operating in HA mode are grouped in I/O cells which brings together two I/O nodes that access one or more disk arrays. The I/O cell contains either OSTs or MDTs, but both are exclusive.

Usually, nodes are connected directly to the storage systems ports, without intermediate switches or HUBs. This point to point linking avoids additional active components which increase the risk of system failure by also being SPOFs (Single Point of Failure).

The LUNs within the storage systems are accessible for both nodes of the I/O cell, enabling OSSs and MDSs to retrieve their data when they are moved to the other node. But each LUN must be used by only one node at a time to avoid data corruption.

An I/O fencing mechanism is implemented so that the faulty node can not access the LUNs again after the OSSs or MDSs are restarted on the peer node of the I/O cell. In any case the node which fails is powered off.

The underlying mechanism which ensures OSS and MDS migration and I/O fencing is provided by **Cluster Suite**. The failover process relies on basic entities known as failover services. When a node fails, **Cluster Suite** determines how each service should be relocated.

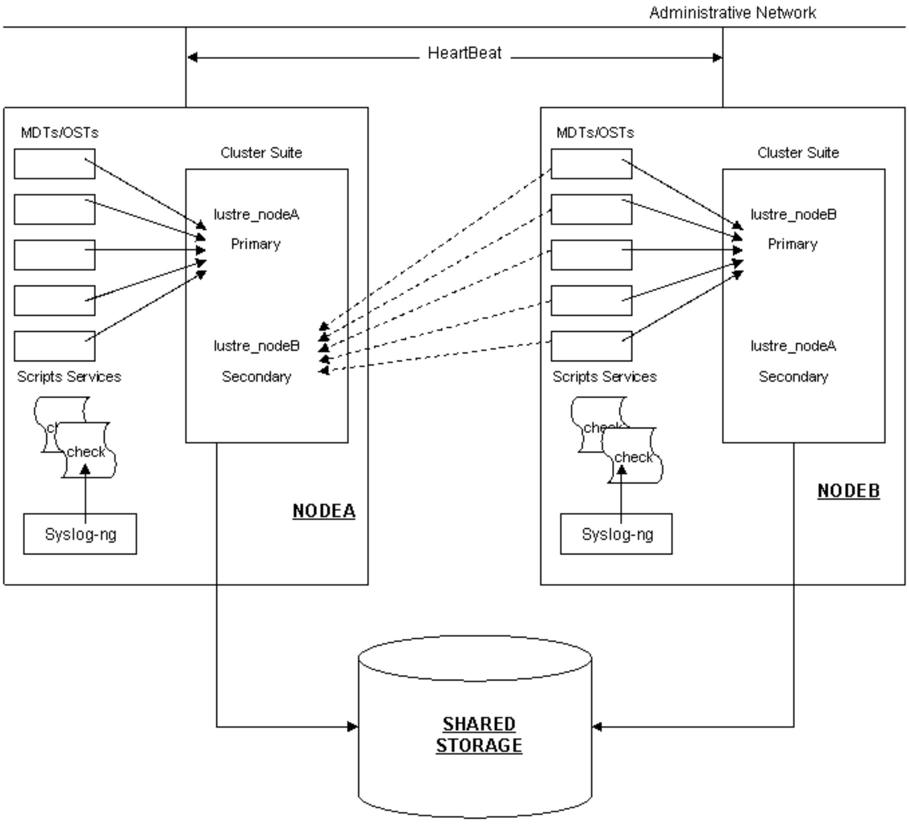


Figure 5-4. I/O Cell diagram

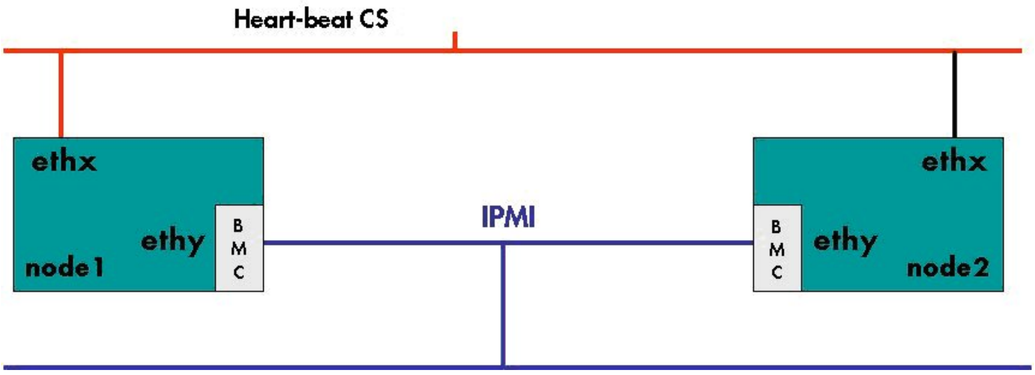


Figure 5-5. High Availability/Cluster Suite on NovaScale R440 and R460 IO/MDS nodes

In the case of multi-types I/O nodes (nodes which serve as both OSSs and MDSs), the Lustre file systems must be configured so that the same node does not support both MDT and OSTs services for the same file system. If not, a failure of this type of node constitutes a double failure (MDS + OSS) for the Lustre file system and its recovery is not guaranteed. The following figure illustrates how you can position OSTs and MDTs for two file systems **FS1** and **FS2**.

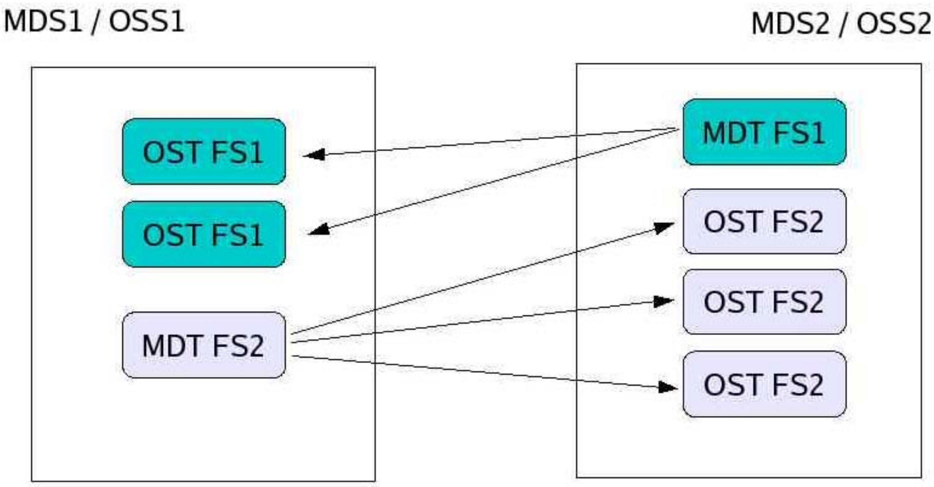


Figure 5-6. MDT/OST Dispatching on two nodes

5.4 High Availability Policy

In a Cluster Manager environment the customer can simply spread the application services across the clustered servers in any way that seems appropriate. All nodes will be actively running a share of the total load.

This is called **Active/Active clustering**, as all servers are active. If one server shuts down, the other servers will pick up the running load of its services.

The High Availability mode to be applied in the Bull HPC context is ***mutual takeover*** for each node of a pair of nodes in the same I/O cell. In standard state, each I/O node supports its own Lustre services (MDTs or OSTs), whereas in a failure state, one node manages its own services plus all the services from the failing node.

Firstly, all Lustre services from the failing node will be migrated to the second node, even if the failure concerns only one Lustre service. It means that for each I/O node, one failover service is defined which includes all the currently installed Lustre services.

Cluster Suite requires a service script for each failover service that will be managed. The script must be able to stop, start and report status for the service.

On each unitary High Availability cluster based on the I/O cells, two failover services are defined: one for the Lustre services of each node.

In the I/O Cell above (Figure 5-4) we have:

lustre_nodeA with primary node NODEA and secondary node NODEB
lustre_nodeB with primary node NODEB and secondary node NODEA

A different failover script is associated with each of the two services provided they are not composed of the same Lustre components (MDTs / OSTs).

At any moment, the Lustre failover service on an I/O node is composed of all the Lustre services (MDTs / OSTs) associated with the **active** file systems. Its composition is subject to change according to the Lustre file systems activation.

On an I/O node, the Lustre services (MDTs / OSTs) are not started for the boot but only by Lustre administrative tools by means of file system start. This to ensure consistency of the Lustre file system services start on all the nodes it relies on.

For a reboot of a failed MDS or I/O node, the Lustre services are not automatically relocated. This is may be done only by the administrator using a Lustre management tool. This mechanism is chosen to avoid inopportune Lustre services migration when there is a partial repair of the primary node.

Important

A simultaneous migration of the management station and of the metadata server is considered as a double failure and is not supported.

5.5 Managing Lustre High Availability

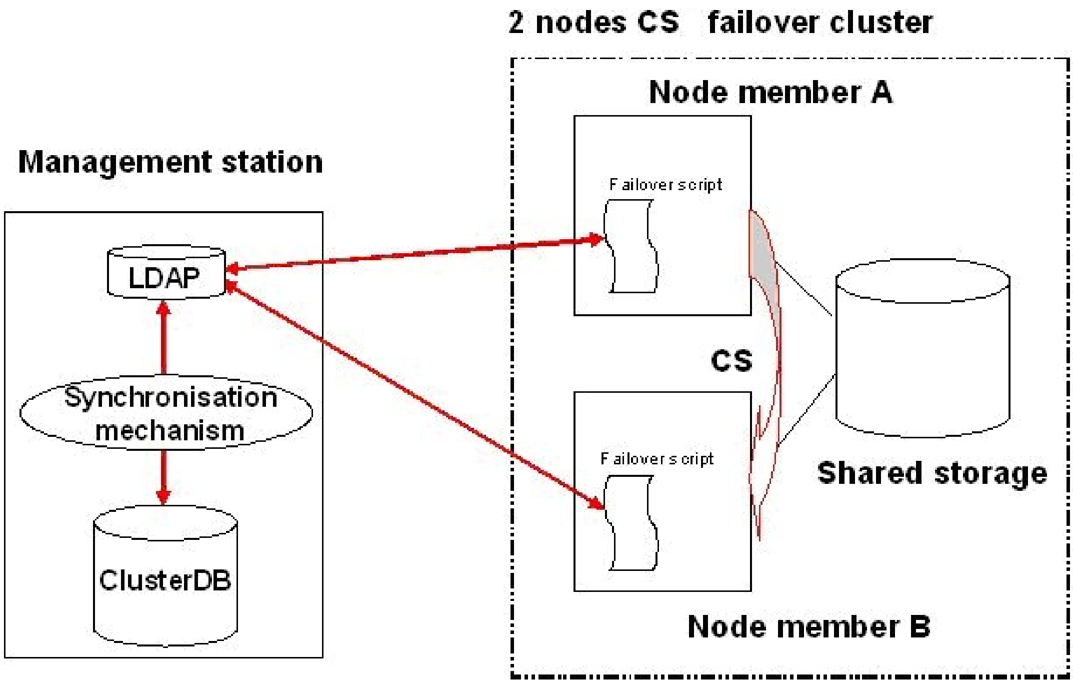


Figure 5-7. Lustre High-Availability Management architecture

When targets (MDT/OST) failover is configured, not only the information about paired targets (a cell) is stored in the Lustre configuration information of the ClusterDB, but also which instance of that target is the currently active one. It is the High Availability application (Cluster Suite script) that has to dynamically update the active instance information of the Lustre tables into the ClusterDB.

LDAP Directory

The LDAP directory is the pivot of the Lustre High Availability architecture. It has been proved to be the more flexible and more efficient way to share Lustre configuration information on a large cluster. It is the repository of the actual file systems distribution on the cluster:

- Which file systems are active,
- Lustre services (OST/MDT) migrations.

The information status it contains is updated by the Lustre failover script every time a High Availability event occurs.

This information status has to be synchronized with the Cluster DB one in order to ensure file systems status and Lustre services migrations monitoring by the Lustre administration tools.

Synchronization with the Cluster DB is carried out by an HTTP server named **lustredbd** which runs on the same node as the **LDAP** daemon.

Using such a mechanism allows Lustre file systems relying on some migrated nodes, to be stopped and restarted respecting the actual Lustre services node distribution.

Clients /Servers Reconnections

An 'epoch' number on every storage controller, an incarnation number on the metadata server/cluster, and a generation number associated with connections between clients and other systems form the infrastructure for Lustre recovery, enabling clients and servers to detect restarts and select appropriate and equivalent servers.

Failover Scripts

The failover scripts for a HA pair of nodes will include different Lustre services. These change according which Lustre file systems are activated/de-activated.

A generic failover script **/usr/sbin/lustre_failover** is implemented on the I/O nodes. The set of Lustre services to be managed is determined dynamically by checking the **LDAP** directory for the active file system and its group of services on a node.

To ensure consistency of services, the failover script associated with a failover service is a symbolic link to the generic script whose name includes the primary node name:

/usr/sbin/lustre_failover_<primary_node_name>

The group of Lustre services to be checked is determined by parsing the script name. The symbolic link is configured when the cluster is deployed.

A **naming convention** is established which is used when the **Lustre** file systems and **Cluster Suite** are configured:

failover service name = lustre_<primary_node_name>

5.6 Error Detection and Prevention Mechanisms

Lustre Single Point of Failure (SPOF) tracking

Tracked Lustre services SPOF include:

1. Service crash (no longer running on the node).

2. Lustre services in an unavailable state (hanging, starting, etc.).

3. Repetitive abnormal comportment (systematic client eviction, etc.).

4. I/O errors on the back-end device.

The Lustre services failures (points 1 to 3) detection relies on the intrinsic Lustre health monitoring system. This internal failures management mechanism maintains diagnose items in the local **/proc/sys/lustre** and **/proc/fs/lustre** directories of each I/O or metadata node.

A regular check of **Lustre** services is scheduled by **Cluster Suite** by the failover script, and the results are returned to the status indicator.

This check will process the diagnose items maintained by **Lustre**.

The detection of I/O errors on the back-end device (point 4) detection relies on the storage management monitoring daemon **storfilter**. When it detects a problem, this daemon warns the Lustre failover script using a dedicated target.

When one these SPOF is detected, a Lustre services migration is triggered followed by a node power off.

5.7 Lustre Failures covered by High Availability

5.7.1 I/O Node and Metadata Failures

I/O Node Panic/Hang

When a Primary Node hangs or panics, the heartbeat message will not be sent within the authorized period. This silence is detected by the Secondary HA node which fences the silent node and then takes over the cluster services.

I/O Node Power down

The Primary Node does not send its heartbeat messages within the authorized period. This silence is detected by the Secondary HA node which fences the silent node, and then takes over the cluster services.

Lustre Software Failure

Two scenarios can take place leading to the same action:

- The health monitoring mechanism of the Lustre system detects the failure and update the health information in the **/proc** directory. The next time the status target of the Lustre failover script is activated by the **Cluster Suite**, it will detect the problem and power off the failing node.
- The status parameter of the **Lustre** failover script detects that some **Lustre** services are missing, or not in the correct state. It will firstly try to restart the service, and if there is no response it will reboot the node where the service is running.

Both scenarios trigger the **I/O node power down** failure treatment.

5.7.2 Storage Failures

Fibre Channel Adapter Errors

Fibre channel adapters are used to access to external storage systems, shared by the two nodes of the HA I/O cell. They store the OSS and MDS data.
Fibre channel adapter errors are ignored. Fibre Channel Link errors are usually transient and do not lead to a node failover. If the adapter error is significant, it leads to a Linux disk error which triggers an alert. In this situation the Primary Node will fail and **Cluster Suite** will switch the service to the Secondary node.

Disk Subsystem Controller Failure

The node detects a disk error when a disk controller fails. Disk errors are monitored by the **I/O status** service which alerts the I/O nodes. In this situation the Primary Node will fail and **Cluster Suite** will switch the service to the Secondary node.

Note **Lustre** verifies that the device generating the I/O error is being used by Lustre. If it is not then no corrective action will be taken.

Local Disk errors

When a local disk fails **Cluster Suite** will switch to the Secondary Node if this disk supports a system file system, e.g. /root.

5.7.3 Ethernet Network Failures

A failure of the heartbeat network will stop heartbeat exchanges leading each node to initiate to service take over. A fencing race starts between both nodes.

Note Only the heartbeat network is monitored by **Cluster Suite**. A failure on another Ethernet network than the one used for heartbeat will not lead to service takeover; however the failure will be displayed on the management node via **NovaScale Master - HPC Edition**.

Ethernet Network Access Failure (NIC or link Failure)

The management network is also used to send fence requests.
The node which is unable to use the management network cannot fence its peer node.
Thus, the peer node wins the fencing race, and takes over the cluster services.
There is no risk of split brain (i.e. both nodes of the I/O cell running simultaneously the same Lustre service).

Management Network Failure

If the management network is unavailable for both nodes of the HA I/O cell, none will be able to fence its peer node. **Cluster Suite** does not initiate any failover action.
If one the node of the HA pair is able to fence the peer node, it wins the fencing race and takes over the services.

Chapter 6. Configuring High Availability for Lustre

Important

Lustre File System High Availability for I/O nodes is not supported for all Bull Advanced Server (BAS) system releases.

See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

6.1 Checking the Cluster Environment

Important

- These checks are mandatory. A cluster may start if they are not all verified but will generate errors at runtime.
- Lustre High Availability is only supported on clusters which include a dedicated Management Node with the Cluster Database in place.

6.1.1 Checking the High Availability Paired Nodes

Run the command below to check the status of the HA paired nodes:

```
lustre_migrate nodestat -n all
# lustre_migrate nodestat -n <nodes list>
```

This will give output similar to that below:

```
HA paired nodes status

----------------------
node name      node status    HA node name    HA node status
xena1            MIGRATED           xena2                 OK
xena10                 OK          xena12           MIGRATED
```

If the nodes are not correctly paired then edit the Cluster DB using the command below

```
lustre_io_node_dba set  --nid <node_id>
                        [--fid <ha_node_id>]
                        [--cid <cluster_id>]
                        [--net <net_status>]
                        [--stor <storage_status>]
                        [--stat <lustre_status>]
                        [--db <db_name>]
```

Important

If you are migrating to BAS5 for Xeon v1.2 the existing HA paired nodes will be used.

See When installing for the first time the I/O resources for Lustre should be configured now, as described in the *Configuring I/O Resources for Lustre* section, in the *Configuring and Sharing File Systems* chapter, in the BAS5 for Xeon *Installation and Configuration Guide*.

6.1.2 Checking the connection between the I/O nodes and the Management Node

The I/O nodes must be connected to the Management Node and have the right to restart some of the services on the Management Node, as and when needed.

Run the command below to check the connection between the I/O nodes and the Management Node (xena0 in the example, below)

```
pdsh -w <all IO nodes> "ssh <management node> echo OK"
```

Example

```
pdsh -w xena[1-2,10,12] "ssh xena0 echo OK"
```

Example output

```
xena10: OK
xena2:  OK
xena1:  OK
xena12: OK
```

6.1.3 Verifying the connection between the I/O nodes and the shared storage device

The same storage links must be available for both High Availability paired I/O nodes. The command below should be used on all I/O paired nodes:

```
pdsh -w <node1,node2> "stormap -l | sort" | dshbak -c
```

Run the command below to identify the links that are missing for a node:

```
stormap -c
```

If this does not work, run the command below:

```
modprobe lpfc
```

Finally, look for a hardware failure.

See The *Configuring I/O Resources for Lustre* section, in the *Configuring File Systems* chapter, in the **BAS5 for Xeon** *Installation and Configuration Guide* for more information.

6.2 Using Cluster Suite

Large clusters may contain multiple I/O cells, and within each I/O cell, the Cluster Suite must be configured. This process is fully automated by Bull cluster management tools. All the necessary information is extracted from the Cluster DB and used to generate the Cluster Suite configuration files. These files are pushed to each node which must be controlled by the Cluster Suite.

Important

Cluster Suite commands not described in the present paragraph must not be used, as they may lead to fatal inconsistency for the Lustre file system. The GUI must not be used as well. All the Cluster Suite setup is predefined to enable the failover process expected by the Lustre file system, and prevent any risk of split brain (i.e. both nodes of the I/O cell running simultaneously the same Lustre service). Administrator must not attempt to modify the Cluster Suite's configuration within I/O cells.

The management tasks for **Cluster Suite** are:

- Distributing the configuration file
- Starting the Cluster Suite

By default, there is not automatic start at boot time, and it is not recommended to enable this.

6.2.1 Distributing the cluster.conf file on the I/O Node

The **/etc/cluster/cluster.conf** is generated using node HA pair defined in the ClusterDB. The following options are selected, and must not be changed:

- Name of the services to be managed.
- Manual start of services (to avoid split brain if a node can not join its peer node).
- List of nodes.
- Heartbeat through the Management Network for **NovaScale R440** and **R460** machines.

Note The quorum disk must have been defined in the model file in order that a LUN is dedicated to it. If a model file is not used and **stordiskname** has been used as an alternative, then a LUN must be configured as a quorum disk (see **mkqdisk** help for more information).

If quorum disk is not included in the Cluster Suite configuration then **stordepha** can be used as follows:

```
stordepha -a -c configure
```

If quorum disk is included in the Cluster Suite configuration, **stordepha** must be used with the **–q** option, as below:

```
stordepha -a -c configure -q
```

The **–a** flag means all nodes. It is possible to exclude some nodes (**-e** flag) or to specify a list of target nodes (**–i** flag, exclusive with **–a**). See the **stordepha** man page for more information.

Note The **cluster.conf** file is not preserved when a node is reinstalled by **KSIS.** It must also not be integrated in a node image. After each node has been deployed, the **stordepha** command must be used to restore the node's configuration.

Restoring a node

After restoring the system on a node, the **cluster.conf** file has to be rebuilt using the **stordepha** command, as described above.

6.2.2 Starting / Stopping Cluster Suite's Daemons

Cluster Suite's daemons can be configured from the Management Node on all or on a subset of the High Availability I/O nodes, or configured locally on each node. In both cases, all the daemons required must be started and stopped in the right order.
Starting **Cluster Suite** will start the Cluster Suite daemons but not the Cluster Suite services as automatic start is disabled. Stopping **Cluster Suite** on a node causes its services to fail on the active node.

- Run the command below from the Management Node:

```
stordepha -c start|stop -a
```

The **–a** flag means all nodes. It is possible to exclude some nodes (**-e** flag) or to specify a list of target nodes (**–i** flag, exclusive with **–a**). See the **stordepha** man page for more information.
Or

- Run the command below on the node locally:

```
storioha -c start|stop
```

6.2.3 Checking the Status of Cluster Suite

Cluster Suite's status can be verified from the Management Node on all or on a subset of the High Availability I/O nodes, or locally on each node.

- Run the command below from the Management Node:

```
stordepha -c status -a
```

The **–a** flag means all nodes. It is possible to exclude some nodes (**-e** flag) or to specify a list of target nodes (**–i** flag, exclusive with **–a**). See the **stordepha** man page for more information.

Or:

- Run the command on a node locally:

```
storioha -c status
```

Alternatively, it is also possible to use the Cluster Suite's **clustat** command:

```
clustat
```

Or:

```
clustat -i <refresh period>
```

6.3 Configuring Lustre High Availability

Lustre High Availability is configured using different add-ons and specific tools from the Management Node.

The management tasks for **Lustre** failover are:

- Setting up the hardware and software configuration information.
- Enabling failover support for the file systems.
- Managing the migration of the nodes and take over of the **Lustre** services.

6.3.1 Cluster DB Information

Two kinds of information are included in the **Lustre** Cluster DB tables to allow failover management and the monitoring of **Lustre** services:

- Static information linked with the cabling schema for the paired nodes.
- Dynamic information about the migration of the nodes and the distribution of **Lustre** services.

The **lustre_io_node** table for each M(etadata) and I(/O) node lists:

- The identity of the paired nodes, pre-loaded when the cluster is installed.
- The current migration status, this is kept up to date by the failover management tools.

lustre_ost and **lustre_mdt** tables for each Lustre service list:

- The primary and secondary nodes installed by the storage/Lustre deployment process
- The node that is active. This is set dynamically by the failover management tools.

If necessary, this information can be updated using the standard **lustre_tables_dba** tools.

Note The **mds_ha_node** and **oss_ha_node** fields are initialized using the contents of the **lustre_io_node** tables.

6.3.2 LDAP Directory – the lustre_ldap Utility

The **Lustre LDAP** directory on the Management Node contains the description of all **Lustre** file systems installed on the cluster’s I/O nodes, including details of the distribution of their services.

When a file system starts, it is noted as active in the **LDAP** directory. The **lustre_util** utility uses failover scripts to manage the takeover of the services for a **Lustre** file system.

Lustre failover scripts check the **LDAP** directory each time a failover event occurs on a node, in order to obtain the list of **Lustre** services to act on. The scripts update the **LDAP** directory each time there is a migration. A synchronization mechanism transfers this information to the cluster database.

Important

lustre_ldap can also be used to update the **LDAP** directory. This must be done very carefully otherwise the consistency of the failover information may be broken.

6.3.3 Installing the Lustre LDAP Directory

1. Create the **/var/lib/ldap/lustre** directory:

```
mkdir -p /var/lib/ldap/lustre
chown ldap.ldap /var/lib/ldap/lustre
```

2. Enable and start the **LDAP** service:

```
chkconfig --level 345 ldap
service ldap start
```

3. For a first installation initialize the **LDAP** directory, using the command below:

```
ldapadd -D cn=Manager,fs=lustre -w secret -x -H ldap://<hostname>/ -f
/usr/share/lustre/top.ldif
```

4. Go to the **/etc/sudoers** configuration file and verify that the **ldap** user has access to the **lustre_tables_dba** commands:

   The following should appear:

```
dap ALL=(root)NOPASSWD:/usr/sbin/lustre_ost_dba *
ldap ALL=(root)NOPASSWD:/usr/sbin/lustre_mdt_dba *
ldap ALL=(root)NOPASSWD:/usr/sbin/lustre_io_node_dba *
ldap ALL=(root)NOPASSWD:/usr/sbin/lustre_ldap *
ldap ALL=(root)NOPASSWD:/usr/sbin/lustre_util tune_servers ,
/usr/sbin/lustre_util tune_servers  *
```

   If not, use the **visudo** tool to update the **/etc/sudoers** file.

5. Edit the **/etc/lustre/lustre.cfg** management configuration file and set the **LUSTRE_LDAP_URL** with the name of the Management Node for the failover file systems (ldap://<Mgmt_Node_name>/).

6.3.4 Managing LDAP

The **lustre_ldap** utility manages the LDAP directory, using:

- Callback for **lustre_util** and for the **LDAP** server to ensure the Cluster DB is synchronized.
- An online interface to display the **LDAP** directory contents.

Lustre file system HA status

The command below displays the High Availability status of the **Lustre** file system:

```
lustre_ldap show [-f <file_system_name>]
```

The **active** status is shown as either:

- **Yes** if the file system is managed by High Availability
- **No** if the file system is NOT managed by High Availability

If no file system parameters are specified, the command will show the status of all the file systems loaded in the **LDAP** directory.

Lustre file system description

The command below lists the **LDAP** descriptor of the file system in **LDIF** format.

```
lustre_ldap list [-f <file_system_name>]
```

If no file system parameters are specified, the command will list the descriptors for all the file systems loaded in the **LDAP** directory. The **LDIF** display format is mainly used for maintenance purposes.

6.3.5 Cluster DB Synchronisation using lustredbd

1. Set the **/etc/lustre/lustre.cfg** parameters, shown below, to allow the replication of **LDAP** in the Cluster DB:

   LUSTRE_DB_DAEMON_HOST=<hostname>

   LUSTRE_DB_DAEMON_PORT=<tcp port>

2. Set **LUSTRE_DEBUG** to "yes" to enable the failover tools trace feature.

   On each I/O node, the Lustre failover scripts will log events in the **/var/log/lustre** directory. On the Management Node, the **lustre_ldap** daemon will log events in the **/tmp/log/lustre** directory.

3. Start **lustredbd**, the **LDAP** replication daemon:

```
service lustredbd.sh start
```

```
Starting lustredbd:
.....
lustredbd started [OK]
```

4. Check the status for the **LDAP** replication daemon:

```
service lustredbd.sh status
```

```
lustredbd is running
```

Note A daily log file will be created daily in the **/var/log/lustre** directory on the Management Node.

6.3.6 Configuring and starting the Lustre MGS Service

See The *Configuring the Lustre MGS Service* section, in the *Configuring File Systems* chapter, in the **BAS5 for Xeon** *Installation and Configuration Guide* for more details on configuring and starting the **MGS** service.

6.3.7 Managing Lustre Failover Services on the I/O and Metadata Nodes – the lustre_migrate Tool

Lustre failover services are used by **Cluster Suite** to control the migration of the Lustre OST/MDT services.

⚠ WARNING
The failover services have to be started before the Lustre file systems are started. They can be stopped only when all Lustre file systems have stopped running.

The **lustre_migrate** command manages the **Lustre** failover services on the cluster.

If no file system parameters are specified, the command will act on all the I/O and Metadata nodes.

Start/Stop/Status failover services

1. Use the command below to display the status of the **Lustre** failover services on the I/O and Metadata Nodes specified. If no file system parameters are specified, the command will act on all the I/O and Metadata nodes.

```
lustre_migrate hastat -n <node_list>
```

2. Use the command below to start the **lustre_failover** services on the nodes listed. The use of **node_list** is mandatory; **-n all,** which uses the nodes listed in the Cluster DB, is also supported.

```
lustre_migrate hastart -n <node_list>
```

Before starting High Availability with the **-n all** option, try the option with the **lustre_migrate hastat** command to see which nodes are listed. The command will fail if one of the nodes is missing.

Example

node1 and **node2** are a pair of HA nodes. If for some reason **node1** is down, then there will be problems when **Cluster Suite** is launched for the first time, and if **node1** was up the last time **Cluster Suite** launched the **node1** service.

When you run the command, below, to start the **node1** service on **node1** or on the owner of the **node1** service, the command will fail.

```
lustre_migrate hastart -n node1
```

In this example when the command fails try running the command:

```
lustre_migrate hastart --force -n node1
```

This command will try to force **node1** to start. If it is unable to start **node1** it will try to start the **node1** service on its failover node.

3. Use the command below to stop the **Lustre** failover services on the I/O and metadata nodes listed. If no file system parameters are specified, the command will act on all the I/O and Metadata nodes.

```
lustre_migrate hastop -n <node_list>
```

6.3.8 Migration failover services

The command, below, will migrate the **Lustre** services for a node to its HA paired node, so that it can be stopped without disturbing the **Lustre** file system. This is used for maintenance purposes.

```
lustre_migrate export -n <node_name>
```

Specifically, this command will stop the **lustre_<node_name>** failover service (**<node_name>** node is the primary node), and restart it on the secondary node. The secondary node information is taken from the Cluster DB.

Note If the **lustre_<node_name>** failover service was already running on its secondary node, the command has no effect.

Use the command below to relocate a failover service to its primary node, once it has been repaired:

```
lustre_migrate relocate -n <node_name>
```

This command will stop the **lustre_<node_name>** failover service (**<node_name>** node is the primary node) on the secondary node, and restart it on the primary node. The primary node information is taken from the Cluster DB.

Note If the **lustre_<node_name>** failover service was already running on its primary node, the command has no effect.

6.3.9 Configuring File Systems for Failover

Configuring file systems for failover means configuring two HA paired **OSS/MDS** Servers to support either **OST** or **MDT** targets, one server will be the primary node, the other the secondary node.

The failover feature is declared in the **/etc/lustre/models/<file_system_name>.lmf** model file. In the file system model update the following parameter:

- **failover=yes** Enables the failover configuration to be generated

A file system is usually described as being composed of one **MDT** and several **OST**s taken from the Cluster DB.

The secondary node declared in the Cluster DB will be taken into account for the secondary **OST/MDT** declaration.

The file system is managed using the **lustre_util** utility in a standard way. The failover specifics (**LDAP** directory interaction, status display, alternative mount, etc.) are automatically supported by the **Lustre** management tools.

6.4 Lustre High Availability Processes

This diagram below shows the sequence of events following a system failure leading to switch of nodes for the Lustre services. The switch is handled automatically by **Cluster Suite**.

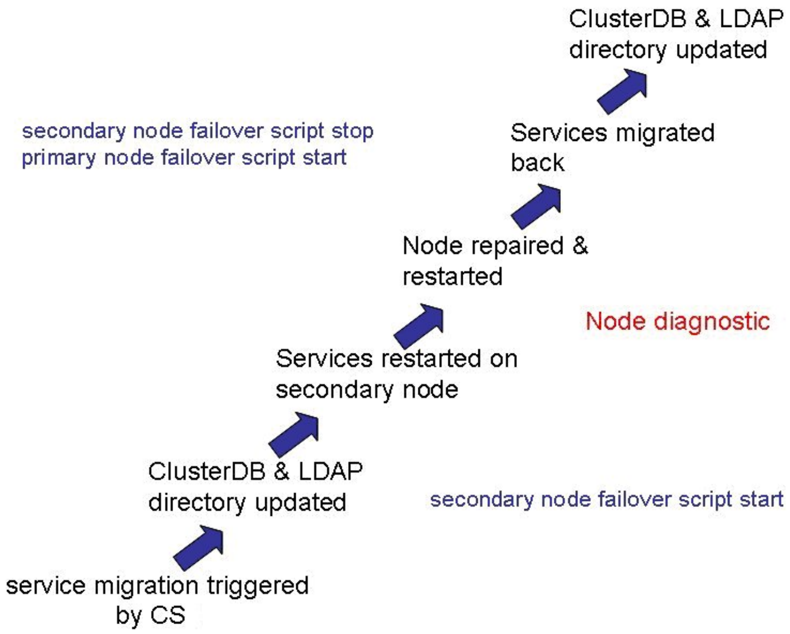

ClusterDB & LDAP directory updated
secondary node failover script stop
primary node failover script start
Services migrated back
Node repaired & restarted
Node diagnostic
Services restarted on secondary node
ClusterDB & LDAP directory updated
secondary node failover script start
service migration triggered by CS

Figure 6-1. Service migration triggered by Cluster Suite

This diagram below shows the sequence of events following a system failure leading to switch of nodes for the **Lustre** services. In this case the Administrator manually controls operations, for example when there is a need to insulate a node without stopping the **Lustre** system.

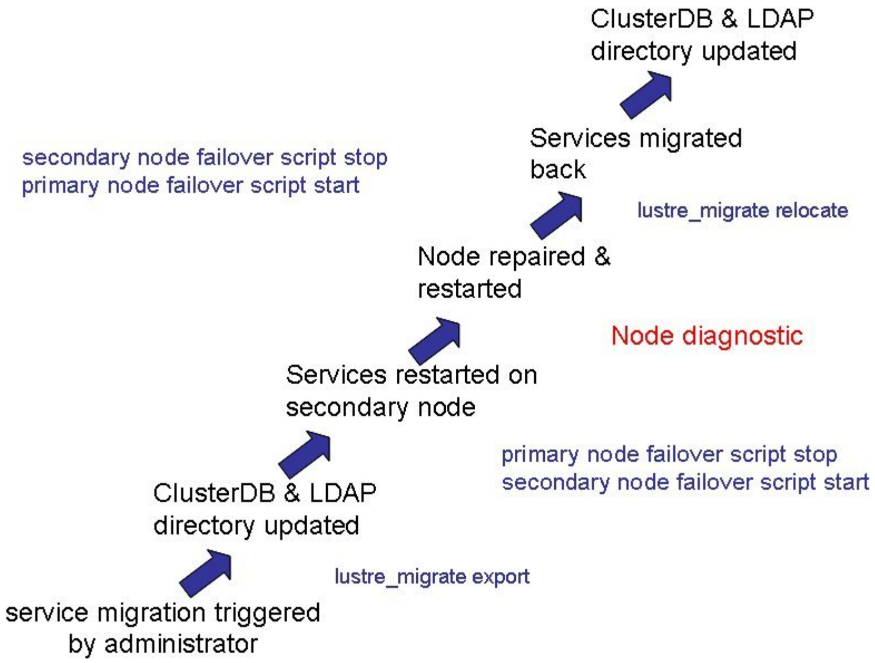

ClusterDB & LDAP directory updated
Services migrated back
secondary node failover script stop
primary node failover script start
lustre_migrate relocate
Node repaired & restarted
Node diagnostic
Services restarted on secondary node
primary node failover script stop
secondary node failover script start
ClusterDB & LDAP directory updated
lustre_migrate export
service migration triggered by administrator

Figure 6-2. Service migration triggered by the Administrator

6.5 Monitoring Lustre High Availability

Two approaches are available from the monitoring tools: nodes migrations and the resulting OST/MDT file systems actual distribution.

On line commands allow the administrator to get an instant status of the Lustre High-Availability system.

If the cluster has a management node, important global health indicators are available via **NovaScale Master - HPC Edition** main view. They constitute a warning system for the administrator.

A trace system can be activated for debug and problem resolution purpose.

6.5.1 Command Line Monitoring

The following command displays the current failover paired nodes status under the form of an array with one line for each pair of nodes, as follows:

```
lustre_migrate nodestat
```

```
node name     node status      node HA name     node HA status
ns6           OK               ns7                 MIGRATED
```

For each node, the status is that of the Lustre failover service it is primary for:

KO the Lustre failover service is UP

WARNING the Lustre failover service is UP some Lustre services are missing. A node migration may be in progress

MIGRATED the Lustre failover service has successfully migrated to the paired node and is now running on it

CRITICAL the Lustre failover service is no longer operating. The node migration has failed

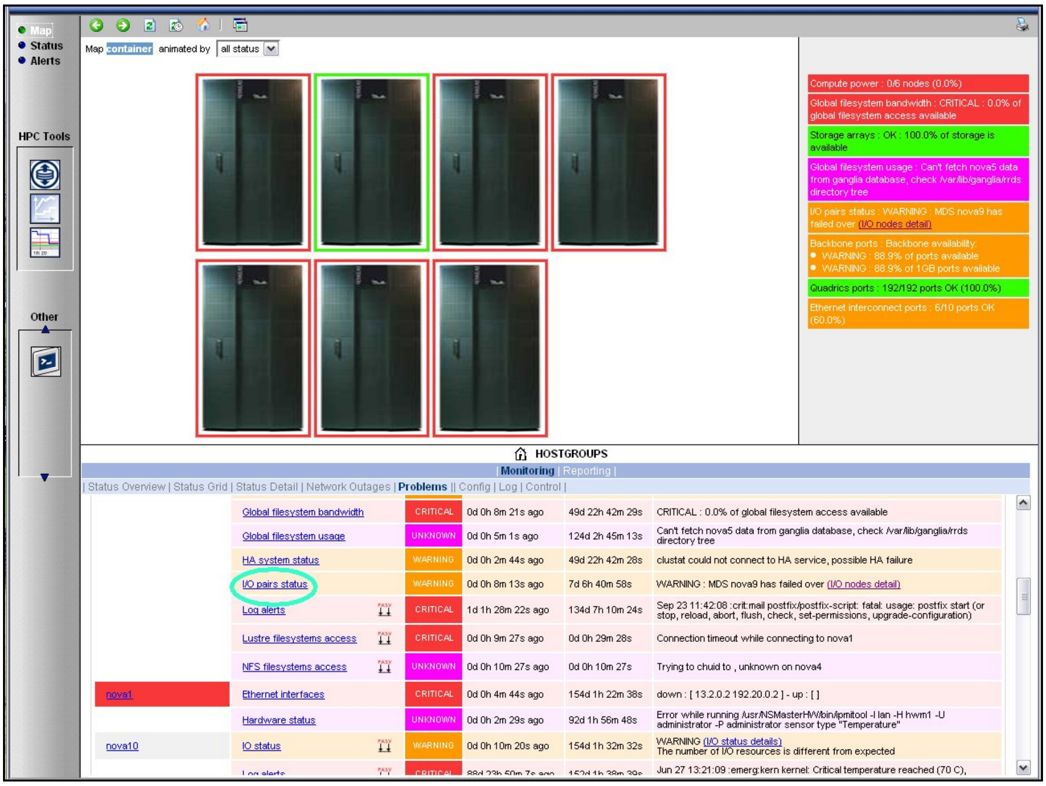
The following command displays the current Lustre failover services distribution and status as seen by the Cluster Suite.

```
lustre_migrate hastat
```

For each Lustre file system installed on the cluster, the following command displays the detailed distribution of the MDTs and OSTs.

```
lustre_util info -f <File system name>
```

6.5.2 Graphic Monitoring


Map
Status
Alerts
HPC Tools
Other
Map container animated by all status
HOSTGROUPS
| Monitoring | Reporting |
| Status Overview | Status Grid | Status Detail | Network Outages | Problems || Config | Log | Control |

Compute power : 0/6 nodes (0.0%)

Global filesystem bandwidth : CRITICAL : 0.0% of global filesystem access available

Storage arrays : OK : 100.0% of storage is available

Global filesystem usage : Can't fetch nova5 data from ganglia database, check /var/lib/ganglia/rrds directory tree

I/O pairs status : WARNING : MDS nova9 has failed over (I/O nodes detail)

Backbone ports : Backbone availability:
- WARNING : 88.9% of ports available
- WARNING : 88.9% of 1GB ports available

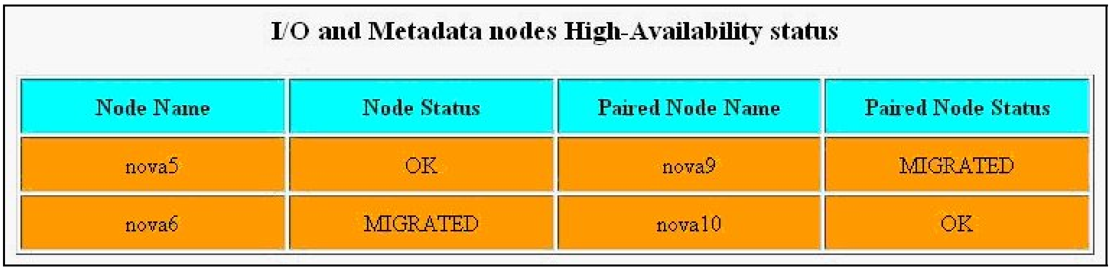
Quadrics ports : 192/192 ports OK (100.0%)

Ethernet interconnect ports : 6/10 ports OK (60.0%)

	Global filesystem bandwidth	CRITICAL	0d 0h 8m 21s ago	49d 22h 42m 29s	CRITICAL : 0.0% of global filesystem access available
	Global filesystem usage	UNKNOWN	0d 0h 5m 1s ago	124d 2h 45m 13s	Can't fetch nova5 data from ganglia database, check /var/lib/ganglia/rrds directory tree
	HA system status	WARNING	0d 0h 2m 44s ago	49d 22h 42m 28s	clustat could not connect to HA service, possible HA failure
	I/O pairs status	WARNING	0d 0h 8m 13s ago	7d 6h 40m 58s	WARNING : MDS nova9 has failed over (I/O nodes detail)
	Log alerts	CRITICAL	1d 1h 28m 22s ago	134d 7h 10m 24s	Sep 23 11:42:08 :crit:mail postfix/postfix-script: fatal: usage: postfix start (or stop, reload, abort, flush, check, set-permissions, upgrade-configuration)
	Lustre filesystems access	CRITICAL	0d 0h 9m 27s ago	0d 0h 29m 28s	Connection timeout while connecting to nova1
	NFS filesystems access	UNKNOWN	0d 0h 10m 27s ago	0d 0h 10m 27s	Trying to chuid to , unknown on nova4
nova1	Ethernet interfaces	CRITICAL	0d 0h 4m 44s ago	154d 1h 22m 38s	down : [13.2.0.2 192.20.0.2] - up : [ ]
	Hardware status	UNKNOWN	0d 0h 2m 29s ago	92d 1h 56m 48s	Error while running /usr/NSMasterHW/bin/ipmitool -I lan -H hwm1 -U administrator -P administrator sensor type "Temperature"
nova10	IO status	WARNING	0d 0h 10m 20s ago	154d 1h 32m 32s	WARNING (I/O status details) The number of I/O resources is different from expected
	Log alerts	CRITICAL	88d 23h 50m 7s ago	152d 1h 38m 39s	Jun 27 13:21:09 :emerg:kern kernel: Critical temperature reached (70 C),

Figure 6-3. NovaScale Master Map all status screen with the **I/O pairs status** service highlighted for a host

The **I/O pairs status** alert indicates if a migration of the Metadata server has occurred. If a Warning is displayed click the **IO nodes details** link to see more details. The HA status for the I/O and Metadata nodes appears as shown in the example below

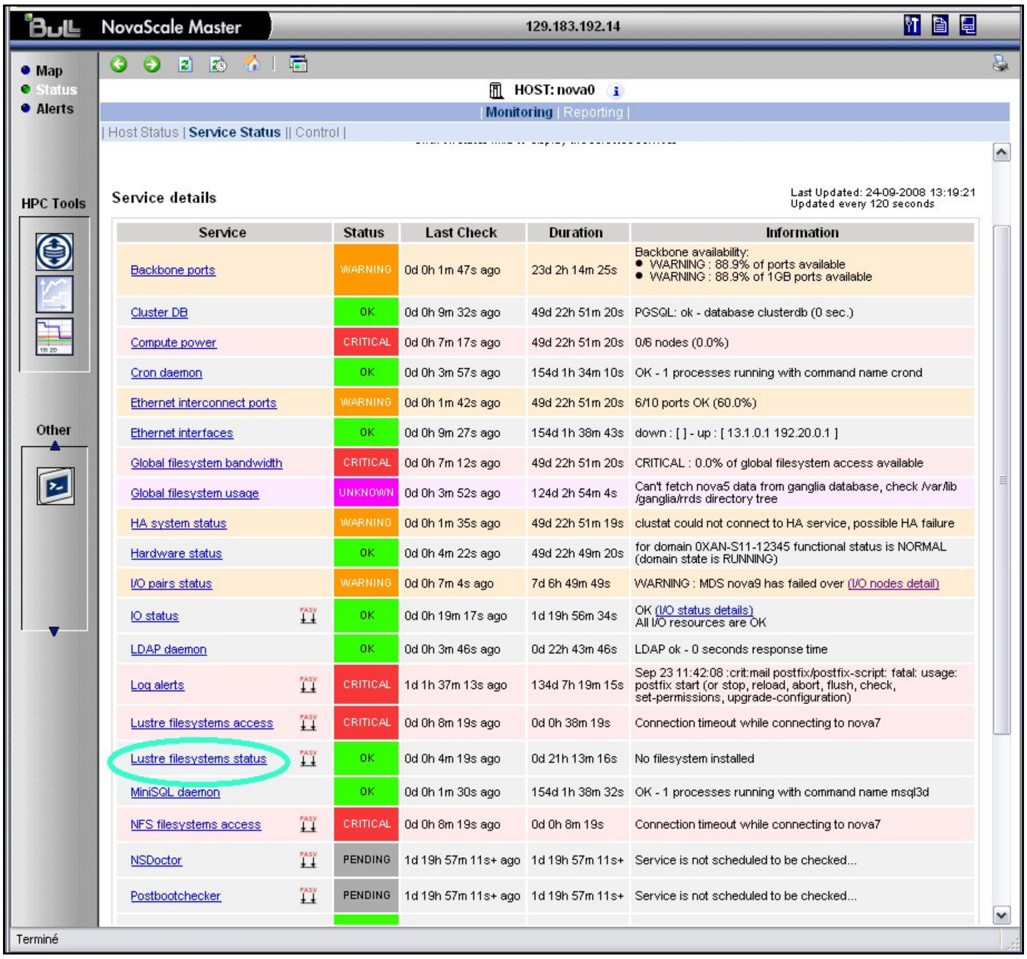


I/O and Metadata nodes High-Availability status

Node Name	Node Status	Paired Node Name	Paired Node Status
nova5	OK	nova9	MIGRATED
nova6	MIGRATED	nova10	OK

Figure 6-4. I/O and MetaData Node HA status

If a migration has occurred, the **Lustre** system is no longer Highly Available and an intervention by the Administrator to restore will be required highly urgently.



Service details

Last Updated: 24-09-2008 13:19:21
Updated every 120 seconds

Service	Status	Last Check	Duration	Information
Backbone ports	WARNING	0d 0h 1m 47s ago	23d 2h 14m 25s	Backbone availability: ● WARNING : 88.9% of ports available ● WARNING : 88.9% of 1GB ports available
Cluster DB	OK	0d 0h 9m 32s ago	49d 22h 51m 20s	PGSQL: ok - database clusterdb (0 sec.)
Compute power	CRITICAL	0d 0h 7m 17s ago	49d 22h 51m 20s	0/6 nodes (0.0%)
Cron daemon	OK	0d 0h 3m 57s ago	154d 1h 34m 10s	OK - 1 processes running with command name crond
Ethernet interconnect ports	WARNING	0d 0h 1m 42s ago	49d 22h 51m 20s	6/10 ports OK (60.0%)
Ethernet interfaces	OK	0d 0h 9m 27s ago	154d 1h 38m 43s	down : [ ] - up : [13.1.0.1 192.20.0.1]
Global filesystem bandwidth	CRITICAL	0d 0h 7m 12s ago	49d 22h 51m 20s	CRITICAL : 0.0% of global filesystem access available
Global filesystem usage	UNKNOWN	0d 0h 3m 52s ago	124d 2h 54m 4s	Can't fetch nova5 data from ganglia database, check /var/lib /ganglia/rrds directory tree
HA system status	WARNING	0d 0h 1m 35s ago	49d 22h 51m 19s	clustat could not connect to HA service, possible HA failure
Hardware status	OK	0d 0h 4m 22s ago	49d 22h 49m 20s	for domain 0XAN-S11-12345 functional status is NORMAL (domain state is RUNNING)
I/O pairs status	WARNING	0d 0h 7m 4s ago	7d 6h 49m 49s	WARNING : MDS nova9 has failed over (I/O nodes detail)
IO status	OK	0d 0h 19m 17s ago	1d 19h 56m 34s	OK (I/O status details) All I/O resources are OK
LDAP daemon	OK	0d 0h 3m 46s ago	0d 22h 43m 46s	LDAP ok - 0 seconds response time
Log alerts	CRITICAL	1d 1h 37m 13s ago	134d 7h 19m 15s	Sep 23 11:42:08 :crit:mail postfix/postfix-script: fatal: usage: postfix start (or stop, reload, abort, flush, check, set-permissions, upgrade-configuration)
Lustre filesystems access	CRITICAL	0d 0h 8m 19s ago	0d 0h 38m 19s	Connection timeout while connecting to nova7
Lustre filesystems status	OK	0d 0h 4m 19s ago	0d 21h 13m 16s	No filesystem installed
MiniSQL daemon	OK	0d 0h 1m 30s ago	154d 1h 38m 32s	OK - 1 processes running with command name msql3d
NFS filesystems access	CRITICAL	0d 0h 8m 19s ago	0d 0h 8m 19s	Connection timeout while connecting to nova7
NSDoctor	PENDING	1d 19h 57m 11s+ ago	1d 19h 57m 11s+	Service is not scheduled to be checked...
Postbootchecker	PENDING	1d 19h 57m 11s+ ago	1d 19h 57m 11s+	Service is not scheduled to be checked...

Figure 6-5. **Lustre filesystem status** indicator in the Host service status window

The **Lustre filesystems status** indicator warns about failures. Clicking on the info link will display a detailed status for MDTs/OSTs.

6.5.3 Traces and Debug

Failover Tools Traces

These are enabled by setting the **LUSTRE_DEBUG** parameter of the **/etc/lustre/lustre.cfg** file to **yes**.

On the Management Node, a daily log file, for example **//tmp/log/lustre/LDAP-<dd mm>.log**, is recorded under the **/tmp/log/lustre** directory by the **lustre_ldap** daemon. It gives information about migration events transmitted to the **LDAP** directory.

On the Management Node, a daily log file is recorded under the **/var/log/lustre** directory by the **lustredbd** daemon. It records **all** the connections and update requests to the Cluster DB.

On the I/O and metadata nodes, a daily log file is recorded under the **/var/log/lustre** directory by the **lustre_failover** scripts. It gives information about failover events and their management.

Note The Administrator should remove the old log files on a regular basis.

System Log Files

On each I/O and Metadata Node, **Cluster Suite** and the failover scripts log events in the **/var/log/messages** and the **/var/log/syslog** files. These files are centralized on the Management Node by the **syslog-ng** system.

Chapter 7. Lustre and Management Node High Availability

Important

Lustre File System High Availability together with Management Node High Availability is not supported for all Bull Advanced Server (BAS) system releases.

See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

The section describes the **Lustre** features which have to be configured so that Management Node High Availability is supported.

See *Chapters 2*, *3* and *4* in this manual for more information on the Installation and Configuration of Management Node High Availability

7.1 Lustre and the Management node

In a standard configuration three specific **Lustre** components run on the Management Node:

Lustre MGS service

This component is used by **Lustre** to store the global Lustre configuration and includes information such as the location of the different Lustre server nodes. Lustre clients refer to it for all requests. Lustre I/O nodes call it to enable their connections. It is not a problem if **MGS** is missing, as long as a Lustre component is not re/started. All running Lustre components include a copy of **MGS** in their cache. **MGS** has to be available, so that it can be updated, when a new Lustre component is started.

Lustredbd (Lustre DataBase daemon) service

The **lustredbd** service keeps the coherence between the **ClusterDB** and **ldap** up to date This service is required when there is a HA failover at start time for a service, so that the service can be restarted remotely. The **lustre_migrate hastart** command refers to **lustredbd** when there is a software, or hardware failover, on any Lustre I/O High Availability node.

Ldap daemon

This has the same use as **lustredbd**. It is referred to by all Lustre I/O nodes when there is a status request. Lustre I/O nodes have a local copy of the **ldap** content, and are capable of responding to a status request without contacting the **ldap** server. **Ldap** is mandatory when a Lustre service starts.

7.1.1 Lustre Services Managed by the Management Node

High Availability is provided by the High Availability feature, some data is shared and some services are managed by the High Availability software, as shown in *Figure 7-1. Management of Lustre components for systems with Management Node High Availability.*.

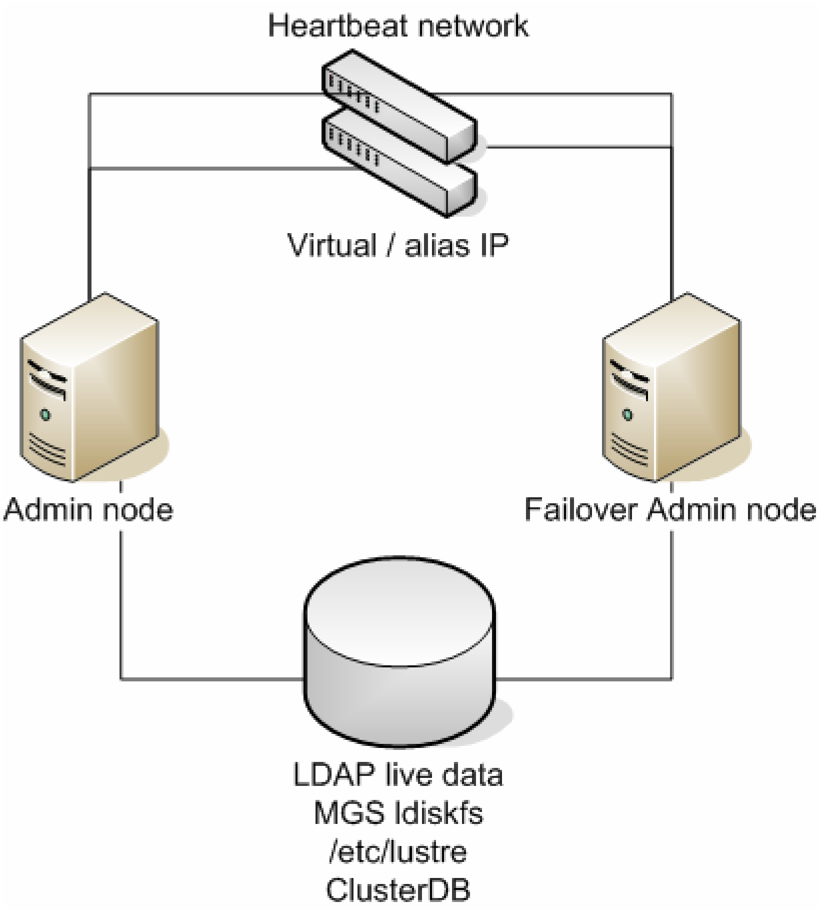


Figure 7-1. Management of Lustre components for systems with Management Node High Availability.

Shared data

To support High Availability on the Management Node, the location of all the files and folders required when Lustre is running should be looked at carefully. All dynamic data has to be shared by both the primary and secondary Management Nodes using a shared storage device. Cluster Suite's management tools will mount the shared data on the active Management Node only for an active/passive Highly Available Management Node.

The dynamic **Lustre** data is stored in the **/etc/lustre** folder on the Management Node.

WARNING
To avoid the shared data being mounted twice the High Availability software should manage the mounting of the shared storage device.

Required services

The **Lustre** services that are managed by the HA Management Node software should be listed in the HA scripts, for example **lustre_failover** which lists the **start**, **stop** and **status** targets. These services include **MGS**, **LDAP** and **lustredbd**. Lustre also needs access to the Cluster DB, through **lustredbd** for failover cases or directly through the use of the **lustre_util** tool.

7.1.2 Configuring the MGS Service

See the diagram, below, for details of how the **MGS** Service should be configured on Highly Available Management Nodes.

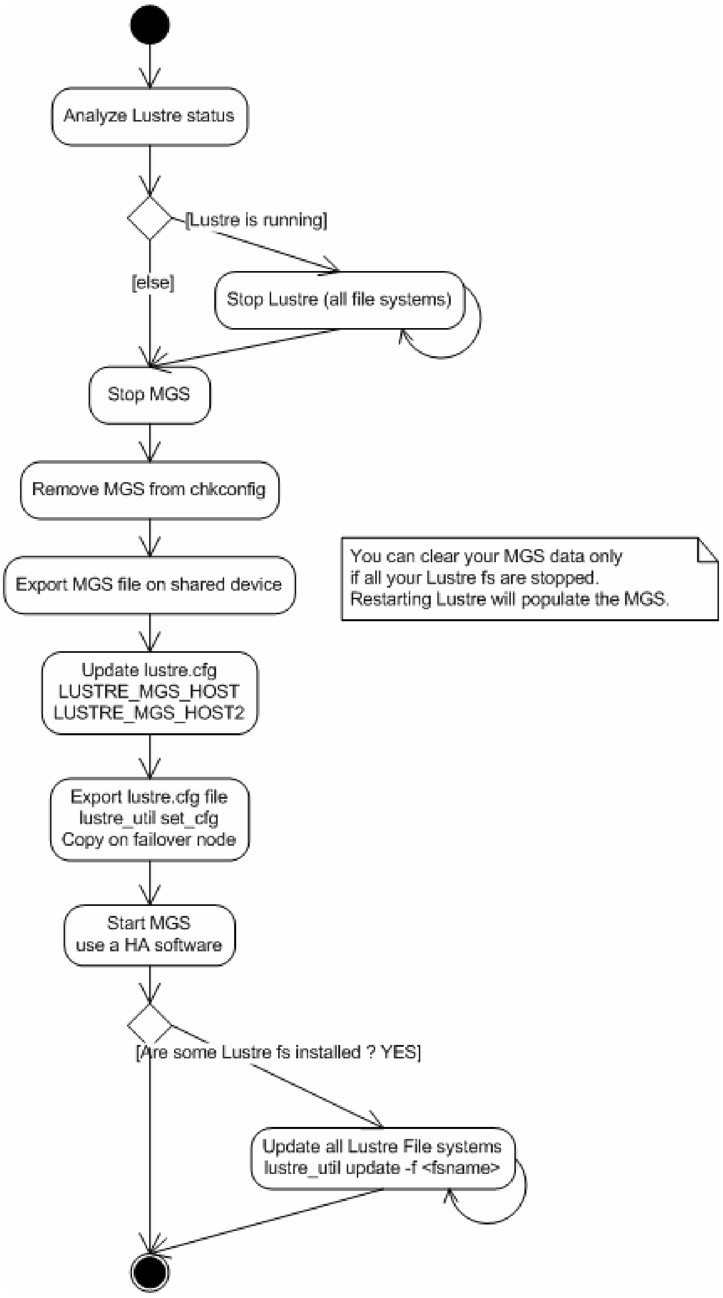

Analyze Lustre status
[Lustre is running]
[else]
Stop Lustre (all file systems)
Stop MGS
Remove MGS from chkconfig
Export MGS file on shared device
You can clear your MGS data only
if all your Lustre fs are stopped.
Restarting Lustre will populate the MGS.
Update lustre.cfg
LUSTRE_MGS_HOST
LUSTRE_MGS_HOST2
Export lustre.cfg file
lustre_util set_cfg
Copy on failover node
Start MGS
use a HA software
[Are some Lustre fs installed ? YES]
Update all Lustre File systems
lustre_util update -f <fsname>

Figure 7-2. Configuring the MGS Service on Highly Available Management Nodes

WARNING
The MGS service does not support the use of IP address aliases. Always use two different IP addresses or hostnames in the lustre.cfg file.

7.1.3 Configuring LDAP

1. Stop the **ldap** service, if it is running on the Management Node.

```
service ldap stop
```

2. Move the **ldap** data to the shared storage device : **/var/lib/ldap/lustre**

3. Add the **ldap** service to Management Node High Availability software.

4. Change the **ldap** URL in the **/etc/lustre/lustre.cfg** file and provide an alias IP address for both Management Nodes.

5. Restart the **ldap** service using the High Availability software.

```
service ldap start
```

7.1.4 Configuring lustredbd

1. Stop the **lustredbd** service, if it is running on the Management Node.

```
service lustredbd.sh stop
```

2. The **lustredbd** data is contained in the Cluster DB and does not need to be moved.

3. Change the **lustredbd** URL in the **/etc/lustre/lustre.cfg** file and provide an alias IP address for both Management Nodes.

4. Restart the **lustredbd** service using the High Availability software.

```
service lustredbd.sh start
```

Chapter 8. NFSv3 High Availability for I/O Nodes

Important

NFS High Availability for I/O nodes is not supported for all Bull Advanced Server (BAS) system releases.

See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

Important

The High Availability solution described in this chapter is supported for NFS V3 ONLY.

8.1 Hardware Architecture

Bull High Availability management of the NFS system requires a specific hardware architecture.

The I/O nodes operating in High Availability mode are grouped in an I/O cell bringing together two I/O nodes that access the disk array. The two I/O nodes are connected to separate switches, one for the BMC network, and one for the heartbeat Ethernet network, as shown in *Figure 6.1*.

Usually, nodes are connected directly to the storage systems ports, without intermediate switches or HUBs. This point to point linking avoids additional active components which increase the risk of system failure by also being SPOFs (Single Point of Failure).

The LUNs within the storage systems are accessible for both nodes of the I/O cell. But each LUN must be used by only one node at a time to avoid data corruption.

An I/O fencing mechanism is implemented so that the faulty node cannot access the LUNs following a failure and switch in active node within an I/O cell. In addition a node which fails is powered off.

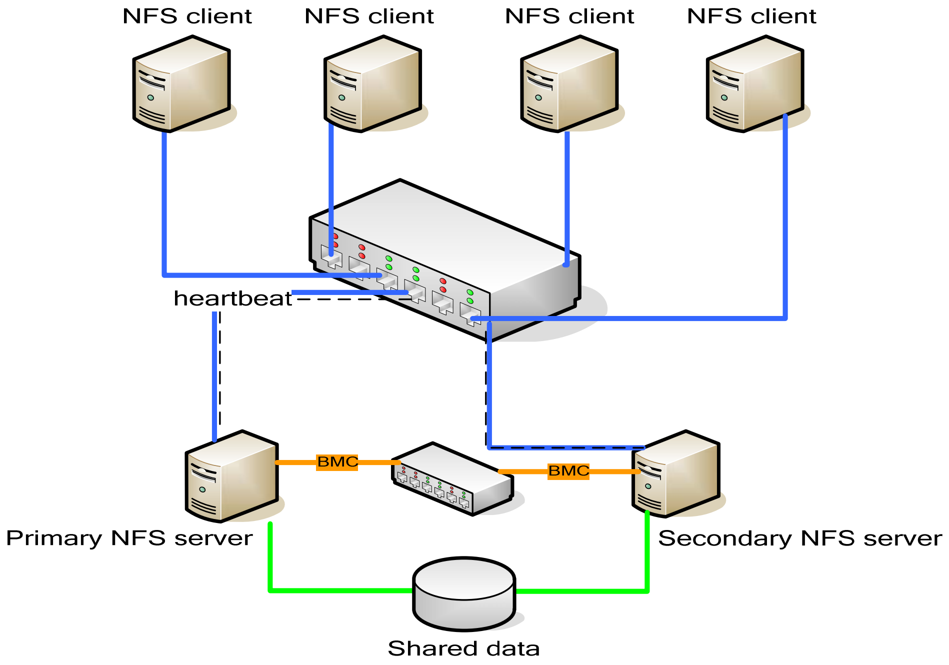


Figure 8-1. I/O Cell

An **active/passive** High Availability configuration is implemented for **NFS** services. This means that only the Primary Node is used for the NFS services, the virtual IP address and for mounting the file systems. If the Primary Node/Server fails, the Secondary Node/Server takes over automatically. The transition will be seamless and the client will not be aware of the failover.

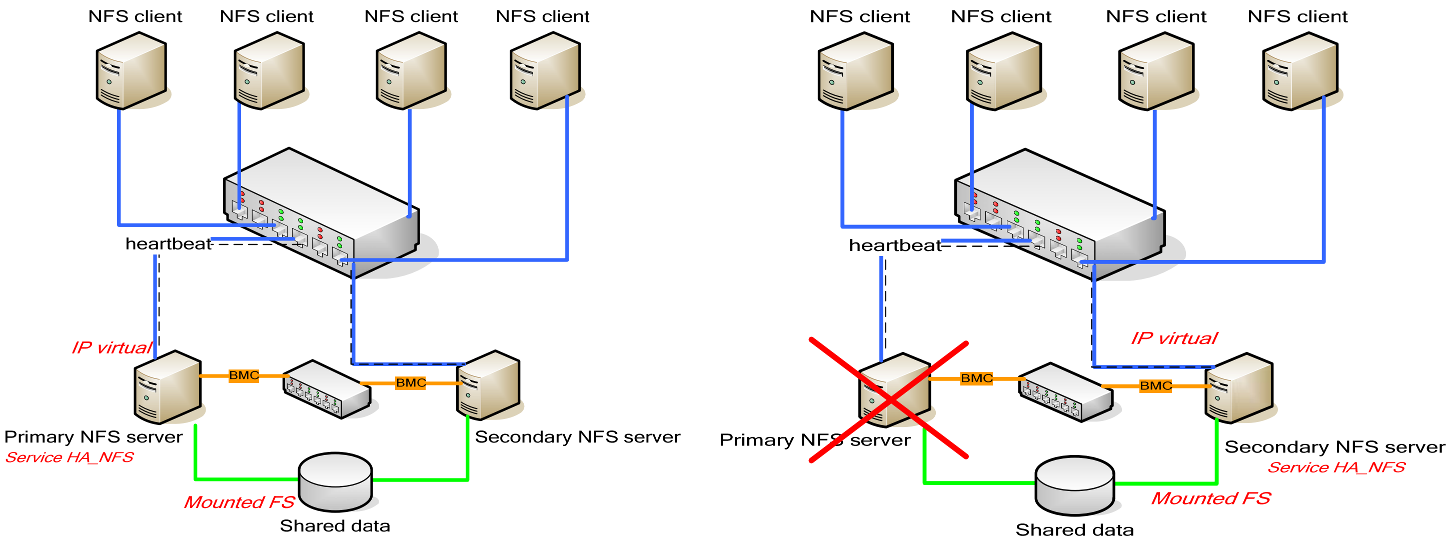


Figure 8-2. Active/Passive architecture for NFS services

Cluster Suite requires a service script to stop, start and report the status of each failover service that it manages. The failover script for the NFS service is **/usr/sbin/hanfs**.

8.2 Error Detection and Prevention Mechanisms

NFS Single Point of Failure (SPOF) tracking

Tracked NFS service SPOFs include:

1. A service crash when NFS stops running on the node.
2. Mounted points no longer available
3. Unconfigured virtual IP address
4. I/O errors on the back-end device

A regular check of **NFS** services is scheduled by **Cluster Suite** by the failover script, and the results are returned to the status indicator.

Cluster Suite manages and monitors the mount points and the virtual IP address associated with the **HA_NFS** service.

The detection of I/O errors on the back-end device relies on the **storfilter** storage management monitoring daemon. When it detects a problem, this daemon triggers the NFS failover script.

8.3 NFSv3 Failures covered by High Availability

8.3.1 I/O Node Failures

I/O Node Panic/Hang

When a Primary Node hangs or panics, the heartbeat message will not be sent within the authorized period. This silence is detected by the Secondary HA node which fences the silent node and then takes over the cluster services.

I/O Node Power down

The Primary Node does not send its heartbeat messages within the authorized period. This silence is detected by the Secondary HA node which fences the silent node, and then takes over the cluster services.

NFS Software Failure

The status parameter of the **HA_NFS** failover script detects that the NFS service is missing or not in the correct state. It will firstly try to restart the service, if there is no response it will reboot the node where the service is running.

8.3.2 Storage Failures

Fibre Channel Adapter Errors

Fibre channel adapters are used to access the external storage systems, shared by the two nodes in the HA I/O cell. Fibre channel adapter errors are ignored. Fibre Channel Link errors are usually transient and do not lead to a node failover. If the adapter error is significant, it leads to a Linux disk error which triggers an alert. In this situation the Primary Node will fail and **Cluster Suite** will switch the service to the Secondary node.

Disk Subsystem Controller Failure

The node detects a disk error when a disk controller fails. Disk errors are monitored by the **I/O status** service which alerts the I/O nodes. In this situation the Primary Node will fail and **Cluster Suite** will switch the service to the Secondary node.

Note   If the second disk controller fails it will not be possible to recover the system.

Local Disk errors

When a local disk fails **Cluster Suite** will switch to the Secondary Node.

8.3.3 Ethernet Network Failures

Note   Only the heartbeat network is monitored by **Cluster Suite**. Any other Ethernet network failures will not lead to service takeover; however these failures will be displayed on the Management Node via **NovaScale Master - HPC Edition**.

BMC Ethernet Network Access Failure for the switch or link

The BMC management network is also used to send fence requests. The node which is unable to use the BMC management network cannot fence the other node in the High Availability pair. Thus, the second node which can access the Management network wins the fencing race, and takes over the cluster services. There is no risk of split brain (i.e. both nodes within the I/O cell simultaneously mounting the same device).

Ethernet Heartbeat Network (management or backbone) failure for the switch or link

A failure of the Ethernet heartbeat network stops heartbeat exchanges and results in each node in the I/O cell attempting to take over the service. **Cluster Suite's Quorum disk** is used to prevent the nodes from initiating a fencing race.

Chapter 9. Configuring High Availability for NFSv3

Important

NFS High Availability for I/O nodes is not supported for all Bull Advanced Server (BAS) system releases.

See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

Important

The High Availability solution described in this chapter is supported for NFS V3 ONLY.

Large clusters may contain multiple I/O cells, and **Cluster Suite** must be configured for each I/O cell. This process is fully automated by Bull cluster management tools. All the information necessary is taken from the Cluster DataBase and is used to generate the **Cluster Suite** configuration files. These files are pushed to all nodes that are managed by **Cluster Suite**.

Important

Cluster Suite commands not described in this section must not be used, as they may lead to fatal inconsistency for NFS. The GUI must not be used as well. The Cluster Suite setup is predefined to enable the failover process to work smoothly, and prevent the risk of split brain (i.e. both nodes of the I/O cell simultaneously running the same NFS service). The Administrator must not attempt to modify Cluster Suite's configuration for I/O cells.

The management tasks for **Cluster Suite** are:

- Distributing the configuration file
- Starting **Cluster Suite**.

By default, **Cluster Suite** is not automatically started at boot time, and it is recommended that this is NOT enabled.

For **NFS High Availability**, **Cluster Suite** manages the virtual IP addresses and the mount points. But the Administrator should configure the **/etc/exports** file to export a mixture of file systems, some using shared storage and some using local storage, as **Cluster Suite** will not do this.

9.1 Pre-requisites

- The following packages must be installed:
  - **HighAvailability-confignfs** on the Management Node
  - **HighAvailability-nfs** on the I/O Nodes
- The **I/O cell** for the I/O nodes must be defined in the Cluster DataBase. If this is not the case, use the **storiocellctl** command to define the I/O Cell.

Important

The I/O NFS nodes that include the High Availability functionality must be deployed using a dedicated KSIS I/O node image that strictly excludes Lustre.

9.2 Configuring Disks

To ensure persistency of storage system device names for the cluster, aliases have to be created for the devices.

If the storage system has been configured using a storage model file, then aliases are created by using the **stordepmap** command remotely from the Management Node.

If there is no model available for the storage subsystem, symbolic links must be created using the **stordiskname** command from Management Node remotely:

```
stordiskname -c -r <node1_name1>,<node2_name>
```

Then, for each node:

```
ssh <node_name> "stormap -c"
```

Check and retrieve symbolic link name for each node:

```
ssh <node_name> "stormap -l"
```

Note The Quorum disk must have been defined in the model file in order that a **LUN** is dedicated to it. If a model file is not used and **stordiskname** has been used as an alternative, then a LUN must be configured as the Quorum disk (see **mkqdisk** help for more information).

9.3 Configuring File Systems

The **ext3** or **xfs** file systems must be created using the appropriate commands (**parted**, **mkfs**) on the primary node. These file system must not defined in the **/etc/fstab** file because the file systems in this file are managed by the **Cluster Suite**. But both nodes within the I/O cell must export the **ext3** and **xfs** file systems.

9.4 Configuring Cluster Suite

1. Configure NFS High Availability on both the NFS I/O nodes automatically from the remote Management Node by editing the **/etc/storageadmin/haionfs.conf** file.

   In this file both the devices and mount points must be specified, for example:

   ```
   /dev/ldn.da0.1 /mount_point1 xfs
   ```

Note The **/mount_point1** directory must have been created previously on all the nodes.

2. Choose the IP interface address aliases to be managed by **Cluster Suite**. The interfaces include the admin interface and possibly the backbone and/or **InfiniBand** interface. These IP address aliases are used to ensure that any server breakdown is transparent on the NFS client side.

   On the Management Node edit the **/etc/hosts-tpl/** file and add the IP address aliases, for example:

   ```
   xxx.xxx.1.76    xenanfs
   xxx.xxx.xxx.135    xenanfs_BK
   xxx.xxx.1.76    xenanfs_ic0
   ```

3. Run the following command to generate the new **/etc/hosts** file:

   ```
   dbmConfig configure --service syshosts --force
   ```

4. Deploy the **/etc/hosts** file on to the **I/O** and **NFS** client nodes:

   ```
   pdcp -w basename[x,y] /etc/hosts /etc/hosts
   ```

5. Run the **stordepha.nfs** command using the correct parameters, for example:

   ```
   stordepha.nfs -c configure –q -i xena1,xena11
   –I xxx.xxx.1.76/16,xxx.xxx.1.76/16,xxx.xxx.xxx.135/24
   ```

Note The **-i** option above refers to the NFS nodes and the `-I` option refers to the Alias IP addresses.

Important

The netmask address must be the same as the primary IP address defined for the interface.

6. It is possible to define the heartbeat network on the backbone network, as opposed to the administration network. In this case, and assuming the I/O nodes are connected to the backbone network, **stordepha.nfs** must be used with the **-hb** option, as shown below:

```
stordepha.nfs -c configure –q -i xena1,xena11
–I xxx.xxx.1.76/16,xxx.xxx.1.76/16,xxx.xxx.xxx.135/24 –hb backbone
```

See The **stordpha.nfs – help** file for a description of the different parameters for the command.

7. The configuration file for the Cluster Suite (**/etc/cluster/cluster.conf**) is automatically generated and distributed on both nodes within an I/O cell.

Restoring a node

After restoring the system on a node, the **cluster.conf** file has to be rebuilt using the **stordepha.nfs** command, as described above.

9.4.1 Starting Cluster Suite

1. To start **Cluster Suite** on the I/O nodes, run the command below from the Management Node remotely:

```
stordepha.nfs -c start -i <node1_name>,<node2_name>
```

Or, locally on the node, run the command below:

```
storioha -c start
```

2. To check **Cluster Suite's** status, run the command below from the Management Node remotely:

```
stordepha.nfs –c status –i <node1_name>,<node2_name>
```

3. Activate the **nfs** service with the command below on **ONLY ONE** of the I/O nodes:

```
ssh <node_name> "clusvcadm -e nfs_service"
```

Important

If an I/O node is fenced, Cluster Suite must be restarted after the reboot manually.

9.4.2 Configuring NFS Servers

Cluster Suite manages the IP address aliases, the mount points, and the **NFS** service.

Therefore, on both I/O target nodes, the **HighAvailability-nfs** package installation disables the **NFS** service from being activated during the boot sequence.

For a NFS active/passive HA configuration, it is necessary to define the mount points to be exported to each target I/O node, for example, on each NFS server:

1. Edit the **/etc/exports** file:

```
/mount_point1 *(rw,no_root_squash,sync,fsid=7)
```

Note The **fsid** number varies according to the file system

2. Run the command below so that the NFS High Availability is configured for the I/O cell:

```
exportfs -va
```

9.4.3 Configuring NFS Clients

NFS clients must mount the **NFS** file system using an administration IP address alias and without any special options.

For example, add the mount point for the **/etc/fstab** file :

```
xenanfs:/mount_point1 /mount_point1 nfs defaults 0 0
```

Note **/tstha** must have been created previously on the target node and mounted using the **mount -a** option.

9.5 Monitoring NFS High Availability

System Log Files

Cluster Suite log events in the **/var/log/messages** and **/var/log/syslog** files on each I/O NFS node. These files are centralized on the Management Node by the **syslog-ng** service.

Checking Cluster Suite settings

Once **Cluster Suite** activates the **NFS** service the settings can be checked using the commands below:

- To check the mount points

```
df
```

- To check the alias IP addresses

```
ip addr show
```

- To check the status/localization of the NFS service

```
clustat
```

Chapter 10. High Availability for Resources and Jobs

Important

Resource Manager High Availability varies according to the system.
See the System Release Bulletin delivered with your BAS version for details of any limitations which may apply to your release.

10.1 PBS Professional High Availability

PBS Professional High Availability is only possible on a cluster which includes High Availability for the Management Node.

See *Chapter 3* in the **BAS5 for Xeon** *Installation and Configuration Guide* for details on installing the **FLEXlm** license sever and **PBS Professional**.

Pre-requisites

The following pre-requisites should be in place:

- A Secondary Management Node for the Management Node High Availability solution
- An NFS Server, this can either be on the Secondary Management Node or on a different Service Node, Login or otherwise.
- 3 **FLEXlm** license servers - one on the Primary and Secondary Management Nodes, and one the Service Node, Login or otherwise.

10.1.1 Hardware Architecture for PBS Professional High Availability

The cluster architecture, and the location of the **FLEXlm** license servers, varies according to the size of the cluster, as shown in the following diagrams.

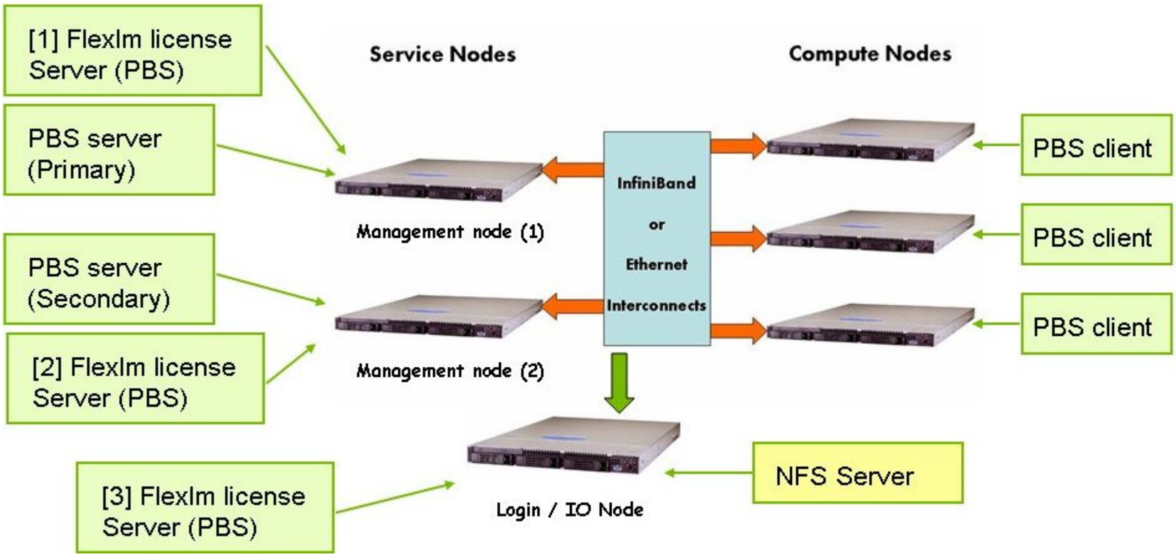


Figure 10-1. An example of the cluster Architecture for a medium sized cluster with a **FLEXlm** license server and **NFS** Server on the same Login/IO Service Node

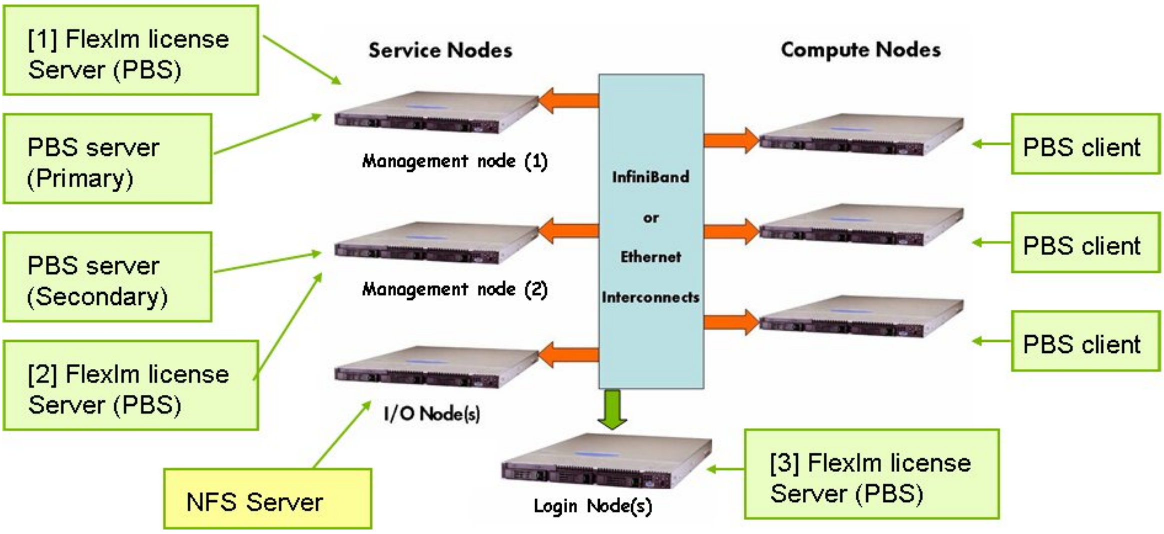


Figure 10-2. An example of a cluster architecture for a large sized cluster with an NFS Server on a dedicated I/O Node and a FLEXlm license server on a dedicated Login Node

10.1.2 Configuring High Availability for PBS Professional

Important

Before configuring High Availability for PBS Professional, the BAS software has to be installed and correctly configured on the cluster.

1. Install the **FLEXlm** license on each one of the nodes designated as a license server.
2. Install **PBS Professional** on the Primary and Secondary Management Nodes and on the PBS Professional Reference Nodes.

See *Chapter 3* in **BAS5 for Xeon** *Installation and Configuration Guide* for details on how to install the **FLEXlm** licenses and **PBS Professional**.

3. Edit the **PBS Professional** Configuration file on the two Management Nodes, as shown below:

Primary Management Node

```
cat /etc/pbs.conf

   PBS_EXEC=/usr/pbs
   PBS_HOME=/var/spool/PBS
   PBS_START_SERVER=1
   PBS_START_MOM=0
   PBS_START_SCHED=1
   PBS_SERVER=<hostname 1>
   PBS_PRIMARY=<hostname 1>
   PBS_SECONDARY=<hostname 2>
   PBS_SCP=/usr/bin/scp
```

Secondary Management Node

```
cat /etc/pbs.conf

   PBS_EXEC=/usr/pbs
   PBS_HOME=/var/spool/PBS
   PBS_START_SERVER=0
   PBS_START_MOM=0
   PBS_START_SCHED=1
   PBS_SERVER=<hostname 1>
   PBS_PRIMARY=<hostname 1>
   PBS_SECONDARY=<hostname 2>
   PBS_SCP=/usr/bin/scp
```

4. Edit the **PBS Professional** Configuration file on the COMPUTE(X) Reference Nodes, as shown below:

COMPUTE(X) Reference Node

```
cat /etc/pbs.conf

   PBS_EXEC=/usr/pbs
   PBS_HOME=/var/spool/PBS
   PBS_START_SERVER=0
   PBS_START_MOM=1
   PBS_START_SCHED=0
   PBS_SERVER=<hostname 1>
   PBS_PRIMARY=<hostname 1>
   PBS_SECONDARY=<hostname 2>
   PBS_SCP=/usr/bin/scp
```

5. Edit the **PBS Professional** Configuration file on the Login Reference Nodes, as shown below:

LOGIN Reference Node

```
cat /etc/pbs.conf

   PBS_EXEC=/usr/pbs
   PBS_HOME=/var/spool/PBS
   PBS_START_SERVER=0
   PBS_START_MOM=0
   PBS_START_SCHED=0
   PBS_SERVER=<hostname 1>
   PBS_PRIMARY=<hostname 1>
   PBS_SECONDARY=<hostname 2>
   PBS_SCP=/usr/bin/scp
```

10.2 LSF High Availability

10.2.1 Configuring OCFS2 (for SLURM and LSF/SLURM coupling)

A Cluster File System allows all nodes in a cluster to access a device concurrently via the standard file system interface. This enables that applications that run across a cluster can be managed easily; this is essential to guarantee the functionality of High Availability.

There are two tools packages: **ocfs2-tools**, which includes the command line tools and **ocfs2console**, which includes the GUI front -end for the tools.

Before using **ocfs2-tools**, you must:

1. Create a partition using the **parted** command.

2. Create the mount directory on each Management Node.

3. Run the **mkfs** command on each Management Node.

Then configure an **OCFS2** Cluster File System (`/ocfs2shared` for example), as described in the OCFS2 User's Guide available at:

http://oss.oracle.com/projects/ocfs2/documentation/

10.2.2 Configuring SLURM (active/active system not managed by the Cluster Suite)

To use the native failover mechanism, we must define a **SLURM** backup controller.

The **SLURM** controller, **slurmctld**, will periodically save the status of the job into a directory so that the status may be recovered after a fatal system error. When using a backup controller, the file system in which this directory resides should be shared between the Primary Node (ControlMachine) and Back-up Node (BackupController). The location where job accounting logs are to be written must also be defined in the shared directory.

Below is a **slurm.conf** configuration file example (**SLURM v1.0.15**) that uses the **/ocfs2shared/slurmdata** shared file system:

```
ControlMachine=ns0
ControlAddr=10.1.0.1
BackupController=bali1
BackupAddr=10.1.0.13
StateSaveLocation=/ocfs2shared/slurmdata
JobAcctType=jobacct/log
JobAcctLoc=/ocfs2shared/slurmdata/slurm_accounting.log
```

Change the owner properties for the **/ocfs2shared/slurmdata** file system with the command:

```
chown -R slurm:root /ocfs2shared/slurmdata
```

10.2.3 Configuring LSF

- Clusters with **LSF/RMS** coupling or LSF alone:

  In this case, LSF is fully managed by the **Cluster Suite** (LSF service stopping, starting and **/usr/share/lsf** mounting). The LSF installation is standard.

- Clusters with **LSF/SLURM**:

  In this case the **Cluster Suite** manages **LSF** service stopping and starting, but not the shared resource. So LSF must be installed on the **ocfs2** file system that has been created for **SLURM**. Then the **LSF** directories must be symbolically linked to this file system:

```
ln -sf /ocfs2shared/lsf /usr/share/lsf
ln -sf /ocfs2shared/hptc_cluster /hptc_cluster
```

In both cases it is necessary to define the virtual name of the active Management Node as an alias in a file specific to LSF:

In the **$LSF_ENVDIR** directory create a hosts file as follows:

```
cat $LSF_ENVDIR/hosts
10.1.0.65 ns ns0 ns13
10.1.0.1 ns ns0
10.1.0.13 ns ns13
```

10.3 RMS High Availability

10.3.1 Configuring RMS

Usually the RMS partitions are not in **auto-start** mode. When a node rocks to another node, there is no existing High Availability mechanism to restart automatically the partitions. To ensure that the job submitted continues to be treated, it is recommended to activate the **auto-start** mode on the RMS partitions that the administrator wants to be used for the job.

Please refer to the RMS documentation for more information about using **auto-start** mode.

Create an **rmshost-if** attribute initialized with the virtual IP address:

```
rcontrol create attribute rmshost-if val '10.0.0.65'
```

Increase the **reconnect-timeout** attribute value:

```
rcontrol set attribute reconnect-timeout val 480
```

Glossary and Acronyms

A

ACL

Access Control List.

B

Bisectional Bandwidth

The bandwidth flowing through a fabric while half the nodes send and receive a full duplex stream of data to the other half of the nodes.

C

CGI

Common Gateway Interface.

ConMan

A management tool, based on telnet, enabling access to all the consoles of the cluster.

Cron

A UNIX command for scheduling jobs to be executed sometime in the future. A cron is normally used to schedule a job that is executed periodically - for example, to send out a notice every morning. It is also a daemon process, meaning that it runs continuously, waiting for specific events to occur.

Cygwin

A Linux-like environment for Windows. The Bull cluster management tools use Cygwin to provide ssh support on a Windows system, enabling access in command mode from the Cluster management system.

D

DNS

Domain Name Server. A server that retains the addresses and routing information for TCP/IP LAN users.

G

Ganglia

A distributed monitoring tool used to view information associated with a node, such as CPU load, memory consumption, network load.

GID

Group ID.

GPT

GUID Partition Table.

H

HBA

Host Bus Adapter.

Hyper-Threading

Hyper-Threading technology is an innovative design from Intel that enables multi-threaded software applications to process threads in parallel within each processor resulting in increased utilization of processor execution resources. To make it short, it is to place two logical processors into a single CPU die.

HPC

High Performance Computing.

K

KSIS

Utility for image building and development.

L

LDAP

Lightweight Directory Access Protocol.

LDIF

The **LDAP Data Interchange Format** is a standard plain text data interchange format to represent LDAP directory contents and update requests. LDIF conveys directory content as a set of records, one record for each object (or entry). It represents update requests, such as Add, Modify, Delete, and Rename, as a set of records, one record for each update request.

LKCD

Linux Kernel Crash Dump. A tool capturing and analyzing crash dumps.

LOV

Logical Object Volume.

LVM

Logical Volume Manager.

M

MIB

Management Information Base.

MDS

MetaData Server.

MDT

MetaData Target.

MkCDrec

Make CD-ROM Recovery. A tool making bootable system images.

MPI

Message Passing interface.

MTBF

Mean Time Between Failures.

N

Nagios

A powerful monitoring tool, used to monitor the services and resources of Bull HPC clusters.

NFS

Network File System.

NIC

Network Interface Card.

NTP

Network Time Protocol.

O

OpenSSH

Open Source implementation of the SSH protocol.

OSC

Object Storage Client.

OSS

Object Storage Server.

OST

Object Storage Targets.

P

PDSH

A parallel distributed shell.

R

RMS

Resource Management System. Manages the cluster resources.

S

SAN

Storage Area Network.

SIS

System Installation Suite.

SLURM

Simple Linux Utility for Resource Management.

SPOF

Single Point of Failure

SSH

Secure Shell. A protocol for creating a secure connection between two systems.

Syslog-ng

Syslog New Generation, a powerful system log manager.

T

U

UID

User ID

V

VNC

Virtual Network Computing. It is used to enable access to Windows systems and Windows applications from the Bull NovaScale cluster management system.

W

WWPN

World Wide Port Name- a unique identifier in a Fibre Channel SAN.

X

XFS

eXtended File System.

XHPC

Xeon High Performance Computing

XIB

Xeon InfiniBand

Index

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

REFERENCE
86 A2 21FA 00