



Red Hat Reference Architecture Series

Deploying a Highly Available Web Server on Red Hat Enterprise Linux 6

Mark Heslin
Principal Software Engineer

Version 1.0
August 2011





1801 Varsity Drive™
Raleigh NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

Linux is a registered trademark of Linus Torvalds. Red Hat, Red Hat Enterprise Linux and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group.

Intel, the Intel logo and Xeon are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

© 2011 by Red Hat, Inc. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

The information contained herein is subject to change without notice. Red Hat, Inc. shall not be liable for technical or editorial errors or omissions contained herein.

Distribution of modified versions of this document is prohibited without the explicit permission of Red Hat Inc.

Distribution of this work or derivative of this work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from Red Hat Inc.

The GPG fingerprint of the security@redhat.com key is:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Send feedback to refarch-feedback@redhat.com



Table of Contents

1 Executive Summary.....	1
2 Component Overview.....	2
2.1 Red Hat Enterprise Linux 6.....	2
2.2 Red Hat Network Satellite Server.....	3
2.3 High Availability Add-On.....	4
2.3.1 Quorum.....	4
2.3.2 Resource Group Manager.....	4
2.3.3 Fencing.....	5
2.3.3.1 IPMI.....	6
2.3.4 Failover Domains.....	7
2.3.5 Conga.....	8
2.3.5.1 Luci.....	8
2.3.5.2 Ricci.....	8
2.3.6 CCS.....	9
2.4 HA-LVM Volume.....	10
2.4.1 Logical Volume Manager.....	10
2.4.2 File System.....	10
2.4.3 DM Multipath.....	11
2.5 Cluster Web Service.....	12
3 Reference Architecture Configuration.....	13
3.1 Cluster Management Server.....	14
3.2 Cluster Server - Node 1.....	14
3.3 Cluster Server - Node 2.....	15
3.4 Cluster Server - Node 3.....	15
3.5 Fibre Channel Storage Array.....	16
4 Cluster Deployment.....	17
4.1 Deployment Task Flow.....	17
4.2 Deploy Management Server	18
4.2.1 Install Red Hat Enterprise Linux 6.....	18
4.2.2 Configure Networks.....	19



4.2.3 Configure Firewall.....	20
4.2.4 Install Cluster Management Software.....	21
4.3 Deploy Cluster Nodes.....	23
4.3.1 Install Red Hat Enterprise Linux 6.....	23
4.3.2 Configure Networks and Bonding.....	24
4.3.3 Configure Firewall.....	27
4.3.4 Install Cluster Node Software.....	29
4.3.5 Configure Storage.....	29
4.3.5.1 Configure Multipathing.....	29
4.3.5.2 Create HA-LVM Volume.....	31
4.3.6 Configure Web Server.....	33
4.3.7 Configure SELinux Security Parameters.....	34
4.4 Cluster Creation via Conga.....	36
4.4.1 Create Cluster.....	36
4.4.2 Add Fence Devices.....	39
4.4.3 Add Failover Domain.....	44
4.4.4 Add Resources.....	46
4.4.5 Add Service Group.....	49
4.4.6 Verify Cluster Web Service.....	51
4.5 Cluster Creation via CCS.....	52
4.5.1 Create Cluster.....	52
4.5.2 Add Nodes.....	53
4.5.3 Add Fence Devices.....	53
4.5.4 Add Failover Domain.....	54
4.5.5 Add Resources.....	54
4.5.6 Add Service Group.....	55
4.5.7 Activate Cluster.....	55
4.5.8 Verify Cluster Web Service.....	56
5 Cluster Management.....	57
5.1 Adding Cluster Nodes.....	57
5.2 Removing Cluster Nodes.....	61
5.3 Relocating Cluster Web Services.....	65
5.4 Fencing Cluster Nodes.....	69
5.5 Importing a Cluster.....	71
6 Conclusion.....	73
Appendix A: References.....	74



Appendix B: Red Hat Enterprise Linux 6 – Satellite Configuration Details.....	75
Appendix C: Fibre Channel Storage Provisioning.....	85
Appendix D: Cluster Configuration File (cluster.conf).....	89
Appendix E: HA-Web Status Script.....	91
Appendix F: Cluster Configuration Matrix.....	93
Appendix G: Deployment Checklists.....	94



1 Executive Summary

Red Hat's High Availability Add-On for Red Hat Enterprise Linux 6 is Red Hat's premier high availability clustering solution. The High Availability Add-On provides reliability, availability and scalability (RAS) to critical production services by eliminating single points of failure and providing automatic *failover* of those services in the event of a cluster node failure or error condition.

This reference architecture details the deployment, configuration and management of a highly available web service using the High Availability Add-On for Red Hat Enterprise Linux 6. A three node cluster is deployed to run the web service and a dedicated management station is deployed to manage and configure the cluster.

The underlying storage for the web service utilizes a highly-available LVM (HA-LVM) volume. HA-LVM volumes permit cluster service data to be shared exclusively among highly-available nodes for the least cost and highest possible performance. The HA-LVM volume within this document is created on a Fibre Channel storage array but other shared storage technologies (e.g. - *iSCSI*) may be used.

Additional redundancy and performance increases are achieved through the use of separate public and private (cluster interconnect) networks. Multiple network adapters are used on these networks with all interfaces bonded together. Similarly, device mapper multipathing is used to maximize performance and availability to the HA-LVM volume.

Two methods are available for creating and managing clusters with the High Availability Add-On for Red Hat Enterprise Linux 6. The first method is through the use of Conga and the **luci** graphical user interface (GUI). The second method is through the use of the Cluster Configuration System (CCS) command line interface (CLI). Both approaches are detailed within this reference architecture.

The most common cluster management tasks (adding, removing nodes, relocating cluster services, fencing nodes, importing an existing cluster) are also demonstrated.

This document does not require extensive Red Hat Enterprise Linux experience but the reader is expected to have a working knowledge of Linux administration and clustering concepts.



2 Component Overview

This section provides an overview on the Red Hat Enterprise Linux operating system, Red Hat's High Availability Add-On and the other components used in this reference architecture.

2.1 Red Hat Enterprise Linux 6

Red Hat Enterprise Linux 6.1, the latest release of Red Hat's trusted datacenter platform, delivers advances in application performance, scalability, and security. With Red Hat Enterprise Linux 6.1, physical, virtual and cloud computing resources can be deployed within the data center. Red Hat Enterprise Linux 6.1 provides the following features and capabilities:

Reliability, Availability, and Security (RAS):

- More sockets, more cores, more threads, and more memory
- RAS hardware-based hot add of CPUs and memory is enabled
- Memory pages with errors can be declared as “poisoned” and can be avoided

File Systems:

- ext4 is the default filesystem and scales to 16TB
- XFS is available as an add-on and can scale to 100TB
- Fuse allows file systems to run in user space allowing testing and development on newer fuse-based file systems (such as cloud file systems)

High Availability:

- Extends the current clustering solution to the virtual environment allowing for high availability of virtual machines and applications running inside those virtual machines
- Enables NFSv4 resource agent monitoring
- Introduction of CCS. CCS is a command line tool that allows for complete CLI administration of Red Hat's High Availability Add-On

Resource Management:

- cgroups organize system tasks so that they can be tracked and so that other system services can control the resources that cgroup tasks may consume
- cpuset applies CPU resource limits to cgroups, allowing processing performance to be allocated to tasks

There are many other feature enhancements to Red Hat Enterprise Linux 6. Please see the Red Hat website for more information.



2.2 Red Hat Network Satellite Server

This reference architecture uses Red Hat Network Satellite (RHN) server to provision the cluster nodes. All RHN functionality is on the network, allowing much greater flexibility and customization. The satellite server connects with Red Hat over the public Internet to download new content and updates. This model also allows customers to take their Red Hat Network solution completely off-line if desired. Advantages of using RHN Satellite include:

- Security - an end-to-end secure connection is maintained from the client systems to the RHN Satellite without connecting to the public Internet.
- Efficiency - packages are delivered significantly faster over a local area network.
- Control - clients' System Profiles are stored on the local RHN Satellite, not on the central Red Hat Network Servers.
- Customized updates - create a truly automated package delivery system for custom software packages required by client systems, as well as Red Hat packages. Custom channels allow fine-grained control of the delivery of custom packages.
- Access control - system administrators can be restricted to access only those systems within their maintenance responsibilities.
- Bandwidth management - the bandwidth used for transactions between the clients and the RHN Satellite is controlled by the organization on the local area network; RHN Satellite clients do not have to compete with other clients accessing the central Red Hat Network file servers.
- Scalability - RHN Satellite may oversee an entire organization's servers in combination with RHN Proxy Server.

RHN Satellite is Red Hat's on-premises systems management solution that provides software updates, configuration management, provisioning and monitoring across both physical and virtual Red Hat Enterprise Linux servers. It offers customers opportunities to gain enhanced performance, centralized control and higher scalability for their systems, while deployed on a management server located inside the customer's data center and firewall.



2.3 High Availability Add-On

The High Availability Add-On for Red Hat Enterprise Linux provides high availability of services by eliminating single points of failure. By offering failover services between nodes within a cluster, the High Availability Add-On supports high availability for up to 16 nodes. (Currently this capability is limited to a single LAN or datacenter located within one physical site.)

The High Availability Add-On also enables failover for off-the-shelf applications such as Apache, MySQL, and PostgreSQL, any of which can be coupled with resources like IP addresses and single-node file systems to form highly available services. The High Availability Add-On can also be easily extended to any user-specified application that is controlled by an init script per UNIX System V (SysV) standards.

When using the High Availability Add-On, a highly available service can fail over from one node to another with no apparent interruption to cluster clients. The High Availability Add-On also ensures absolute data integrity when one cluster node takes over control of a service from another cluster node. It achieves this by promptly evicting nodes from the cluster that are deemed to be faulty using a method called "fencing", thus preventing data corruption. The High Availability Add-On supports several types of fencing, including both power and storage area network (SAN) based fencing.

The following sections describe the various components of the High Availability Add-On in the context of this reference architecture.

2.3.1 Quorum

Quorum is a voting algorithm used by the cluster manager (CMAN). CMAN manages cluster quorum and cluster membership. CMAN runs as a service on all the cluster nodes. To maintain *quorum*, the nodes in the cluster must agree about their status among themselves. The *quorum* determines which nodes in the cluster are dominant. For example, if there are three nodes in a cluster and one node loses connectivity, the other two nodes communicate with each other and determine that the third node needs to be fenced. The action of fencing ensures that the node which lost connectivity does not corrupt data.

By default each node in the cluster has one *quorum* vote, although this is configurable. There are two methods the nodes can communicate with each other to determine *quorum*. The first method *quorum* via network consists of a simple majority (50% of the nodes +1 extra). The second method is by adding a *quorum* disk. The *quorum* disk allows for user-specified conditions to exist which help determine which node(s) should be dominant.

This reference architecture uses network *quorum* - a dedicated *quorum* disk is not required.

2.3.2 Resource Group Manager

The resource group manager (*rgmanager*) provides failover capabilities for collections of cluster resources known as resource groups or resource trees. *Rgmanager* works by allowing systems administrators to define, configure, and monitor cluster services. In the event of a node failure, *rgmanager* relocates the clustered service to another node to restore service



availability. Services can be restricted to certain nodes, such as restricting *httpd* to one set of nodes while *mysql* can be restricted to a separate set of nodes.

The following list summarizes the various processes and agents that constitute *rgmanager*:

- Failover Domains - An ordered subset of members to which a service may be bound
- Service Policies - *rgmanager*'s service startup and recovery policies
- Resource Trees - Representations of resources, their attributes, parent / child and sibling relationships
- Resource Agents

rgmanager runs as a service on all the nodes in a cluster. If the service is not running, the resources are not available to be brought online. Recovery of *rgmanager* depends on the Distributed Lock Manager (DLM). In the event of a failure, the DLM must recover prior to *rgmanager* recovering services from a failed host.

2.3.3 Fencing

Fencing is the disconnection of a node from the cluster's shared storage. Fencing prevents the affected node from issuing I/O to shared storage, thus ensuring data integrity. The cluster infrastructure performs fencing through *fenced*, the fence daemon.

When CMAN determines that a node has failed, it communicates to other cluster-infrastructure components to inform them that the node has failed. The failed node is fenced when *fenced* is notified. Other cluster-infrastructure components determine what actions to take - that is, they perform any recovery that needs to be done. For example, distributed lock manager (*DLM*) and Global File System version 2 (*GFS2*), when notified of a node failure, suspend activity until they detect that *fenced* has completed fencing the failed node. Upon confirmation that the failed node is fenced, *DLM* and *GFS2* perform recovery. *DLM* releases locks of the failed node; *GFS2* recovers the journal of the failed node.

The fencing program (*fenced*) determines from the cluster configuration file which fencing method to use. Two key elements in the cluster configuration file define a fencing method: fencing agent and fencing device. The fencing program makes a call to a fencing agent specified in the cluster configuration file. The fencing agent, in turn, fences the node via a fencing device. When fencing is complete, the fencing program notifies the cluster manager. The High Availability Add-On provides a variety of fencing methods:

- Power fencing - A fencing method that uses a power controller to power off an inoperable node
- Storage fencing - Includes fencing methods that disable the Fibre Channel port that connects storage to an inoperable node. SCSI-3 persistent reservations are another commonly used storage fencing method in which access to a common shared storage device can be revoked to an inoperable node.
- Systems management fencing - Fencing methods that disable I/O or power to an inoperable node. Examples include IBM® BladeCenter, Dell® DRAC/MC, HP® ILO, IPMI, and IBM RSA II.



2.3.3.1 IPMI

The Intelligent Platform Management Interface (IPMI) is a standardized computer interface that allows administrators to remotely manage a system. Centered around a baseboard management controller (BMC), IPMI supports functions to access the system BIOS, display event logs, power on, power off and power cycle a system.

This reference architecture uses IPMI to fence faulty cluster nodes across the public network through the **fence_ipmilan** agent.



2.3.4 Failover Domains

A *failover domain* is an ordered subset of cluster members to which a service may be bound. *Failover domains*, while useful for cluster customization, are not required for operation.

The following is a list of semantics governing the options as to how the different configuration options affect the behavior of a *failover domain*.

Ordering, *restriction*, and *nofailback* are flags that can be combined in various ways (e.g., *ordered+restricted*, *unordered+unrestricted*, etc.). These combinations affect both where services start after initial quorum formation and which cluster members take over services in the event that the service has failed.

- Preferred node or preferred member - The *preferred node* is the member designated to run a given service if the member is online. This behavior can be emulated by specifying an unordered, unrestricted *failover domain* of exactly one member.
- Restricted domain - Services bound to the domain may only run on cluster members which are also members of the *failover domain*. If no members of the *failover domain* are available, the service is placed in the **stopped** state. In a cluster with several members, using a restricted *failover domain* can ease configuration of a cluster service (such as **httpd**), which requires identical configuration on all members that run the service. Instead of setting up the entire cluster to run the cluster service, set up only the members in the restricted *failover domain* that are associated with the cluster service.
- Unrestricted domain - The default behavior. Services bound to this domain may run on all cluster members, but run on a member of the domain whenever one is available. If a service is running outside of the domain and a member of the domain comes online, the service migrates to that member.
- Ordered domain - The order specified in the configuration dictates the order of preference of members within the domain. The highest-ranking online member of the domain hosts the service. If member A has a higher-rank than member B, the service relocates to A (if currently running on B) when A transitions from offline to online.
- Unordered domain - The default behavior. Members of the domain have no order of preference; any member may run the service. In an unordered domain, services always migrate to members of their *failover domain* whenever possible.
- Failback - Services on members of an ordered *failover domain* should relocate back to the original node once the failure is addressed. Failback can also be disabled to prevent service from failing over back and forth between frequently failing nodes.



2.3.5 Conga

Conga is an agent/server architecture for the remote administration of cluster nodes. The agent component is called **ricci** and the server component is called **luci**. One **luci** management server can communicate with **ricci** agents installed on multiple cluster nodes. When a system is added to a **luci** management server, authentication is only done the first time. No authentication is necessary afterwards. The **luci** management interface allows administrators to configure and manage cluster nodes. Communications between **luci** and **ricci** is done via XML over SSL.

2.3.5.1 Luci

Luci provides a web-based graphical user interface that helps visually administer the nodes in a cluster, manage fence devices, failover domains, resources, service groups and other cluster attributes.

Luci can create a cluster from scratch or import an existing cluster. When an existing cluster is imported, all that is required is network access to a single node that is a member of the cluster. **Luci** parses the `/etc/cluster/cluster.conf` file, locates the other nodes in the cluster and then imports them. **Luci** can manage multiple clusters as well.

Some of the improvements in the new version of **luci** include a redesigned management interface and enhanced logging capabilities. Included in the new global settings for logging are daemon specific log settings for *rgmanager*, *qdiskd*, *fenced*, *corosync*, *groupd*.

In the configuration described within this reference architecture, **luci** is run on a Red Hat Enterprise Linux 6 virtual machine. **Luci** can be installed on individual cluster nodes but for availability purposes, it is preferable to manage cluster nodes with **luci** on a non-member cluster node. **Luci** requires port 8084 to be open for communications to the **ricci** agents on all cluster member nodes.

2.3.5.2 Ricci

Ricci is the cluster management and configuration daemon that runs on the cluster nodes. When **ricci** is installed it creates a user account called **ricci** and a password is set for the account. All **ricci** accounts must be configured with the same password across all cluster nodes to allow authentication with the **luci** management server. The **ricci** daemon requires port 11111 to be open for both tcp and udp traffic.



2.3.6 CCS

The Cluster Configuration System (CCS) was introduced in Red Hat Enterprise Linux 6.1. CCS provides a powerful way of managing a Red Hat Enterprise Linux cluster from the command line. CCS allows an administrator to create, modify and view cluster configurations from a remote node through **ricci** or on a local file system. CCS has a robust man page detailing all of the options but the ones most commonly used are described in **Table 2.3.6-1: Common CCS Switches**:

Switches	Function
--host	Remote node to run ccs action on
--createcluster	Create a new cluster configuration
--addnode	Add a fence instance to the cluster
--addmethod	Add a fence method to the cluster
--addfencedev	Add a fence device to the cluster
--addfenceinst	Add a fence instance to the cluster
--addfailoverdomain	Add a failover domain to the cluster
--addfailoverdomainnode	Add a failover domain node to the cluster
--addresource	Add a resource to the cluster
--addservice	Add a service to the cluster
--addsubservice	Add a subservice to the cluster
--sync --activate	Synchronize and activate the cluster configuration file across all nodes
--checkconf	Verify all nodes have the same cluster.conf
--startall	Start cluster services on all nodes
--stopall	Stop cluster services on all nodes

Table 2.3.6-1: Common CCS Switches

Cluster configuration details are stored as XML entries in the file `/etc/cluster/cluster.conf`. To avoid configuration errors, changes to this file should always be done through the `ccs` utility.

CCS authenticates with **ricci** agents by using automatically generated certificate files in `~/.ccs/`. These files allow CCS to communicate with **ricci** securely over Secure Socket Layer (SSL) encrypted links. To communicate with **ricci** the password for the **ricci** agent on each cluster node must be known by the administrator as CCS prompts for the password at the first connection.



2.4 HA-LVM Volume

The underlying storage for the web service utilizes a highly-available LVM (HA-LVM) volume. HA-LVM volumes permit cluster service data to be shared and activated exclusively among highly-available nodes for the least cost and highest possible performance. HA-LVM volumes are ideal for active/passive (*failover*) cluster configurations where the nodes running a cluster service require exclusive access to a storage volume. When the node running the cluster service is failed-over, the HA-LVM volume is mounted by the next node allocated to resume the service.

The HA-LVM volume within this reference architecture is created on a Fibre Channel storage array but other shared storage technologies (*e.g. iSCSI*) may be used.

2.4.1 Logical Volume Manager

Volume managers create a layer of abstraction between between physical storage by presenting logical volumes that can be flexibly managed with little to no impact on the applications or services accessing them. Logical volumes can be increased in size, the underlying storage relocated to another physical device without need to unmount the file system.

The architecture of LVM consists of three components:

- Physical Volume (PV)
- Volume Group (VG)
- Logical Volume (LV)

Physical volumes (PV) are the underlying physical storage – i.e. a block device such as a whole disk or partition. A volume group (VG) is the combination of one or more physical volumes. Once a volume group has been created, logical volumes (LV) can be created from it with each logical volume formatted and mounted similar to a physical disk.

The HA-LVM volume within this reference architecture consists of one physical volume (PV) that is a member of a volume group (VG) from which a single logical volume (LV) is created. The cluster web service data is stored on the logical volume.

2.4.2 File System

Ext4 is the successor to the *Ext3* filesystem. *Ext4* is POSIX compliant and provides support for several system calls such as *read()*, *write()* and *seek()*. *Ext4* is the fourth generation of the extended filesystem family and is the default filesystem in Red Hat Enterprise Linux 6. *Ext4* has been offered in test previews since Red Hat Enterprise Linux 5.3, giving customers confidence in its maturity. *Ext4* can read and write to *Ext2* or *Ext3* filesystems, but *Ext2* or *Ext3* can not read and write to an *Ext4* filesystem. However, *Ext4* adds several new and improved features that are common with most modern filesystems, such as the following:

- Extent-based metadata
- Delayed allocation
- Journal check-summing
- Large storage support



A more compact and efficient way to track utilized space in a filesystem is the usage of extend-based metadata and the delayed allocation feature. These features improve file system performance and reduce the space consumed by metadata. Delayed allocation allows the filesystem to postpone selection of the permanent location for newly written user data until the data is flushed to disk. This enables higher performance since it can allow for larger, more contiguous allocations, allowing the filesystem to make decisions with much better information.

Additionally, filesystem check and repair time (fsck) in *Ext4* is much faster than in *Ext2* and *Ext3*. Some filesystem repairs have demonstrated up to a six-fold increase in performance. Currently, Red Hat's maximum supported size for *Ext4* is 16TB in both Red Hat Enterprise Linux 5 and Red Hat Enterprise Linux 6. Application performance depends on many variables; in addition to the actual filesystem chosen, the application performance also depends on the specific I/O pattern the application generates and the type of server and storage hardware used.

This reference architecture uses the *Ext4* filesystem on the HA-LVM volume.

2.4.3 DM Multipath

Device mapper multipathing (DM Multipath) allows multiple I/O paths to be configured between a server and the connection paths to SAN storage array volumes. The paths are aggregated and presented to the server as a single device to maximize performance and provide high availability. A daemon (**multipathd**) handles checking for path failures and status changes.

This reference architecture uses DM Multipath on the cluster web service HA-LVM volume.



2.5 Cluster Web Service

Managed by *rgmanager*, the cluster web service is configured through the High Availability Add-On during cluster creation. The cluster web service consists of the following four resources:

- IP Address (10.16.143.150)
- Logical Volume Management (HA-LVM)
- File System (Ext4)
- Script (httpd)

Client systems connect to the web service via the IP Address from a web browser. For demonstration purposes, a script (*HA-Web-Status*) is run from the */ha/ha-web/cgi-bin* directory (on the HA-LVM volume) that displays the contents of */etc/motd*, current date and time. Details on how to configure the HA-Web-Status script can be found in **Appendix E: HA-Web Status Script**.

In the event of a node failover, *rgmanager* manages the relocation of the web service. During relocation, *rgmanager* mounts the HA-LVM volume and starts the web server on the node resuming control. The process is transparent to client systems with only a brief delay as the web service is relocated to the next cluster node. From a cluster management perspective, the cluster web service plus the HA-LVM can be thought of as one logical entity as depicted in **Figure 2.5-1: Cluster Web Service**:

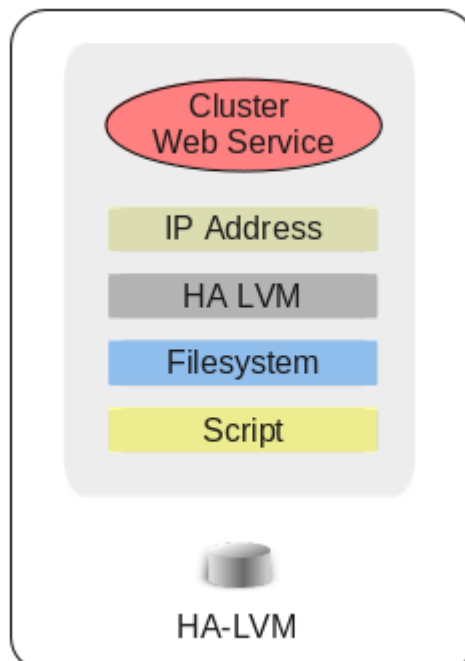


Figure 2.5-1: Cluster Web Service



3 Reference Architecture Configuration

This section provides an overview of the hardware components that were used in the deployment of this reference architecture. The cluster nodes (*ha-web1*, *ha-web2*, *ha-web3*) were configured on an HP BladeSystem c7000 enclosure using three HP ProLiant BL460c G6 Blade servers. Two 10 Gb/s ethernet networks were configured for use as the public and cluster interconnect networks. The HP Blade servers share exclusive access (one node at a time) to the HA-LVM (web service) volume located on an HP StorageWorks MSA2324fc fibrechannel storage array. During node failover or relocation of the web service, the HA-LVM volume is mounted by the node resuming control, granting exclusive access to the volume.

The cluster management server was deployed as a KVM virtual machine. The hypervisor host was configured with a bridged network to allow the virtual machine to manage cluster nodes over the cluster interconnect. Client access to the cluster web service is over the public network using a web browser.

Figure 3-1: Cluster Configuration depicts an overview of the cluster configuration:

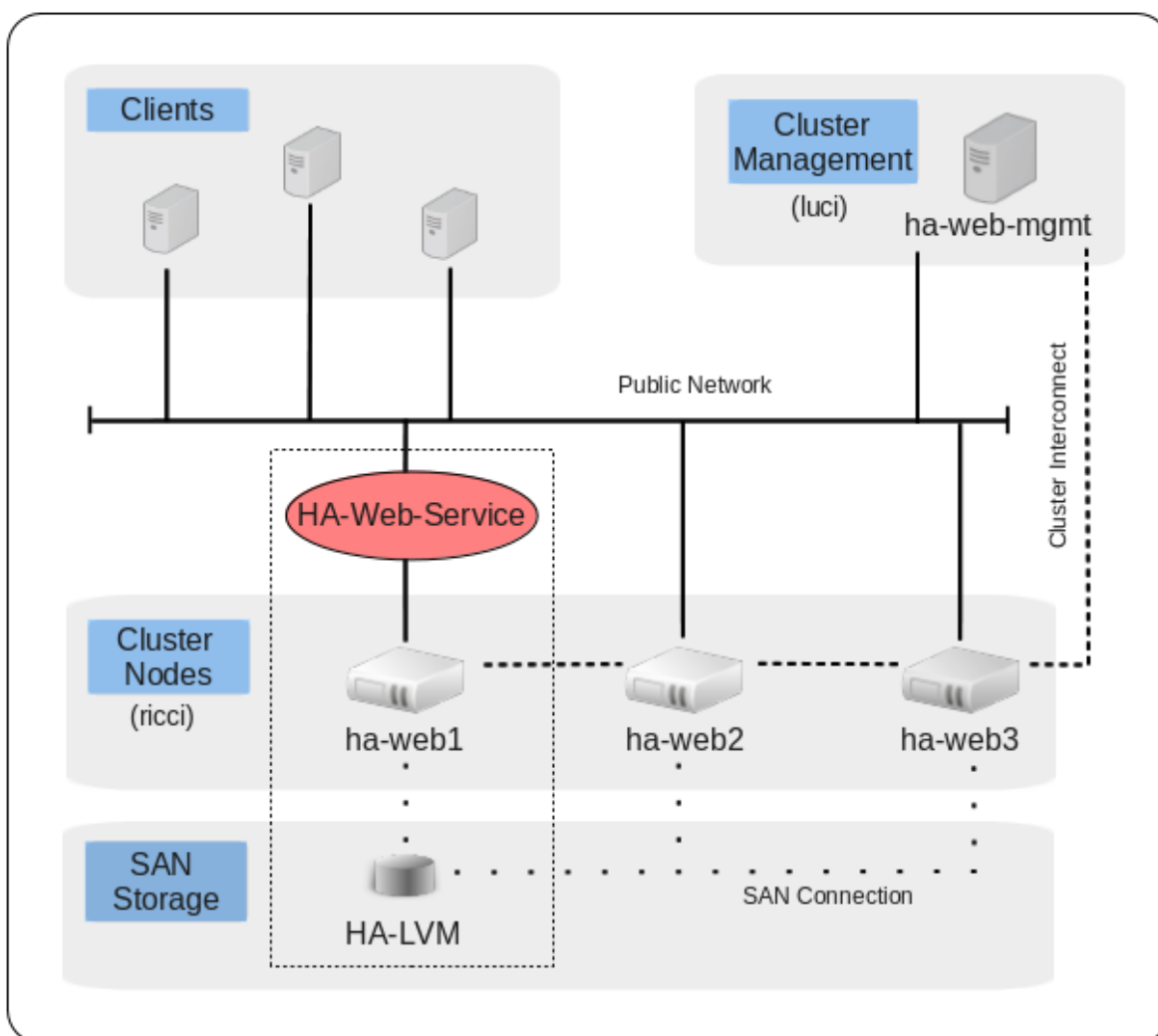


Figure 3-1: Cluster Configuration



3.1 Cluster Management Server

Component	Detail
Hostname	<i>ha-web-mgmt</i>
Operating System	Red Hat Enterprise Linux 6.1 (64-bit) (2.6.32-131.0.15.el6.x86_64 kernel)
System Type	Virtual Machine (KVM)
Processor	2 Core
Memory	4 GB
Storage	30Gb
Network	2

Table 3.1: Cluster Management Server Configuration

3.2 Cluster Server - Node 1

Component	Detail
Hostname	<i>ha-web1</i>
Operating System	Red Hat Enterprise Linux 6.1 (64-bit) (2.6.32-131.4.1.el6.x86_64 kernel)
System Type	HP ProLiant BL460c G6
Processor	Quad Socket, Quad Core (16 cores) Intel® Xeon® CPU X5550 @2.67GHz
Memory	48 GB
Storage	4 x 146 GB SATA internal disk drive (RAID 1) 2 x Qlogic QMH2562 8Gb FC HBA 1 x 20GB
Network	8 x Broadcom NetXtreme II BCM57711E XGb

Table 3.2: Cluster Node1 Configuration



3.3 Cluster Server - Node 2

Component	Detail
Hostname	<i>ha-web2</i>
Operating System	Red Hat Enterprise Linux 6.1 (64-bit) (2.6.32-131.4.1.el6.x86_64 kernel)
System Type	HP ProLiant BL460c G6
Processor	Quad Socket, Quad Core (16 cores) Intel® Xeon® CPU X5550 @2.67GHz
Memory	48 GB
Storage	4 x 146 GB SATA internal disk drive (RAID 1) 2 x Qlogic QMH2562 8Gb FC HBA 1 x 20GB
Network	8 x Broadcom NetXtreme II BCM57711E XGb

Table 3.3: Cluster Node2 Configuration

3.4 Cluster Server - Node 3

Component	Detail
Hostname	<i>ha-web3</i>
Operating System	Red Hat Enterprise Linux 6.1 (64-bit) (2.6.32-131.4.1.el6.x86_64 kernel)
System Type	HP ProLiant BL460c G6
Processor	Quad Socket, Quad Core (16 cores) Intel® Xeon® CPU X5550 @2.67GHz
Memory	48 GB
Storage	4 x 146 GB SATA internal disk drive (RAID 1) 2 x Qlogic QMH2562 8Gb FC HBA 1 x 20GB
Network	8 x Broadcom NetXtreme II BCM57711E XGb

Table 3.4: Cluster Node3 Configuration



3.5 Fibre Channel Storage Array

Component	Detail
Hostname	<i>ra-msa20</i>
System Type	HP StorageWorks MSA2324fc (1 x HP MSA70 expansion shelf)
Controllers	CPU Type: Turion MT32 1800MHz Cache: 1GB 2 x Host Ports
Firmware	Storage Controller Code Version: M112R14 Memory Controller FPGA Code Version: F300R22 Storage Controller Loader Code Version: 19.009 Management Controller Code Version: W441R35 Management Controller Loader Code Version: 12.015 Expander Controller Code Version: 1112 CPLD Code Version: 8 Hardware Version: 56
Physical Drives	48 x 146GB SAS drives (24 enclosure, 24 expansion shelf)
Logical Drives	4 x 1.3 TB Virtual Disks (12 disk, RAID 6)

Table 3.5: Cluster Storage Array Configuration



4 Cluster Deployment

4.1 Deployment Task Flow

Figure 4.1-1: Cluster Deployment Task Flow provides an overview of the order in which the deployment of the cluster management server, cluster nodes and cluster creation tasks are performed:

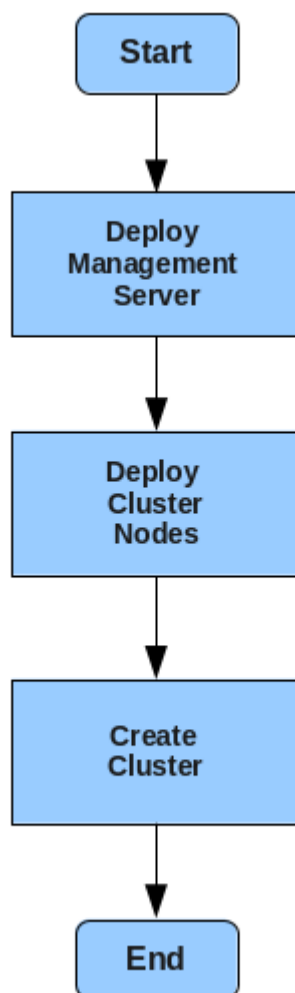


Figure 4.1-1: Cluster Deployment Task Flow

Appendix G: Deployment Checklists provides a detailed list of steps to follow for deploying a Highly Available Web Service on Red Hat Enterprise Linux 6.



4.2 Deploy Management Server

Prior to creating the cluster, the following series of steps is performed to deploy the **luci** management server:

- Install Red Hat Enterprise Linux 6
- Configure Networks
- Configure Firewall
- Install Cluster Management Software (*High Availability Management Add-On*)

The next sections describe how to perform the deployment steps in detail.

4.2.1 Install Red Hat Enterprise Linux 6

The Red Hat Enterprise Linux 6 Installation Guide¹ provides complete details on the installation of Red Hat Enterprise Linux 6 for Intel, AMD, and IBM architectures. The cluster management server can be deployed as either a physical or virtual machine. A KVM virtual machine was used for this reference architecture. Using the **virt-install** utility, the virtual machine was provisioned with 2 processors, 4 GB memory and a bridged network interface as follows:

```
# virt-install --name ha-web-mgmt --ram 4096 --vcpus=2 --cpuset=auto \
  --os-type=linux --os-variant=rhel6 --accelerate \
  --network=bridge:vm-bridge --vnc \
  --disk path=/dev/vm_storage_vg/ha-web-mgmt --pxe
```

A previously created 30 GB logical volume (*ha-web-mgmt*) was created to hold the virtual machine itself on the hypervisor host. For details on the provisioning of KVM virtual machines, please consult the Red Hat Enterprise Linux 6 Virtualization Guide³.

Regardless of whether a physical or virtual machine is used, a Red Hat Enterprise Linux 6 installation involves the following series of stages:

1. Install Red Hat Enterprise Linux 6
2. FirstBoot
3. Apply updates

After the operating system has been installed the system reboots and enters what is referred to as *FirstBoot*. During *FirstBoot*, administrators are guided through the process of setting date and time, configuring software updates, registering with Red Hat Network (RHN), initial user account creation and options for Kernel (*Kdump*) crash dumps. The system then reboots to activate the changes. After login has been completed under the newly created user account, updates to the system are applied to bring the Red Hat Enterprise Linux 6 server to the latest versions of all software.

The Red Hat Enterprise Linux 6 Installation Guide¹ provides complete instructions on each of these stages. Please consult the guide for further installation details.



4.2.2 Configure Networks

The management server is configured to provide access to all cluster nodes across both the public and cluster interconnect (private) networks. The public network (*10.16.143.0*) is configured on the **eth0** interface and the cluster interconnect (*192.168.1.0*) is configured on the **eth3** interface. Static IP addressing is used throughout the cluster configuration.

1. Verify that NetworkManager is disabled on startup to prevent conflicts with the High Availability Add-On cluster services:

```
# chkconfig NetworkManager off
# chkconfig NetworkManager --list
NetworkManager    0:off 1:off 2:off 3:off 4:off 5:off 6:off
```

2. Create the interface file for the public interface and save the file as */etc/sysconfig/network-scripts/ifcfg-eth0*:

```
DEVICE="eth0"
HWADDR="52:54:00:E2:8E:85"
BOOTPROTO=static
ONBOOT="yes"
NM_CONTROLLED="no"
IPADDR=10.16.143.154
NETMASK=255.255.255.0
GATEWAY=10.16.143.254
```

3. Create the interface file for the cluster interconnect and save the file as */etc/sysconfig/network-scripts/ifcfg-eth3*:

```
DEVICE="eth3"
HWADDR="52:54:00:E7:1A:33"
BOOTPROTO=static
ONBOOT="yes"
NM_CONTROLLED="no"
IPADDR=192.168.1.154
NETMASK=255.255.255.0
```

4. Restart the network services to activate the changes:

```
# service network restart
```

5. Update */etc/hosts*

Edit the */etc/hosts* file to include the IP addresses, hostname/aliases of all cluster node and management server interfaces:



```

127.0.0.1      localhost localhost.localdomain
#
#-----#
# Cluster Nodes: #
#-----#
#
10.16.143.151  ha-web1     ha-web1.cloud.lab.eng.bos.redhat.com
192.168.1.151  ha-web1-ci  ha-web1-ci.cloud.lab.eng.bos.redhat.com
10.16.143.152  ha-web2     ha-web2.cloud.lab.eng.bos.redhat.com
192.168.1.152  ha-web2-ci  ha-web2-ci.cloud.lab.eng.bos.redhat.com
10.16.143.153  ha-web3     ha-web3.cloud.lab.eng.bos.redhat.com
192.168.1.153  ha-web3-ci  ha-web3-ci.cloud.lab.eng.bos.redhat.com

#
#-----#
# Management Node: #
#-----#
#
10.16.143.154  ha-web-mgmt  ha-web-mgmt.cloud.lab.eng.bos.redhat.com
192.168.1.154  ha-web-mgmt-ci  ha-web-mgmt-ci.cloud.lab.eng.bos.redhat.com

```

4.2.3 Configure Firewall

Before the cluster can be created, the firewall ports must be configured to allow access to the cluster network daemons. The specific ports requiring access:

Port Number	Protocol	Component
8084	TCP	luci (Conga Administration Interface)

Table 4.1.3: Cluster Node Ports

Firewall access can be configured with either the **system-configuration-firewall** graphical tool or the **iptables** command line utility. Using **iptables**, configure the firewall on each of the three cluster nodes – *ha-web1*, *ha-web2*, *ha-web3* as follows:

1. Create a backup copy of the current **iptables** configuration file:

```
# cp /etc/sysconfig/iptables-config /etc/sysconfig/iptables-config.orig
```

2. Display the current **iptables** configuration:

```
# iptables --list --line-numbers --numeric --verbose
Chain INPUT (policy ACCEPT 599 packets, 48922 bytes)
num  pkts bytes target          prot opt in out source destination
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target          prot opt in out source destination
Chain OUTPUT (policy ACCEPT 379 packets, 38984 bytes)
num  pkts bytes target          prot opt in out source destination
```



3. Create a new **iptables** chain called *ha-web-cluster* and insert it into the **INPUT** chain:

```
# iptables --new-chain ha-web-cluster
# iptables --insert INPUT --jump ha-web-cluster
```

4. Add the rules for the cluster components to the *ha-web-cluster* chain:

```
# iptables --append ha-web-cluster --proto tcp --destination-port 8084 \
--jump ACCEPT
```

5. Display the new **iptables** configuration:

```
# iptables --list --line-numbers --numeric --verbose
Chain INPUT (policy ACCEPT 33 packets, 2894 bytes)
num  pkts bytes target          prot opt in out source          destination
1    340 27206 ha-web-cluster all  --  *  *   0.0.0.0/0      0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target          prot opt in out source          destination

Chain OUTPUT (policy ACCEPT 5 packets, 900 bytes)
num  pkts bytes target          prot opt in out source          destination

Chain ha-web-cluster (1 references)
num  pkts bytes target     prot opt in out source          destination
1    0    0    ACCEPT    tcp  --  *  *   0.0.0.0/0      0.0.0.0/0      tcp dpt:8084
```

6. Save the new rules and verify **iptables** is activated on system boot:

```
# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[ OK ]
# chkconfig iptables on
```

4.2.4 Install Cluster Management Software

1. Log onto Red Hat Network (RHN) and assign the following channels to the cluster management server (*ha-web-mgmt*):

- RHEL Server High Availability (v. 6 for 64-bit x86_64)
- RHN Tools for RHEL (v. 6 for 64-bit x86_64)

2. Confirm the channels are available on the cluster management server:

```
# yum repolist
Loaded plugins: product-id, rhnplugin, subscription-manager
Updating Red Hat repositories.
repo id          repo name
rhel-x86_64-server-6    Red Hat Enterprise Linux Server (v. 6 for 64-bit
x86_64)
rhel-x86_64-server-ha-6 RHEL Server High Availability (v. 6 for 64-bit
x86_64)
rhn-tools-rhel-x86_64-server-6  RHN Tools for RHEL (v. 6 for 64-bit x86_64)
```



3. Install the cluster management software on *ha-web-mgmt*:

```
# yum groupinstall "High Availability Management"
```

This completes the deployment of the cluster management server. Proceed to **Section 4.3 Deploy Cluster Nodes** to begin the process of deploying the three cluster nodes.



4.3 Deploy Cluster Nodes

Prior to creating the cluster, each cluster node is deployed by performing the following series of steps on each cluster node:

- Install Red Hat Enterprise Linux 6
- Configure Networks and Bonding
- Configure Firewall
- Install Cluster Node Software (“High Availability” Add-On)
- Configure Storage
- Configure Web Server Software

The next sections describe how to perform the deployment steps in detail.

4.3.1 Install Red Hat Enterprise Linux 6

The installation of Red Hat Enterprise Linux 6 on each of the three cluster nodes is performed using a Red Hat Satellite server. Details on how the Satellite server was configured can be found in **Appendix B: Red Hat Enterprise Linux 6 – Satellite Configuration Details**. Local media can be used in lieu of a Satellite server deployment.

Once the Satellite server has been configured, perform the following steps to install Red Hat Enterprise Linux 6 on each cluster node:

1. Boot the node using the Preboot eXecution Environment (PXE). Many servers are configured to PXE boot by default. Consult the server vendor documentation for configuration details.
2. After the Red Hat Enterprise Linux 6 installation has completed, verify the repositories on the Satellite server are available:

```
# yum repolist
Loaded plugins: product-id, rhnplugin, subscription-manager
Updating Red Hat repositories.
repo id                repo name
rhel-x86_64-server-6    Red Hat Enterprise Linux Server (v. 6 for 64-bit x86_64)
rhel-x86_64-server-ha-6 RHEL Server High Availability (v. 6 for 64-bit x86_64)
rhn-tools-rhel-x86_64-server-6 RHN Tools for RHEL (v. 6 for 64-bit x86_64)
```

3. Update each node to take in the latest patches and security updates:

```
# yum update
```

Except for the use of PXE boot the installation stages are similar to those as described earlier for the management server in **Section 4.2.1 Install Red Hat Enterprise Linux 6**. Follow the steps above and consult the Red Hat Enterprise Linux 6 Installation Guide¹ for further installation details.



4.3.2 Configure Networks and Bonding

The cluster nodes are configured to provide access to all cluster nodes across both the public and cluster interconnect (*private*) networks. The public network (*10.16.143.0*) is configured on the **eth0** interface and bonded to the **eth1** interface for redundancy. The cluster interconnect (*192.168.1.0*) is configured on the **eth2** interface and bonded to the **eth3** interface for redundancy. Static IP addressing is used throughout the cluster configuration.

1. Verify that NetworkManager is disabled on startup to prevent conflicts with the High Availability Add-On cluster services:

```
# chkconfig NetworkManager off
# chkconfig NetworkManager --list
NetworkManager    0:off 1:off 2:off 3:off 4:off 5:off 6:off
```

2. Create bond configuration files for the public and cluster interconnect networks:

```
# echo "alias bond0 bonding" >> /etc/modprobe.d/bonding.conf
# echo "alias bond1 bonding" >> /etc/modprobe.d/bonding.conf
```

3. Create the bond interface file for the public network and save the file as */etc/sysconfig/network-scripts/ifcfg-bond0*:

```
DEVICE=bond0
IPADDR=10.16.143.151
NETMASK=255.255.255.0
GATEWAY=10.16.143.254
USERCTL=no
BOOTPROTO=static
ONBOOT=yes
BONDING_OPTS="mode=0 miimon=100"
```

4. Create the bond interface file for the cluster interconnect network and save the file as */etc/sysconfig/network-scripts/ifcfg-bond1*:

```
DEVICE=bond1
IPADDR=192.168.1.151
NETMASK=255.255.255.0
USERCTL=no
BOOTPROTO=static
ONBOOT=yes
BONDING_OPTS="mode=1"
```

5. Modify the interface file for the first public interface and save the file as */etc/sysconfig/network-scripts/ifcfg-eth0*:

```
DEVICE=eth0
BOOTPROTO=static
HWADDR=00:17:A4:77:24:3C
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```



6. Create the interface file for the second public interface and save the file as `/etc/sysconfig/network-scripts/ifcfg-eth1`:

```
DEVICE=eth1
BOOTPROTO=static
HWADDR=00:17:A4:77:24:3E
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

7. Modify the interface file for the first cluster interconnect and save the file as `/etc/sysconfig/network-scripts/ifcfg-eth2`:

```
DEVICE=eth2
BOOTPROTO=static
HWADDR=00:17:A4:77:24:40
ONBOOT=yes
MASTER=bond1
SLAVE=yes
USERCTL=no
```

8. Create the interface file for the second cluster interconnect and save the file as `/etc/sysconfig/network-scripts/ifcfg-eth3`:

```
DEVICE=eth3
BOOTPROTO=static
HWADDR=00:17:A4:77:24:42
ONBOOT=yes
MASTER=bond1
SLAVE=yes
USERCTL=no
```

9. Restart the networking service:

```
# service network restart
```

10. Verify the public bond is running:

```
# cat /proc/net/bonding/bond0
```

```
Ethernet Channel Bonding Driver: v3.6.0 (September 26, 2009)
```

```
Bonding Mode: load balancing (round-robin)
```

```
MII Status: up
```

```
MII Polling Interval (ms): 100
```

```
Up Delay (ms): 0
```

```
Down Delay (ms): 0
```

```
Slave Interface: eth0
```

```
MII Status: up
```

```
Link Failure Count: 0
```

```
Permanent HW addr: 00:17:a4:77:24:3c
```

```
Slave queue ID: 0
```

```
Slave Interface: eth1
```



```
MII Status: up
Link Failure Count: 0
Permanent HW addr: 00:17:a4:77:24:3e
Slave queue ID: 0
```

11. Verify the cluster interconnect bond is running:

```
# cat /proc/net/bonding/bond1
```

```
Ethernet Channel Bonding Driver: v3.6.0 (September 26, 2009)
```

```
Bonding Mode: fault-tolerance (active-backup)
Primary Slave: None
Currently Active Slave: eth2
MII Status: up
MII Polling Interval (ms): 0
Up Delay (ms): 0
Down Delay (ms): 0
```

```
Slave Interface: eth2
MII Status: up
Link Failure Count: 0
Permanent HW addr: 00:17:a4:77:24:40
Slave queue ID: 0
```

```
Slave Interface: eth3
MII Status: up
Link Failure Count: 0
Permanent HW addr: 00:17:a4:77:24:42
Slave queue ID: 0
```

12. Edit the */etc/hosts* file to include the IP addresses, hostname/aliases of all cluster node and management server interfaces:

```
127.0.0.1      localhost localhost.localdomain
#
#-----#
# Cluster Nodes: #
#-----#
#
10.16.143.151  ha-web1   ha-web1.cloud.lab.eng.bos.redhat.com
192.168.1.151  ha-web1-ci ha-web1-ci.cloud.lab.eng.bos.redhat.com
10.16.143.152  ha-web2   ha-web2.cloud.lab.eng.bos.redhat.com
192.168.1.152  ha-web2-ci ha-web2-ci.cloud.lab.eng.bos.redhat.com
10.16.143.153  ha-web3   ha-web3.cloud.lab.eng.bos.redhat.com
192.168.1.153  ha-web3-ci ha-web3-ci.cloud.lab.eng.bos.redhat.com
#
#-----#
# Management Node: #
#-----#
#
10.16.143.154  ha-web-mgmt ha-web-mgmt.cloud.lab.eng.bos.redhat.com
192.168.1.154  ha-web-mgmt-ci ha-web-mgmt-ci.cloud.lab.eng.bos.redhat.com
```



13. Distribute the file to the other two cluster nodes. For example, if the file was initially created on cluster node *ha-web1* then copy it to the other nodes as follows:

```
# scp -p /etc/hosts ha-web2:/etc/hosts
# scp -p /etc/hosts ha-web3:/etc/hosts
```

14. Verify all public and cluster interconnect interfaces are properly configured and responding:

```
# ping ha-web1
# ping ha-web1-ci
# ping ha-web2
# ping ha-web2-ci
# ping ha-web3
# ping ha-web3-ci
# ping ha-web-mgmt
# ping ha-web-mgmt-ci
```

4.3.3 Configure Firewall

Before the cluster can be created, the firewall ports must be configured to allow access to the cluster network daemons. The specific ports requiring access:

Port Number	Protocol	Component
5404	UDP	corosync/cman (Cluster Manager)
5405	UDP	corosync/cman (Cluster Manager)
11111	TCP	ricci (Cluster Configuration)
11111	UDP	ricci (Cluster Configuration)
21064	TCP	dlm (Distributed Lock Manager)
16851	TCP	modclusterd
80	TCP	httpd (Apache)
443	TCP	httpd (Apache)

Table 4.2.3: Cluster Node Ports

Firewall access can be configured with either **system-configuration-firewall** (GUI) or the **iptables** utility. Use **iptables** to configure the firewall as per the following series of steps on each of the three cluster nodes (*ha-web1*, *ha-web2*, *ha-web3*):

1. Create a backup copy of the current **iptables** configuration file:

```
# cp /etc/sysconfig/iptables-config /etc/sysconfig/iptables-config.orig
```

2. Display the current **iptables** configuration:

```
# iptables --list --line-numbers --numeric --verbose
Chain INPUT (policy ACCEPT 599 packets, 48922 bytes)
num  pkts bytes target      prot opt in      out     source  destination
```




```
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target          prot opt in     out     source  destination

Chain OUTPUT (policy ACCEPT 379 packets, 38984 bytes)
num  pkts bytes target          prot opt in     out     source  destination
```

3. Create a new **iptables** chain called *ha-web-cluster* and insert it into the **INPUT** chain:

```
# iptables --new-chain ha-web-cluster
# iptables --insert INPUT --jump ha-web-cluster
```

4. Create a new **iptables** chain called *ha-web-service* and insert it into the **INPUT** chain:

```
# iptables --new-chain ha-web-service
# iptables --insert INPUT --jump ha-web-service
```

5. Add the rules for the cluster components to the *ha-web-cluster* chain:

```
# iptables --append ha-web-cluster --proto udp --destination-port 5404 \
--jump ACCEPT
# iptables --append ha-web-cluster --proto udp --destination-port 5405 \
--jump ACCEPT
# iptables --append ha-web-cluster --proto tcp --destination-port 11111 \
--jump ACCEPT
# iptables --append ha-web-cluster --proto udp --destination-port 11111 \
--jump ACCEPT
# iptables --append ha-web-cluster --proto tcp --destination-port 21064 \
--jump ACCEPT
# iptables --append ha-web-cluster --proto udp --destination-port 21064 \
--jump ACCEPT
```

6. Add the rule for web service components to the *ha-web-service* chain:

```
# iptables --append ha-web-service --proto tcp --destination-port 80 \
--jump ACCEPT
# iptables --append ha-web-service --proto tcp --destination-port 443 \
--jump ACCEPT
```

7. Display the new **iptables** configuration:

```
# iptables --list --line-numbers --numeric --verbose
Chain INPUT (policy ACCEPT 5 packets, 404 bytes)
num  pkts bytes target          prot opt in     out     source  destination
1    325  28602 ha-web-service all  --  *  *  0.0.0.0/0  0.0.0.0/0
2    2249 208K  ha-web-cluster all  --  *  *  0.0.0.0/0  0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target          prot opt in     out     source  destination

Chain OUTPUT (policy ACCEPT 3 packets, 508 bytes)
num  pkts bytes target          prot opt in     out     source  destination
```



```
Chain ha-web-cluster (1 references)
num  pkts bytes target prot opt in  out  source      destination
1    0    0    ACCEPT udp  --  *   *   0.0.0.0/0  0.0.0.0/0  udp dpt:5404
2    0    0    ACCEPT udp  --  *   *   0.0.0.0/0  0.0.0.0/0  udp dpt:5405
3    0    0    ACCEPT tcp  --  *   *   0.0.0.0/0  0.0.0.0/0  tcp dpt:11111
4    0    0    ACCEPT udp  --  *   *   0.0.0.0/0  0.0.0.0/0  udp dpt:11111
5    0    0    ACCEPT tcp  --  *   *   0.0.0.0/0  0.0.0.0/0  tcp dpt:21064
6    0    0    ACCEPT udp  --  *   *   0.0.0.0/0  0.0.0.0/0  udp dpt:21064

Chain ha-web-service (1 references)
num  pkts bytes target prot opt in  out  source      destination
1    0    0    ACCEPT tcp  --  *   *   0.0.0.0/0  0.0.0.0/0  tcp dpt:80
2    0    0    ACCEPT tcp  --  *   *   0.0.0.0/0  0.0.0.0/0  tcp dpt:443
```

8. Save the new rules and verify **iptables** is activated on system boot:

```
# service iptables save
# chkconfig iptables on
```

4.3.4 Install Cluster Node Software

Install the *High Availability Add-On* software on each of the three cluster nodes:

```
# yum groupinstall "High Availability"
```

4.3.5 Configure Storage

A volume is created to hold the contents of the cluster web service that can be shared exclusively among the cluster nodes. The volume is configured so that exclusive access is given to the node currently providing the web service. In the event of a failover, access to the volume is changed to be exclusive to the next cluster node providing the web service. Since only one cluster node at a time requires access, the volume is configured with the *Logical Volume Manager (LVM)* and is managed by the *High Availability Logical Manager (HA-LVM)* agent.

The *Logical Unit Number (LUN)* for the volume must be provisioned and accessible to each of the cluster nodes before continuing. **Appendix C: Fibre Channel Storage Provisioning** describes how the LUN used for this reference architecture was provisioned.

4.3.5.1 Configure Multipathing

1. Install the *DM Multipath* Package on each cluster node:

```
# yum install device-mapper-multipath.x86_64
```

2. On the first cluster node (*ha-web1*) create a Multipath configuration file (*/etc/multipath.conf*) with user friendly names disabled and the daemon started:

```
# mpathconf --enable --user_friendly_names n --with_multipathd y
Starting multipathd daemon: [ OK ]
```



3. On the first cluster node (*ha-web1*) view the multipath device, paths and World Wide ID (WWID):

```
# multipathd -ll
3600c0ff000dab7f45ce8f54d01000000 dm-1 HP,MSA2324fc
size=19G features='1 queue_if_no_path' hwhandler='0' wp=rw
`-+- policy='round-robin 0' prio=50 status=active
  |- 0:0:8:1 sdb 8:16 active ready running
  |- 0:0:9:1 sdc 8:32 active ready running
  |- 1:0:8:1 sdd 8:48 active ready running
  `-- 1:0:9:1 sde 8:64 active ready running

# ls /dev/mapper
3600c0ff000dab7f45ce8f54d01000000 control myvg-rootvol
```

4. On the first cluster node (*ha-web1*) edit the file */etc/multipath.conf* and add an alias for the HA-LVM volume (called *ha-web*) using the WWID from the previous step:

```
multipaths {
    multipath {
        alias ha-web
        wwid "3600c0ff000dab7f45ce8f54d01000000"
    }
}
```

5. Copy the file from the first cluster node (*ha-web1*) to the other cluster nodes:

```
# scp -p /etc/multipath.conf ha-web2:/etc/multipath.conf
# scp -p /etc/multipath.conf ha-web3:/etc/multipath.conf
```

6. Restart **multipathd** on each cluster node:

```
# service multipathd restart
Stopping multipathd daemon: [ OK ]
Starting multipathd daemon: [ OK ]
```

7. Verify the change on each cluster node:

```
# multipathd -ll
ha-web (3600c0ff000dab7f45ce8f54d01000000) dm-1 HP,MSA2324fc
size=19G features='1 queue_if_no_path' hwhandler='0' wp=rw
`-+- policy='round-robin 0' prio=50 status=active
  |- 0:0:8:1 sdb 8:16 active ready running
  |- 0:0:9:1 sdc 8:32 active ready running
  |- 1:0:8:1 sdd 8:48 active ready running
  `-- 1:0:9:1 sde 8:64 active ready running

# ls /dev/mapper
control myvg-rootvol
```

8. On each cluster node configure multipath to start on system boot:

```
# chkconfig multipathd on
```



4.3.5.2 Create HA-LVM Volume

1. Ensure the parameter **locking_type** is set to a value of **1** (to prevent LVM metadata corruption) in the global section of the file `/etc/lvm/lvm.conf` on all nodes:

```
# grep "locking_type" /etc/lvm/lvm.conf | grep -v "#"  
locking_type = 1
```

2. Configure the Physical Volume (PV) using the Multipath device - `/dev/mapper/ha-web` and display the attributes. Perform this step on the first cluster node (`ha-web1`) only:

```
# pvcreate /dev/mapper/ha-web  
Physical volume "/dev/mapper/ha-web" successfully created  
  
# pvdisplay /dev/mapper/ha-web  
"/dev/mapper/ha-web" is a new physical volume of "18.63 GiB"  
--- NEW Physical volume ---  
PV Name                /dev/mapper/ha-web  
VG Name  
PV Size                18.63 GiB  
Allocatable           NO  
PE Size               0  
Total PE              0  
Free PE               0  
Allocated PE         0  
PV UUID               7tiJed-mzM2-93cQ-Se0o-RMim-qphs-a0gJkI
```

3. Create a Volume Group (VG) called **HA-Web-VG** and display the attributes. Perform this step on the first cluster node (`ha-web1`) only:

```
# vgcreate --clustered n HA-Web-VG /dev/mapper/ha-web  
Volume group "HA-Web-VG" successfully created  
  
# vgdisplay HA-Web-VG  
--- Volume group ---  
VG Name                HA-Web-VG  
System ID  
Format                lvm2  
Metadata Areas        1  
Metadata Sequence No  1  
VG Access              read/write  
VG Status              resizable  
MAX LV                0  
Cur LV                0  
Open LV               0  
Max PV                0  
Cur PV                1  
Act PV                1  
VG Size                18.62 GiB  
PE Size                4.00 MiB  
Total PE              4768  
Alloc PE / Size       0 / 0  
Free PE / Size        4768 / 18.62 GiB  
VG UUID                UvwCeQ-R8Cr-BjZ7-wuH8-QlHa-N2QM-7rln42
```



4. Create a Logical Volume (LV) called **ha-web-lvol1** and display the attributes. Perform this step on the first cluster node (*ha-web1*) only:

```
# lvcreate --size 10GB --name ha-web-lvol1 HA-Web-VG
Logical volume "ha-web-lvol1" created

# lvs HA-Web-VG
--- Logical volume ---
LV Name                /dev/HA-Web-VG/ha-web-lvol1
VG Name                HA-Web-VG
LV UUID                wSj0Yd-I8Gp-1qjs-TfcN-2In3-YzDu-BvuXrk
LV Write Access        read/write
LV Status              available
# open                 0
LV Size                10.00 GiB
Current LE             2560
Segments               1
Allocation              inherit
Read ahead sectors     auto
- currently set to    256
Block device           253:2
```

5. Format the volume with the ext4 filesystem and a volume label (**HA-Web-Vol-1**). Perform this step on the first cluster node (*ha-web1*) only:

```
# mkfs.ext4 -L HA-Web-Vol-1 /dev/HA-Web-VG/ha-web-lvol1
mkfs 1.41.12 (17-May-2010)
Filesystem label=HA-Web-Vol-1
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
655360 inodes, 2621440 blocks
131072 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2684354560
80 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 29 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

6. Create a mount point for the volume. Perform this step on all cluster nodes:

```
# mkdir -p /ha/ha-web
```



7. Mount the volume and verify it can be written to. Perform this step on the first cluster node (*ha-web1*) only:

```
# mount /dev/HA-Web-VG/ha-web-lvol1 /ha/ha-web
# touch /ha/ha-web/ha-web1.test
# ls -l /ha/ha-web/ha-web1.test
-rw-r--r--. 1 root root 0 Jul 11 11:18 /ha/ha-web/ha-web1.test
```

8. On each cluster node, edit */etc/lvm/lvm.conf* and change the **volume_list** field to match the boot volume (*myvg*) and name of the node cluster interconnect (*ha-web1*). This restricts the list of volumes available during system boot to only the root volume and prevents cluster nodes from updating and potentially corrupting the metadata on the HA-LVM volume:

```
volume_list = [ "myvg", "@ha-web1" ]
```

9. Update the initial ramdisk image so the block device modules are pre-loaded during boot with the **volume_list** restriction and reboot to activate the change. Perform this step on each cluster node:

```
# dracut --hostonly --force /boot/initramfs-$(uname -r).img $(uname -r)
# shutdown -r now "Activating ramdisk LVM changes"
```

4.3.6 Configure Web Server

1. Install the *Web Server* (Apache) software on each of the three cluster nodes:

```
# yum groupinstall "Web Server"
```

2. By default, the **httpd** is not enabled to start on system boot. The cluster facilitates the starting and stopping of the web server as part of the cluster web service. Confirm that **httpd** is disabled during system boot on each cluster node:

```
# chkconfig httpd off
# chkconfig --list httpd
httpd          0:off  1:off  2:off  3:off  4:off  5:off  6:off
```

3. On each cluster node, edit */etc/httpd/conf/httpd.conf* and change **Listen** to include the IP address of the web service and **DocumentRoot** to the mount point of the HA-LVM volume (*/ha/ha-web*):

```
Listen 10.16.143.150:80
DocumentRoot "/ha/ha-web"
```



4.3.7 Configure SELinux Security Parameters

By default, *SELinux* is enabled during the Red Hat Enterprise Linux 6 installation process. For maximum security, Red Hat recommends running Red Hat Enterprise Linux 6 with *SELinux* enabled. In this section, verification is done to ensure that *SELinux* is enabled and the file context set correctly on the */ha/ha-web* filesystem for use by Apache.

1. Verify whether or not *SELinux* is enabled using the **getenforce** utility. Perform this step on all cluster nodes:

```
# getenforce
Enforcing
```

If **getenforce** returns “Permissive” then set to “Enforcing” and verify:

```
# getenforce
Permissive
# setenforce 1
# getenforce
Enforcing
```

2. Edit the file */etc/selinux/config* and set *SELinux* to be persistent across reboots. Perform this step on all cluster nodes:

```
SELINUX=enforcing
```

3. Add (-a) the file context (*fcontext*) for type (-t) *httpd_sys_content* to the directory */ha* and all contents within it. This makes the changes permanent. Perform this step on all cluster nodes:

```
# semanage fcontext -a -t httpd_sys_content_t "/ha(/.*)?"
```

Note: If the *semanage* (*/usr/sbin/semanage*) utility is not available, install the core policy utilities kit:

```
# yum -y install policycoreutils-python
# semanage fcontext -a -t httpd_sys_content_t "/ha(/.*)?"
```

4. View the current security policy file context. Perform this step on all cluster nodes:

```
# ls -ldZ /ha
drwxr-xr-x. root root unconfined_u:object_r:default_t:s0 /ha
```

5. Run the *restorecon* command to apply the changes and view the updated file context. Only perform this step on the cluster node that has */ha/ha-web* mounted:

```
# restorecon -R -v /ha
restorecon reset /ha context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /ha/ha-web context system_u:object_r:file_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /ha/ha-web/lost+found context
system_u:object_r:file_t:s0->system_u:object_r:httpd_sys_content_t:s0
```



```
restorecon reset /ha/ha-web/ha-web1.test context
unconfined_u:object_r:file_t:s0->system_u:object_r:httpd_sys_content_t:s0
# ls -ldZ /ha
drwxr-xr-x. root root system_u:object_r:httpd_sys_content_t:s0 /ha
```

The deployment of the cluster management server and cluster nodes is now complete.

Section 4.3 Cluster Creation via Conga details the process of creating the cluster and configuring the web service using the **luci** web interface. Alternatively, **Section 4.4 Cluster Creation via CCS** details the process of creating the cluster from the command line using the CCS utility.



4.4 Cluster Creation via Conga

The **luci** graphical user interface portion of Conga allows administrators to create, modify and view a cluster configuration file on a remote node through the **ricci** service. Using **luci** an administrator can also start, stop and relocate cluster services on one or more cluster nodes.

In the prior sections, the cluster management server and cluster nodes were fully deployed. Do not proceed with creating the cluster until these tasks have been fully completed:

Cluster Management Station (*ha-web-mgmt*)

- Install Red Hat Enterprise Linux 6
- Configure Networks
- Configure Firewall
- Install Cluster Management Software (“High Availability Management” Add-On)

Cluster Nodes (*ha-web1, ha-web2, ha-web3*)

- Install Red Hat Enterprise Linux 6
- Configure Networks and Bonding
- Configure Firewall
- Install Cluster Node Software (“High Availability” Add-On)
- Configure Storage
- Configure Web Server Software

The next sections describe the steps involved in creating the cluster from the **luci** web interface on the cluster management server.

4.4.1 Create Cluster

Creating a cluster with **luci** consists of naming the cluster, adding cluster nodes to the cluster, entering **ricci** passwords for each node and submitting the request to create a cluster. If the node information and passwords are correct, Conga automatically installs any needed or missing cluster software components onto the cluster nodes and starts the cluster. Follow the steps below to start the required cluster services, create the cluster and add the nodes to it.

1. Start the **ricci** service and configure to start on system boot. Perform this step on all cluster nodes:

```
# service ricci start
Starting oddjobd: [ OK ]
generating SSL certificates... done
Generating NSS database... done
Starting ricci: [ OK ]

# chkconfig ricci on
```



2. Configure a password for the **ricci** user account on all cluster nodes. The same password may be used on all cluster nodes to simplify administration:

```
# passwd ricci
Changing password for user ricci.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

3. Start **luci** on the management server:

```
# service luci start
Generating a 2048 bit RSA private key
writing new private key to '/var/lib/luci/certs/host.pem'
Starting saslauthd: [ OK ]
Start luci... [ OK ]
Point a web browser to https://ha-webmgmt.cloud.lab.eng.bos.redhat.com:8084
(or equivalent) to access luci
```

4. Open the **luci** web interface by pointing a browser to the url specified during the **luci** service start:

https://ha-web-mgmt.cloud.lab.eng.bos.redhat.com:8084

and login using the *root* account.

5. After login, **luci** presents the **Homebase** page. Select **Manage Clusters** on the left side of the **Homebase** as shown in **Figure 4.4-1: Manage Clusters**. Next, select **Create**.

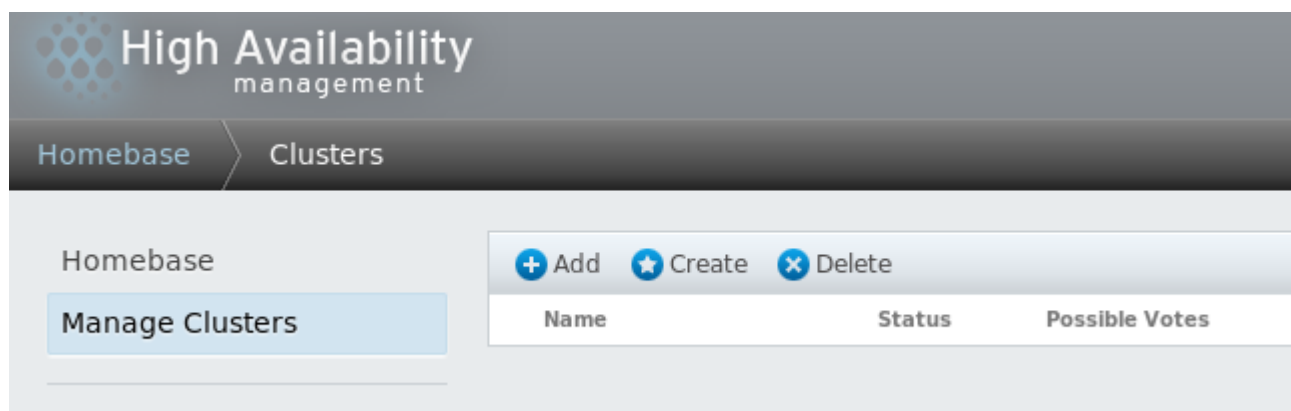


Figure 4.4-1: Manage Clusters



6. The **Create New Cluster** dialog box opens as shown in **Figure 4.4-2: Create New Cluster**. Complete the following in the dialog box:

- Enter the **Cluster Name**. Cluster names are restricted to 15 characters in length.
- Enable the option “**Use the same password for all nodes**”
- Enter the **Node Name** of the first cluster node by specifying the private *cluster interconnect*. This enables cluster communications to run over the private network.
- Enter **Password**. Specify the password for the **ricci** account.
- Enter **Hostname**. Specify the public network name.

Node Name	Password	Ricci Hostname	Ricci Port
ha-web1-ci	●●●●●●●●	ha-web1	11111
ha-web2-ci	●●●●●●●●	ha-web2	11111
ha-web3-ci	●●●●●●●●	ha-web3	11111

Figure 4.4-2: Create New Cluster

Select **Add Another Node** and complete the fields for the other two cluster nodes as shown. When done, enable the option **Download Packages** then select **Create Cluster**.



4.4.2 Add Fence Devices

Fencing provides protection against data corruption caused by multiple cluster nodes performing simultaneous I/O to a shared storage device. Fencing disconnects a cluster node from the shared storage device in the event of a cluster fault or node failure.

The primary fencing method utilized by this reference architecture is IPMI over LAN. Additional fencing methods can be applied based on the resources available within a computing environment. Follow the steps below to add a *fence device*, *fence device instance* and *fence method* to each node in the the cluster.

1. From the **Manage Clusters** screen, select the *ha-web-cluster* link to open the cluster specific page as shown in **Figure 4.4-3: Fence Devices**:

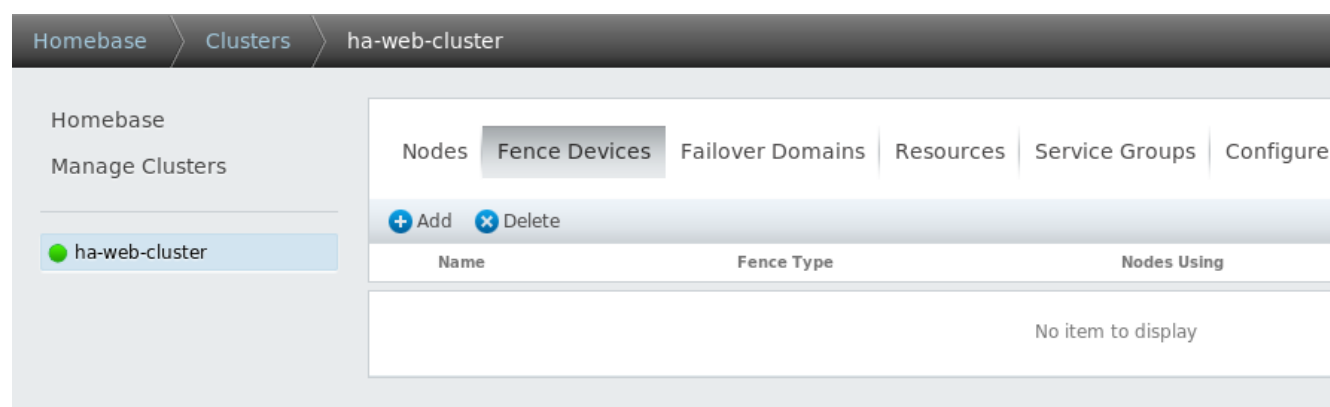


Figure 4.4-3: Fence Devices

Under the **Fence Devices** tab select **Add**.

2. The **Add Fence Device (Instance)** dialog box opens as shown in **Figure 4.4-4: Select Fence Device**:

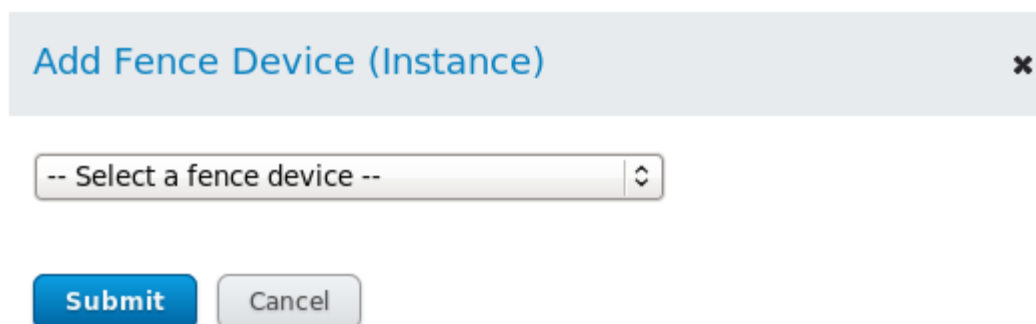


Figure 4.4-4: Select Fence Device

In the – **Select a fence device** – drop-down, select **IPMI Lan** then select **Submit**.



3. The **Add Fence Device (Instance)** dialog box is displayed for the fence type **IPMI Lan**. Complete the following in the **Figure 4.4-5: Add Fence Device Instance** dialog box:

- Enter a **Name** for the IPMI Lan device instance.
- Enter the **IP address or hostname**
- Enter the **Login**. If the device has a different login than the cluster node specify it here.
- Enter **Password**. If the device has a different password than the cluster node specify it here.
- In the **Authentication type** drop-down select **Password**
- Enable the **Use Lanplus** option

Add Fence Device (Instance) ✕	
IPMI Lan	
Fence type	IPMI Lan
Name	IPMI-ha-web1-ci
IP address or hostname	10.16.143.231
Login	root
Password	●●●●●●●●
Password Script (optional)	
Authentication type	Password
Use Lanplus	<input checked="" type="checkbox"/>
Ciphersuite to use	
<input type="button" value="Submit"/> <input type="button" value="Cancel"/>	

Figure 4.4-5: Add Fence Device Instance

Select **Submit** when done.



4. Add a fence device instance for the other cluster nodes by following steps 2 and 3 above for each node. When all instances have been created, the **Fence Devices** tab resembles **Figure 4.4-6: Fence Device Instances** below:

	Name	Fence Type	Nodes Using
<input type="checkbox"/>	IPMI-ha-web1-ci	IPMI Lan	0
<input type="checkbox"/>	IPMI-ha-web2-ci	IPMI Lan	0
<input type="checkbox"/>	IPMI-ha-web3-ci	IPMI Lan	0

Figure 4.4-6: Fence Device Instances

5. Under the **Nodes** tab (see **Figure 4.4-7: Cluster Nodes** below) select the link for the first cluster node *ha-web1-ci*.

!	Node Name	Node ID	Votes	Status
<input type="checkbox"/>	ha-web1-ci	1	1	Cluster Member
<input type="checkbox"/>	ha-web2-ci	2	1	Cluster Member
<input type="checkbox"/>	ha-web3-ci	3	1	Cluster Member

Figure 4.4-7: Cluster Nodes



6. Scroll down to the properties of the cluster node as shown in **Figure 4.4-8: Add Fence Method**. Under **Fence Devices** select **Add Fence Method**.

The screenshot shows the configuration page for a cluster node named 'ha-web1-ci', which is a 'Cluster Member'. The page is divided into several sections: 'Properties', 'Services', 'Failover Domains', and 'Fence Devices'. Under 'Properties', there are three input fields: 'Number of votes' with the value '1', 'ricci host' with the value 'ha-web1', and 'ricci port' with the value '11111'. The 'Fence Devices' section is currently empty, with a button labeled 'Add Fence Method' visible under the 'Method' sub-section.

Figure 4.4-8: Add Fence Method

7. The **Add Fence Method To Node** dialog box (**Figure 4.4-9: Add Fence Method To Node**) opens – enter *Primary* for the **Method Name** and select **Submit**:

The dialog box has a title bar that says 'Add Fence Method To Node' with a close button (x) on the right. Below the title bar, there is a label 'Method Name:' followed by an input field containing the text 'Primary'. At the bottom of the dialog, there are two buttons: a blue 'Submit' button and a grey 'Cancel' button.

Figure 4.4-9: Add Fence Method To Node



- Under the newly created **Primary** fence method, select **Add Fence Instance (Figure 4.4-10: Add Fence Instance)**:

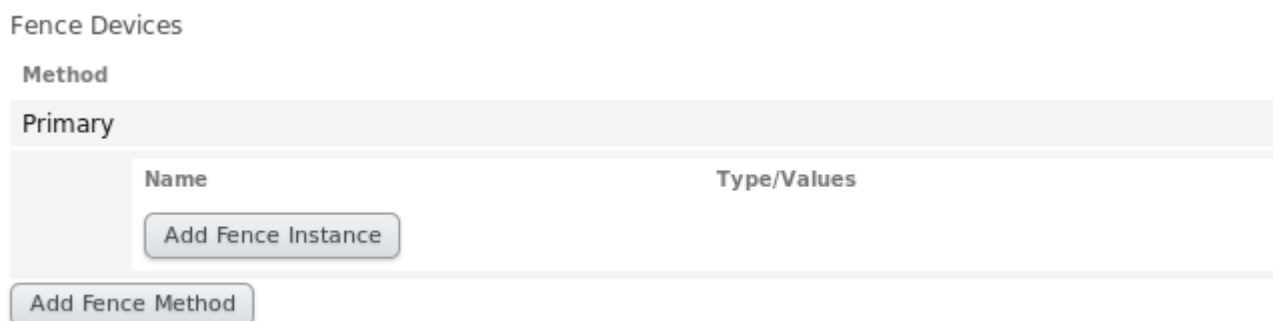


Figure 4.4-10: Add Fence Instance

In the **Add Fence Device (Instance)** dialog box (**Figure 4.4-11: Add Fence Device Instance**), scroll down and select the name of the fence device (previously created in step 3) for the first cluster node (*ha-web1-ci*):

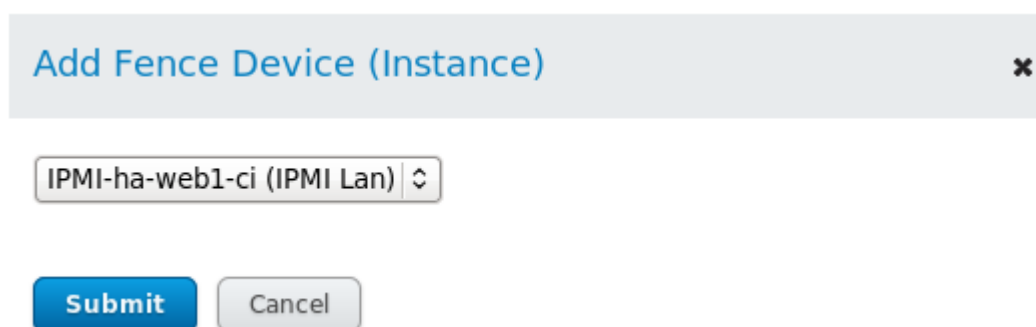


Figure 4.4-11: Add Fence Device Instance

Select **Submit** when done.

- Add a *Primary* fence method and fence device instance for the other cluster nodes by following steps 5, 6, and 7 for each node.



4.4.3 Add Failover Domain

The *failover domain* determines the order and characteristics in which cluster members manage the *failover* and *fallback* of a cluster service. Follow the steps below to add a *failover domain* to the cluster.

1. Under the **Failover Domains** tab (**Figure 4.4-12: Add Failover Domains Tab**) select **Add**:

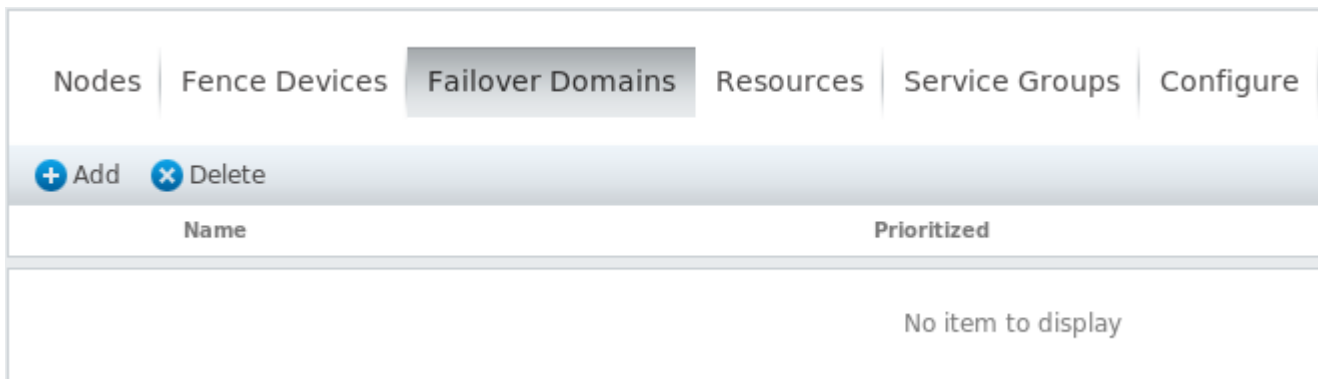


Figure 4.4-12: Add Failover Domains Tab

2. In the **Add Failover Domain To Cluster** dialog box (**Figure 4.4-13: Add Failover Domain To Cluster**), enter the **Name** (*ha-web-failover*), enable the option for **Prioritized** and set the member priorities as shown. Select **Create** when done:

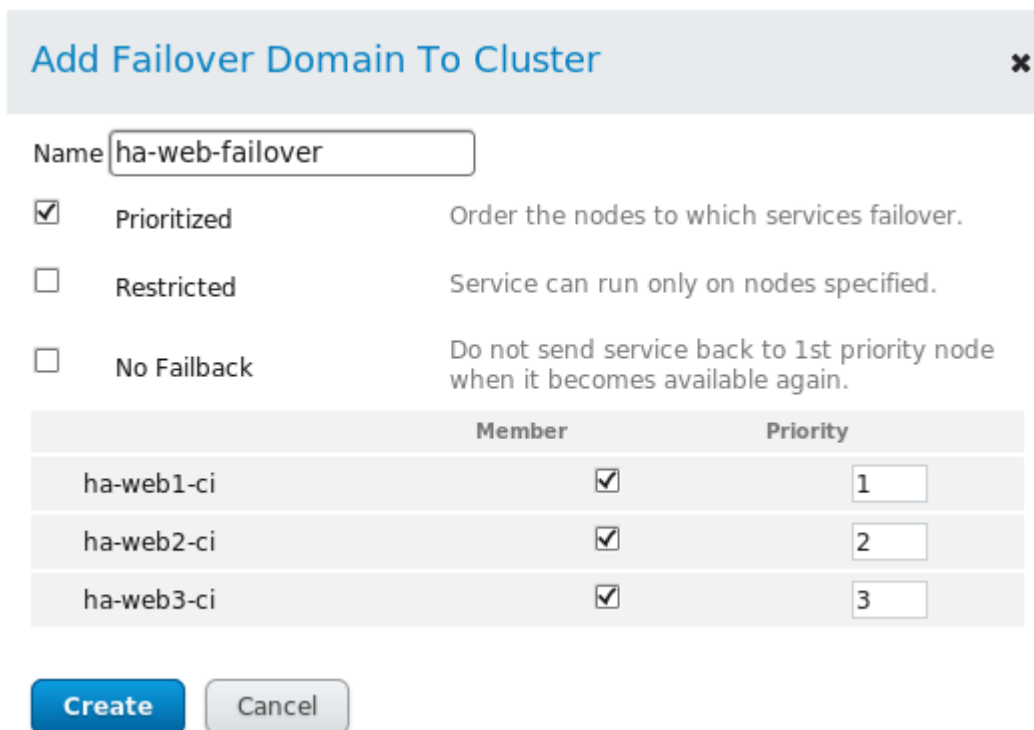


Figure 4.4-13: Add Failover Domain To Cluster



When complete, the **Failover Domains** tab resembles **Figure 4.4-14: Add Failover Domains - Completed** below:

Name	Prioritized	Restricted
<input type="checkbox"/> ha-web-failover	<input checked="" type="checkbox"/>	No

Select an item to view details

Figure 4.4-14: Add Failover Domains - Completed



4.4.4 Add Resources

The highly available web service requires four resources – *IP Address*, *HA LVM*, *Filesystem* and *Script*. Follow the steps below to add the required web service resources to the cluster.

1. From the **Resources** tab (**Figure 4.4-15: Add Resources**) select **Add**:

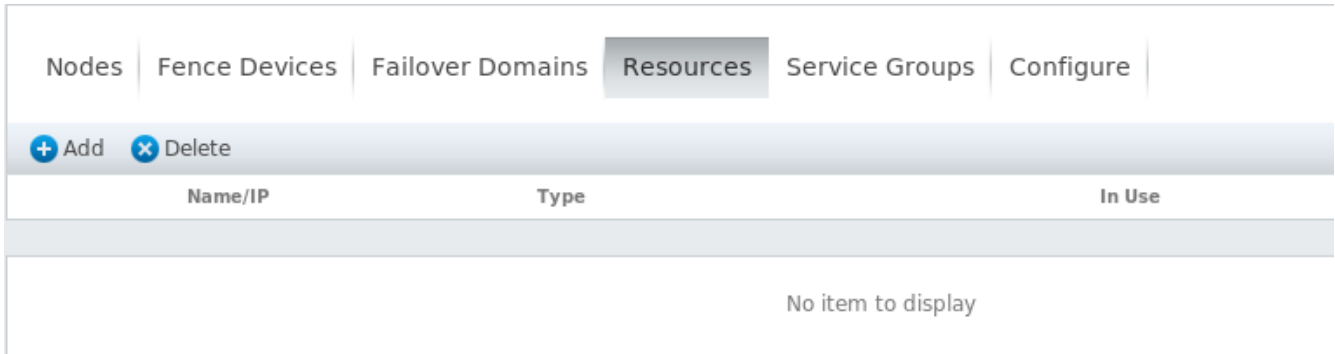


Figure 4.4-15: Add Resources

2. In the **Add Resource To Cluster** dialog box (**Figure 4.4-16: Select Resource Type**), scroll down and select **IP Address**:

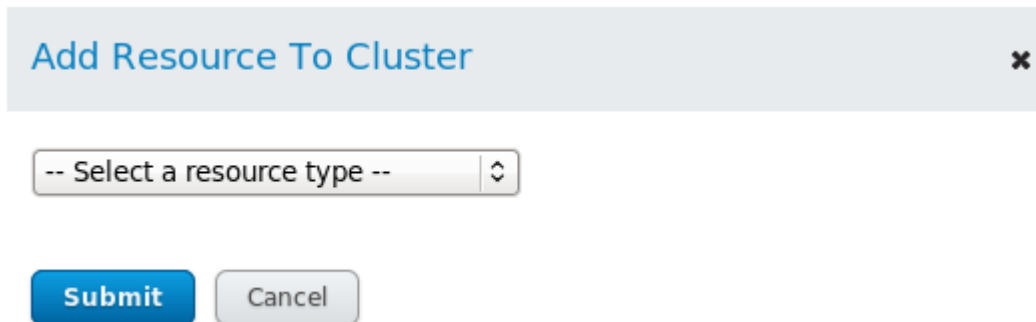


Figure 4.4-16: Select Resource Type



In the **IP Address** resource dialog box (**Figure 4.4-17: Add IP Address Resource**), enter the **IP address** (*10.16.143.150*) to be used for connections to the web service. Enable the option **Monitor link** then select **Submit**:

Add Resource To Cluster ✕

IP Address

IP Address

IP address

Netmask bits (optional)

Monitor link

Disable updates to static routes

Number of seconds to sleep after removing an IP address

Submit **Cancel**

Figure 4.4-17: Add IP Address Resource

- Repeat steps 1 and 2 to add an **HA LVM** resource (*ha-web-HA-LVM*) with the options shown in **Figure 4.4-18: Add HA LVM Resource**. Click **Submit** when done.

Add Resource To Cluster ✕

HA LVM

HA LVM

Name

Volume group name

Logical volume name

Fence the node if it is unable to clean up LVM tags

Submit **Cancel**

Figure 4.4-18: Add HA LVM Resource



- Repeat steps 1 and 2 to add a **Filesystem** resource (*ha-web-filesystem*) with the options shown in **Figure 4.4-19: Add Filesystem Resource**. Click **Submit** when done.

Add Resource To Cluster ✕

Filesystem

Filesystem

Name	ha-web-filesystem
Filesystem type	ext4
Mount point	/ha/ha-web
Device, FS label, or UUID	/dev/HA-Web-VG/ha-web-lvol1
Mount options	
Filesystem ID (optional)	
Force fsck	<input type="checkbox"/>
Use quick status checks	<input type="checkbox"/>
Reboot host node if unmount fails	<input checked="" type="checkbox"/>

Submit **Cancel**

Figure 4.4-19: Add Filesystem Resource

- Repeat steps 1 and 2 to add a **Script** resource (*ha-web-apache-script*) with the options shown in **Figure 4.4-20: Add Script Resource**. Click **Submit** when done.

Add Resource To Cluster ✕

Script

Script

Name	ha-web-apache-script
Full path to script file	/etc/init.d/httpd

Submit **Cancel**

Figure 4.4-20: Add Script Resource



4.4.5 Add Service Group

Service Groups are managed by **rgmanager** and contain the resources needed to provide highly available services. Follow the steps below to add a *Service* named *ha-web-service* and the resources required by the web service to the cluster.

1. From the **Service Groups** tab (**Figure 4.4-21: Add Service Group**), select **Add**:

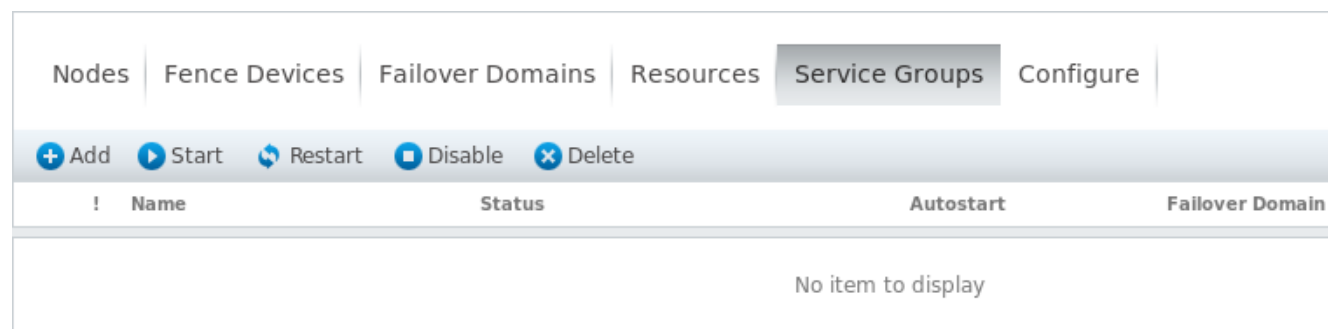


Figure 4.4-21: Add Service Group

2. The **Add Service To Cluster** dialog box opens (**Figure 4.4-22: Add Service To Cluster**). Enter the **Service name** (*ha-web-service*) and enable the options shown. Select **Submit** when done.

Add Service To Cluster ✕

Service name

Automatically start this service

Run exclusive

Failover domain

Recovery policy

Restart options

Maximum number of restart failures before relocating

Length of time in seconds after which to forget a restart

Figure 4.4-22: Add Service To Cluster



3. Add the *IP Address* resource to the service (*ha-web-service*) by selecting **Add Resource**, selecting the resource type *10.16.143.150* from the drop-down and selecting **Submit** as shown in **Figure 4.4-23: Add Resource To Service**:

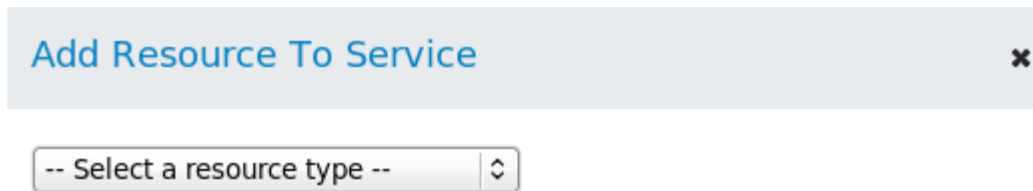


Figure 4.4-23: Add Resource To Service

Repeat the process for the remaining three resources - *HA LVM*, *Filesystem*, *Script*. Each resource should be independent and have no **Child Resources** associated with it.

4. The web service (*ha-web-service*) is displayed at the top of the **Service Groups** tab. The status is listed as **Unknown** until it is started. (**Figure 4.4-24: Service Group Status Unknown**):

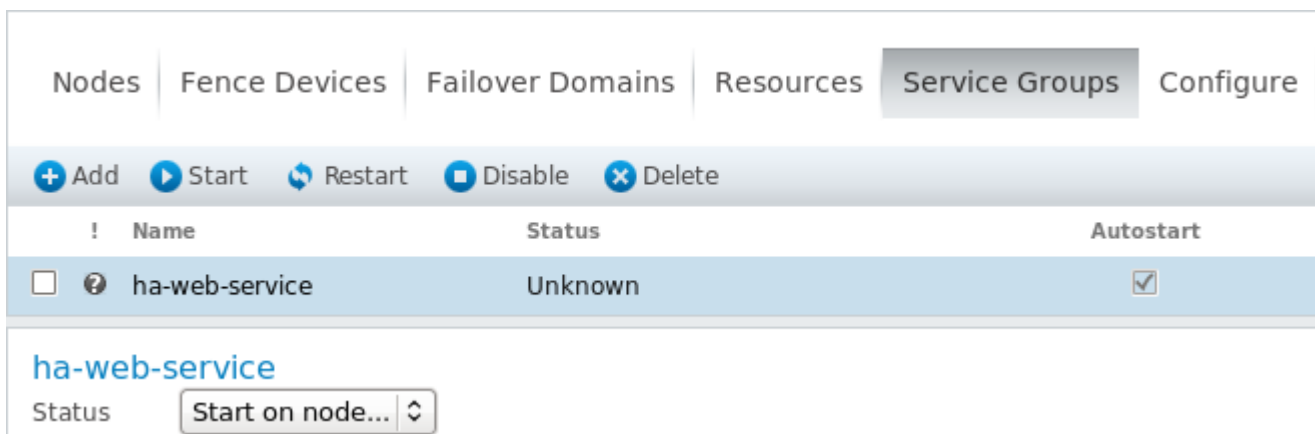


Figure 4.4-24: Service Group Status Unknown

5. Start the web service by enabling the check box next to the **Name** (*ha-web-service*) and selecting **Start** from the menu. Alternatively, the service can be started on a specific node by selecting the node from the **Start on node...** drop-down box then selecting the **Start** icon on the right. Once a service has been started the status is updated as shown in **Figure 4.4-25: Service Group Status Running**:

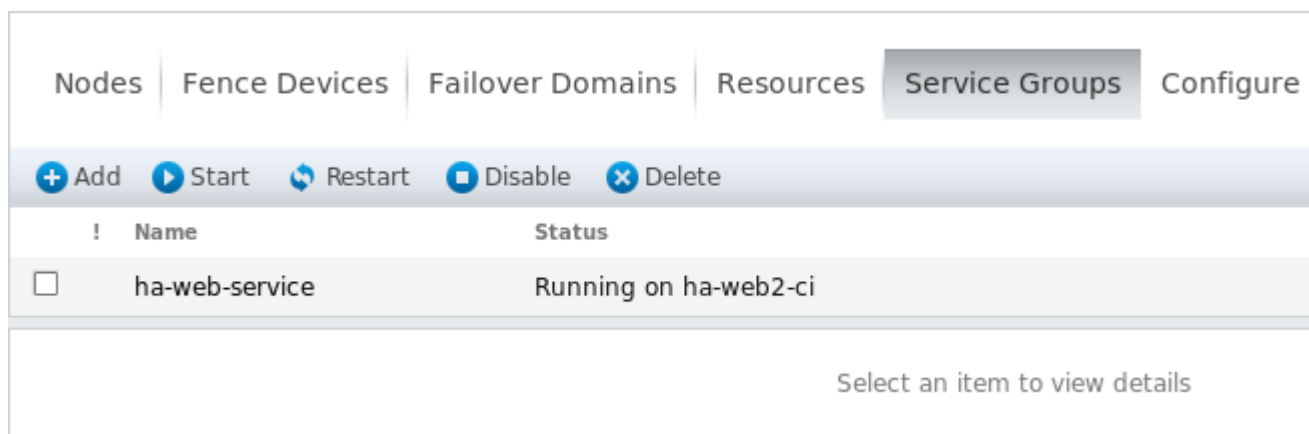


Figure 4.4-25: Service Group Status Running

In the above example, *ha-web-service* is running on cluster node *ha-web2-ci*.

4.4.6 Verify Cluster Web Service

Verify the web service is running by opening a browser window onto the web service (<http://10.16.143.150/cgi-bin/HA-Web-Status>) and running the test script (**Figure 4.4-26: HA-Web-Status**):

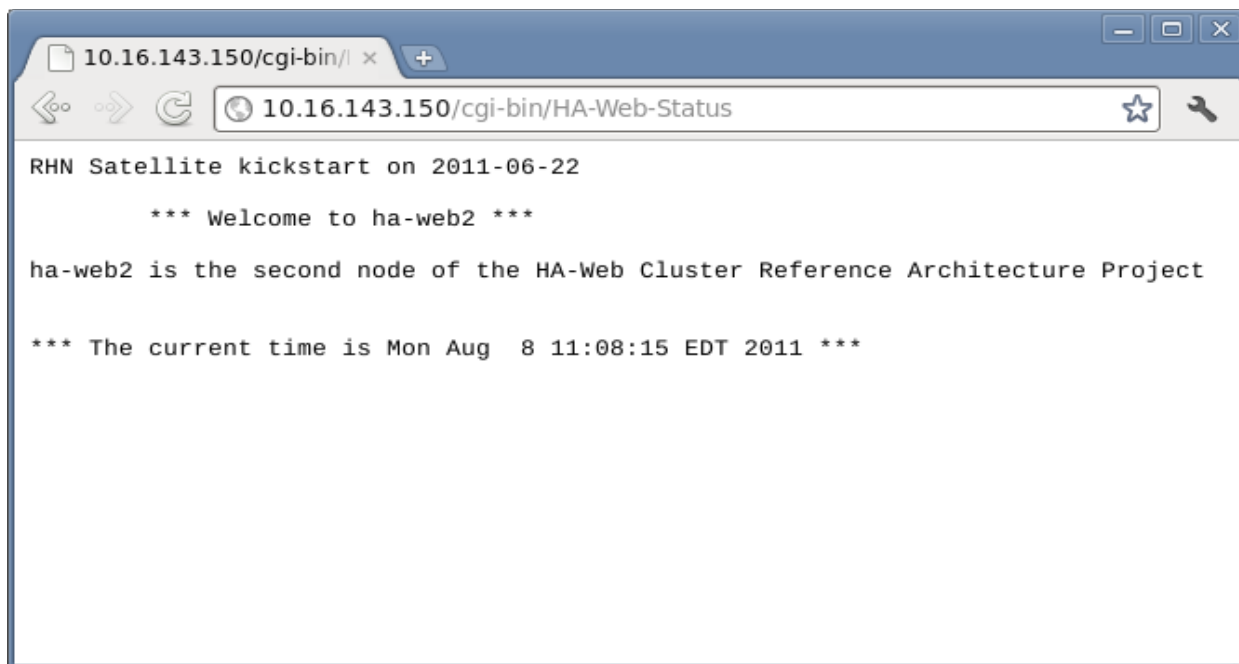


Figure 4.4-26: HA-Web-Status



4.5 Cluster Creation via CCS

The **ccs** (Cluster Configuration System) command line interface allows administrators to create, modify and view a cluster configuration file on a remote node through the **ricci** service or on a local filesystem. Using **ccs** an administrator can also start, stop and relocate cluster services on one or more cluster nodes.

In the prior sections, the cluster management server and cluster nodes were fully deployed. Do not proceed with creating the cluster via CCS until these tasks have been fully completed:

Cluster Management Station (*ha-web-mgmt*)

- Install Red Hat Enterprise Linux 6
- Configure Networks
- Configure Firewall
- Install Cluster Management Software (“High Availability Management” Add-On)

Cluster Nodes (*ha-web1*, *ha-web2*, *ha-web3*)

- Install Red Hat Enterprise Linux 6
- Configure Networks and Bonding
- Configure Firewall
- Install Cluster Node Software (“High Availability” Add-On)
- Configure Storage
- Configure Web Server Software

The next sections describe the steps involved in creating a cluster from the **ccs** command line interface.

4.5.1 Create Cluster

Cluster creation is performed from the cluster management server (*ha-web-mgmt*) and updates are deployed to the cluster nodes across the public network interfaces. The process involves creating a full cluster configuration file (*/etc/cluster/cluster.conf*) on one node (*ha-web1-ci*) then distributing the configuration and activating the cluster on the remaining nodes. Cluster interconnects are specified within the configuration file for all node communications.

Configure the appropriate cluster services then create the cluster.

1. Start the **ricci** service and configure to start on system boot. Perform this step on all cluster nodes:

```
# service ricci start
# chkconfig ricci on
```

2. Configure a password for the **ricci** user account on each node. The same password may be used on all cluster nodes to simplify administration:

```
# passwd ricci
```



3. Create a cluster named **ha-web-cluster** from the cluster management server (*ha-web-mgmt*):

```
# ccs --host ha-web1 --createcluster ha-web-cluster
ha-web1 password: *****
```

4.5.2 Add Nodes

Once the cluster has been created, specify the member nodes in the cluster configuration.

1. Add the three cluster nodes (*ha-web1-ci*, *ha-web2-ci*, *ha-web3-ci*) to the cluster. Perform this step from the cluster management server (*ha-web-mgmt*):

```
# ccs --host ha-web1 --addnode ha-web1-ci -nodeid="1"
Node ha-web1-ci added.
# ccs --host ha-web1 --addnode ha-web2-ci -nodeid="2"
Node ha-web2-ci added.
# ccs --host ha-web1 --addnode ha-web3-ci -nodeid="3"
Node ha-web3-ci added.
```

4.5.3 Add Fence Devices

Add the fence method then add devices and instances for each cluster node to the method. IPMI LAN fencing is used in this configuration. Other fencing methods and devices can be used depending on the resources available. Perform all steps from the cluster management server (*ha-web-mgmt*).

1. Add a fence method for the Primary fencing devices:

```
# ccs --host ha-web1 --addmethod Primary ha-web1-ci
Method Primary added to ha-web1-ci.
# ccs --host ha-web1 --addmethod Primary ha-web2-ci
Method Primary added to ha-web1-ci.
# ccs --host ha-web1 --addmethod Primary ha-web3-ci
Method Primary added to ha-web1-ci.
```

2. Add a fence device for the IPMI LAN device:

```
# ccs --host ha-web1 --addfencedev IPMI-ha-web1-ci \
    agent=fence_ipmilan auth=password \
    ipaddr=10.16.143.231 lanplus=on \
    login=root name=IPMI-ha-web1-ci passwd=password \
    power_wait=5 timeout=20
# ccs --host ha-web1 --addfencedev IPMI-ha-web2-ci \
    agent=fence_ipmilan auth=password \
    ipaddr=10.16.143.232 lanplus=on \
    login=root name=IPMI-ha-web2-ci passwd=password \
    power_wait=5 timeout=20
# ccs --host ha-web1 --addfencedev IPMI-ha-web3-ci \
    agent=fence_ipmilan auth=password \
    ipaddr=10.16.143.233 lanplus=on \
    login=root name=IPMI-ha-web3-ci passwd=password \
    power_wait=5 timeout=20
```



3. Add a fence instance for each node to the Primary fence method:

```
# ccs --host ha-web1 --addfenceinst IPMI-ha-web1-ci ha-web1-ci Primary
# ccs --host ha-web1 --addfenceinst IPMI-ha-web2-ci ha-web2-ci Primary
# ccs --host ha-web1 --addfenceinst IPMI-ha-web3-ci ha-web3-ci Primary
```

4.5.4 Add Failover Domain

The failover domain specifies the membership and failover characteristics for nodes providing a cluster service. Multiple failover domains can be created to provide different failover characteristics for different cluster services.

1. Add a failover domain and add the cluster nodes to it. Specify round-robin failover starting with node1, followed by node2 then node3. Perform these tasks from the cluster management server (*ha-web-mgmt*):

```
# ccs --host ha-web1 --addfailoverdomain ha-web-failover ordered
# ccs --host ha-web1 --addfailoverdomainnode ha-web-failover ha-web1-ci 1
# ccs --host ha-web1 --addfailoverdomainnode ha-web-failover ha-web2-ci 2
# ccs --host ha-web1 --addfailoverdomainnode ha-web-failover ha-web3-ci 3
```

4.5.5 Add Resources

Define each of the resources that comprise the cluster web service. The following resources are required for configuring a highly available web service:

- IP Address
- HA LVM
- Filesystem
- Script

Perform all steps from the cluster management server (*ha-web-mgmt*).

1. Add the **IP Address** resource (**10.16.143.150**) to the cluster configuration. This resource provides remote client access to the web service (**ha-web-service**) and content:

```
# ccs --host ha-web1 --addresource ip address=10.16.143.150 \
    monitor_link=on sleeptime=10
```

2. Add the **HA LVM** resource (**ha-web-HA-LVM**) to the cluster configuration. This resource is responsible for enabling and disabling exclusive access to the HA-LVM volume across cluster nodes:

```
# ccs --host ha-web1 --addresource lvm \
    lv_name=ha-web-lvol1 \
    name=ha-web-HA-LVM \
    self_fence=on vg_name=HA-Web-VG
```



3. Add the **Filesystem** resource (*ha-web-filesystem*) to the cluster configuration. This resource is responsible for the mounting and unmounting of the HA-LVM volume:

```
# ccs --host ha-web1 --addresource fs \
    device=/dev/HA-Web-VG/ha-web-lvol1 \
    fsid=56432 fstype=ext4 \
    mountpoint=/ha/ha-web \
    name=ha-web-filesystem self_fence=on
```

4. Add the **Script** resource (*ha-web-apache-script*) to the cluster configuration. This resource is responsible for the starting and stopping of the Apache (**httpd**) web server:

```
# ccs --host ha-web1 --addresource script \
    file=/etc/init.d/httpd \
    name=ha-web-apache-script
```

4.5.6 Add Service Group

Service groups are collections of resources that simplify the management of highly available cluster services.

1. Create a service group (*ha-web-service*) for the web service and add the required resources (**IP**, **HA-LVM**, **Filesystem**, **Script**) to it:

```
# ccs --host ha-web1 --addservice ha-web-service \
    domain=ha-web-failover \
    max_restarts=3 name=ha-web-service \
    recovery=restart restart_expire_time=3600

# ccs --host ha-web1 --addsubservice ha-web-service \
    ip ref=10.16.143.150
# ccs --host ha-web1 --addsubservice ha-web-service \
    lvm ref=ha-web-HA-LVM
# ccs --host ha-web1 --addsubservice ha-web-service \
    fs ref=ha-web-filesystem
# ccs --host ha-web1 --addsubservice ha-web-service \
    script ref=ha-web-apache-script
```

4.5.7 Activate Cluster

Once the cluster has been created, the configuration needs to be activated and the cluster started on all nodes.

1. Synchronize and activate the cluster configuration across all nodes:

```
# ccs --host ha-web1 --sync --activate
ha-web2-ci password:

# ccs --host ha-web1 --checkconf
All nodes in sync.
```



2. Start the cluster services on all nodes:

```
# ccs --host ha-web1 --startall
Started ha-web2-ci
Started ha-web3-ci
Started ha-web1-ci
```

4.5.8 Verify Cluster Web Service

1. Verify the cluster web service is available from one of the cluster nodes (*ha-web1-ci*).

```
# clustat
Cluster Status for ha-web-cluster @ Mon Aug 8 16:40:08 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, Local, rgmanager
ha-web2-ci           2   Online, rgmanager
ha-web3-ci           3   Online, rgmanager

Service Name         Owner (Last)      State
-----
service:ha-web-service  ha-web1-ci       started
```

2. Verify the web service is running by opening a browser window onto the web service (<http://10.16.143.150/cgi-bin/HA-Web-Status>) and running the test script (**Figure 4.5-1: HA-Web-Status**):

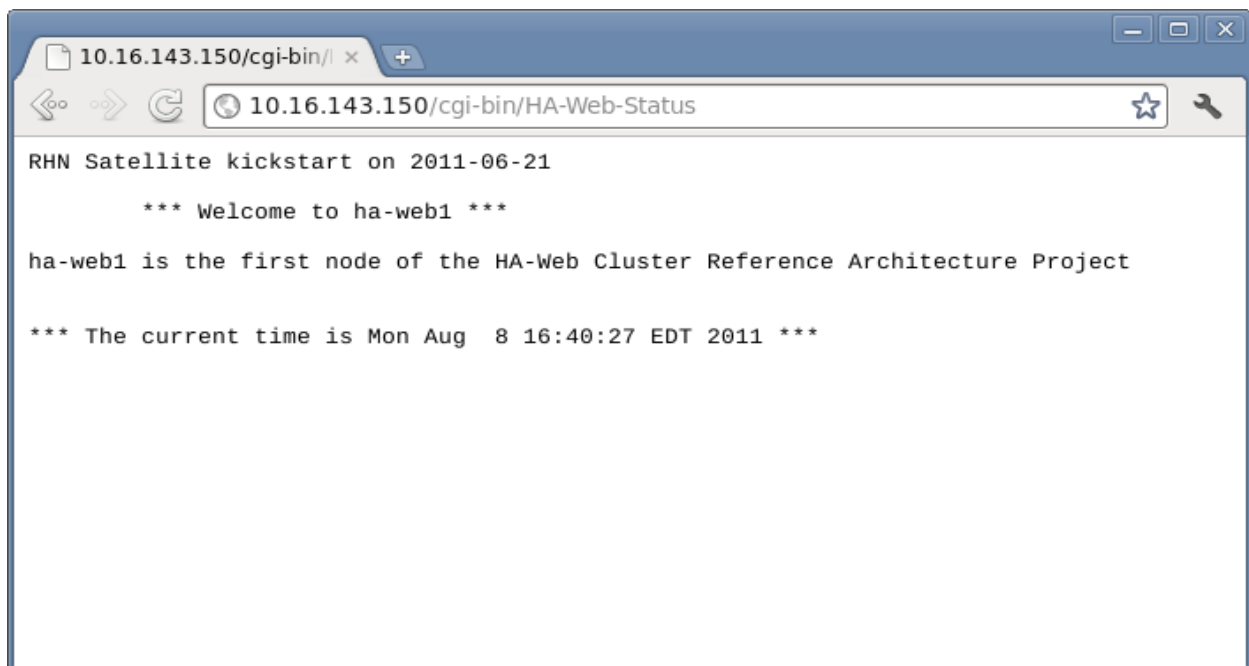


Figure 4.5-1: HA-Web-Status

This completes the cluster creation and verification using CCS. The next section details the common use cases in managing a Red Hat Enterprise Linux 6 highly available web server.



5 Cluster Management

The previous sections of this reference architecture detailed the deployment and configuration tasks associated with deploying a highly available web service using the High Availability Add-On for Red Hat Enterprise Linux. The following sections focus on the most common cluster management tasks using both the Conga (GUI) and CCS (CLI) interfaces.

- Two node clusters are a special case scenario requiring a cluster restart (and brief service downtime) to activate the change in membership when adding (2 -> 3) or removing (3 -> 2) a node. Conga implicitly handles the restart while CCS requires a manual restart. Both methods are demonstrated within the sections that follow describing the adding and removal of cluster nodes.

5.1 Adding Cluster Nodes

Conga (Method 1)

The following steps describe the process of adding a new node (*ha-web3-ci*) to an existing two node cluster using Conga and the **luci** web interface.

1. Open the **luci** web interface and login using the *root* account. Under the **Nodes** tab select **Add (Figure 5.1-1: Add Nodes Tab)**:

The screenshot shows the Conga web interface with the 'Nodes' tab selected. The interface includes a navigation bar with tabs: Nodes, Fence Devices, Failover Domains, Resources, Service Groups, and Configure. Below the navigation bar is a toolbar with buttons: Add, Reboot, Join Cluster, Leave Cluster, and Delete. The main content area displays a table with the following data:

!	Node Name	Node ID	Votes	Status	Uptime
<input type="checkbox"/>	ha-web1-ci	1	1	Cluster Member	00:00:06:24
<input type="checkbox"/>	ha-web2-ci	2	1	Cluster Member	00:00:07:24

Below the table, there is a message: "Select an item to view details".

Figure 5.1-1: Add Nodes Tab



2. In the **Add Nodes To Cluster** dialog box, add the new node to the cluster as shown in **Figure 5.1-2: Add Nodes To Cluster** below. Specify the private *cluster interconnect* name for **Node Name**. Select **Add Nodes** when done:

Add Nodes To Cluster ✕

Use the same password for all nodes

Node Name: ha-web3-ci Password: ●●●●●●●●●● Ricci Hostname: ha-web3 Ricci Port: 11111 ✕

Download Packages
 Use locally installed packages

Reboot nodes before joining cluster

Enable shared storage support

Figure 5.1-2: Add Nodes To Cluster

3. After the node has been added and the cluster configuration updated, the **Nodes** tab resembles **Figure 5.1-3: Cluster Nodes Tab - New Node Added** below:

Nodes | Fence Devices | Failover Domains | Resources | Service Groups | Configure

!	Node Name	Node ID	Votes	Status	Uptime
<input type="checkbox"/>	ha-web1-ci	1	1	Cluster Member	00:00:09:05
<input type="checkbox"/>	ha-web2-ci	2	1	Cluster Member	00:00:10:05
<input type="checkbox"/>	ha-web3-ci	3	1	Cluster Member	00:00:10:25

Select an item to view details

Figure 5.1-3: Cluster Nodes Tab - New Node Added

Configure fencing (**Section 4.4.2 Add Fence Devices**) and add the node to the *failover domain* (**Section 4.4.3 Add Failover Domain**) to complete the node addition.



CCS (Method 2)

As an alternative, cluster nodes can be added from the command line using CCS. The following steps describe the process of adding a new node (*ha-web3-ci*) to an existing two node cluster.

1. Verify the current cluster status from any node:

```
# clustat
Cluster Status for ha-web-cluster @ Fri Jul 29 13:26:46 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, Local, rgmanager
ha-web2-ci           2   Online, rgmanager

Service Name         Owner (Last)      State
-----
service:ha-web-service  ha-web2-ci      started
```

2. Add the new member to the cluster configuration and specify the *nodeid*. When expanding from two nodes to three nodes (or greater), the *two_node* flag must be disabled. This can be run from any cluster node or the management server:

```
# ccs --host ha-web1 --setcman
# ccs --host ha-web1 --addnode ha-web3-ci --nodeid="3"
Node ha-web3-ci added.
```

3. Add the new node to the fence method (*Primary*) and an instance of the fence device (*IPMI-ha-web-ci*) to the fence method. Run this from any cluster node or the management server:

```
# ccs --host ha-web1 --addmethod Primary ha-web3-ci
Method Primary added to ha-web3-ci.
# ccs --host ha-web1 --addfencedev IPMI-ha-web3-ci agent=fence_ipmilan \
    auth=password ipaddr=10.16.143.233 lanplus=on login=root \
    name=IPMI-ha-web3-ci passwd=password power_wait=5 timeout=20
# ccs --host ha-web1 --addfenceinst IPMI-ha-web3-ci ha-web3-ci Primary
```

4. Add the new node to the failover domain. This can be run from any cluster node or the management server:

```
# ccs --host ha-web1 --addfailoverdomainnode ha-web-failover ha-web3-ci 3
```




5. Propagate the change to all cluster members and start the cluster services. A brief downtime is required to allow the cluster nodes to synchronize and activate the change. This can be run from any cluster node or the management server:

```
# ccs --host ha-web1 --stopall
# ccs --host ha-web1 --sync --activate
# ccs --host ha-web1 --checkconf
All nodes in sync.
# ccs --host ha-web1 --startall
Started ha-web2-ci
Started ha-web3-ci
Started ha-web1-ci
```

6. Verify the new cluster status from any node:

```
# clustat
Cluster Status for ha-web-cluster @ Fri Jul 29 13:39:42 2011
Member Status: Quorate

Member Name                ID   Status
-----
ha-web1-ci                 1   Online, Local, rgmanager
ha-web2-ci                 2   Online, rgmanager
ha-web3-ci                 3   Online, rgmanager

Service Name                Owner (Last)      State
-----
service:ha-web-service     ha-web2-ci       started
```



5.2 Removing Cluster Nodes

Conga (Method 1)

The following steps describe the process of removing one node (*ha-web3-ci*) from a running cluster using Conga and the **luci** web interface. If any cluster services are running on the node to be removed, first relocate the service as per **Section 5.3 Relocating Cluster Web Services** before proceeding.

1. Open the **luci** web interface and login using the *root* account. A node must first leave a cluster before it can be removed. Under the **Nodes** tab select **Leave Cluster** (**Figure 5.2-1: Node - Leave Cluster**):

Nodes Fence Devices Failover Domains Resources Service Groups Configure					
+ Add ↻ Reboot 🔗 Join Cluster 🔄 Leave Cluster ✕ Delete					
!	Node Name	Node ID	Votes	Status	Uptime
<input type="checkbox"/>	ha-web1-ci	1	1	Cluster Member	10:02:24:06
<input type="checkbox"/>	ha-web2-ci	2	1	Cluster Member	10:02:25:35
<input checked="" type="checkbox"/>	ha-web3-ci	3	1	Cluster Member	10:02:23:32

Select an item to view details

Figure 5.2-1: Node - Leave Cluster

2. Check the box next to the node to be removed (**Figure 5.2-2: Node - Delete**) and select **Delete**:



Nodes | Fence Devices | Failover Domains | Resources | Service Groups | Configure

+ Add | Reboot | Join Cluster | Leave Cluster | Delete

!	Node Name	Node ID	Votes	Status	Uptime
<input type="checkbox"/>	ha-web1-ci	1	1	Cluster Member	00:00:04:53
<input type="checkbox"/>	ha-web2-ci	2	1	Cluster Member	00:00:05:53
<input checked="" type="checkbox"/> !	ha-web3-ci	3	1	Not a cluster member	

Select an item to view details

Figure 5.2-2: Node - Delete

- The **Node** tab reflects the change in cluster membership (**Figure 5.2-3: Node - Removal Complete**):

Nodes | Fence Devices | Failover Domains | Resources | Service Groups | Configure

+ Add | Reboot | Join Cluster | Leave Cluster | Delete

!	Node Name	Node ID	Votes	Status	Uptime
<input type="checkbox"/>	ha-web1-ci	1	1	Cluster Member	00:00:06:24
<input type="checkbox"/>	ha-web2-ci	2	1	Cluster Member	00:00:07:24

Select an item to view details

Figure 5.2-3: Node - Removal Complete

This completes the removal of a node using Conga and the **luci** web interface.



CCS (Method 2)

As an alternative, cluster nodes can be removed using CCS. The following steps describe the process of removing one node (*ha-web3-ci*) from a running cluster.

1. Verify the current cluster status from any node:

```
# clustat
Cluster Status for ha-web-cluster @ Wed Jul 28 18:28:00 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, rgmanager
ha-web2-ci           2   Online, Local, rgmanager
ha-web3-ci           3   Online, rgmanager

Service Name         Owner (Last)      State
-----
service:ha-web-service  ha-web3-ci      started
```

2. Relocate any web services running on the node being removed to another cluster member. If no services are running on the node being removed continue to the next step. In this example the *ha-web-web* service is running on cluster node *ha-web3-ci* so it is relocated to node1 (*ha-web1-ci*) using the **clusvcadm** tool. This can be run from any cluster node but here node *ha-web2-ci* is used:

```
# clusvcadm -r ha-web-service -m ha-web1-ci
Trying to relocate service:ha-web-service to ha-web1-ci...Success
service:ha-web-service is now running on ha-web1-ci

# clustat
Cluster Status for ha-web-cluster @ Thu Jul 28 18:46:26 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, rgmanager
ha-web2-ci           2   Online, Local, rgmanager
ha-web3-ci           3   Online, rgmanager

Service Name         Owner (Last)      State
-----
service:ha-web-service  ha-web1-ci      started
```



3. Stop the cluster services on the node *ha-web3-ci* and verify the status. This can be run from any cluster node:

```
# ccs --host ha-web3-ci --stop

# clustat
Cluster Status for ha-web-cluster @ Thu Jul 28 19:19:51 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, Local, rgmanager
ha-web2-ci           2   Online, rgmanager
ha-web3-ci           3   Offline

Service Name         Owner (Last)      State
-----
service:ha-web-service  ha-web1-ci       started
```

4. Remove the node from the cluster configuration and propagate the change to the remaining cluster members. Two-node clusters are a unique case that require the ***two_node*** and ***expected_votes*** flags to be enabled. For all other configurations these flags do not need to be enabled. A brief downtime is required to allow the cluster nodes to synchronize and activate the change. This can be run from any cluster node or the management server:

```
# ccs --host ha-web1 --rmnode ha-web3-ci
# ccs --host ha-web1 --setcman two_node=1 expected_votes=1
# ccs --host ha-web1 --stopall
# ccs --host ha-web1 --sync --activate
# ccs --host ha-web1 --checkconf
All nodes in sync.
```

5. Activate the new (2-node) cluster configuration. The cluster services must be restarted when downsizing to a two node cluster configuration. This can be run from any cluster node or the management server:

```
# ccs --host ha-web1 --startall
Started ha-web2-ci
Started ha-web1-ci
```

6. Rename the cluster configuration file on *ha-web3-ci* to prevent it from being accidentally joined to the cluster by accident:

```
# mv /etc/cluster/cluster.conf /etc/cluster/cluster.conf.disabled-07-28-11
```

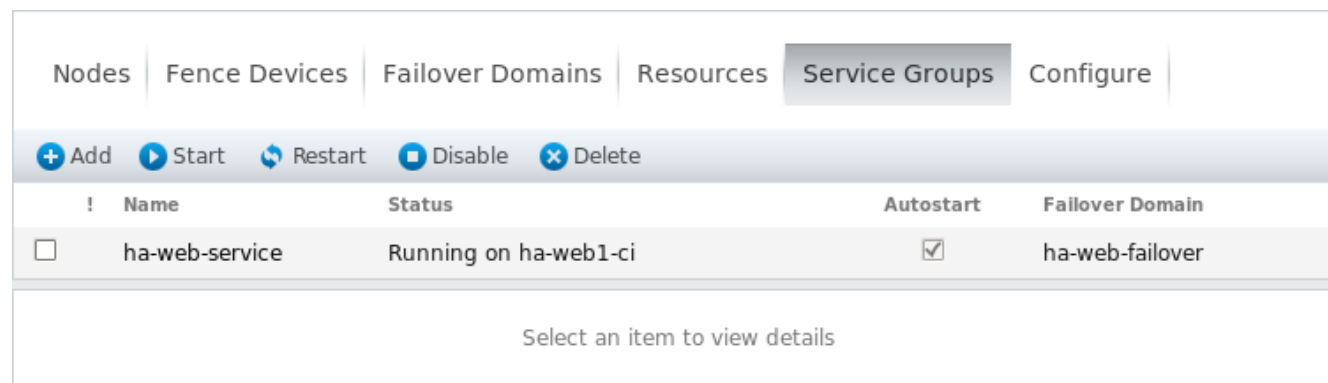


5.3 Relocating Cluster Web Services

Conga (Method 1)

The following steps describe the process of relocating a running cluster web service (*ha-web-service*) from one cluster node to another using Conga and the **luci** web interface.

1. Open the **luci** web interface and login using the *root* account. Under the **Service Groups** tab (**Figure 5.3-1: Web Service Status**) select the link for the *ha-web-service*:



Nodes	Fence Devices	Failover Domains	Resources	Service Groups	Configure
+ Add ▶ Start ↺ Restart ■ Disable ✕ Delete					
!	Name	Status	Autostart	Failover Domain	
<input type="checkbox"/>	ha-web-service	Running on ha-web1-ci	<input checked="" type="checkbox"/>	ha-web-failover	
Select an item to view details					

Figure 5.3-1: Web Service Status

Opening a browser on the the cluster status page also shows the web service as running on node *ha-web1* (**Figure 5.3-2: Web Service Before Relocate**):

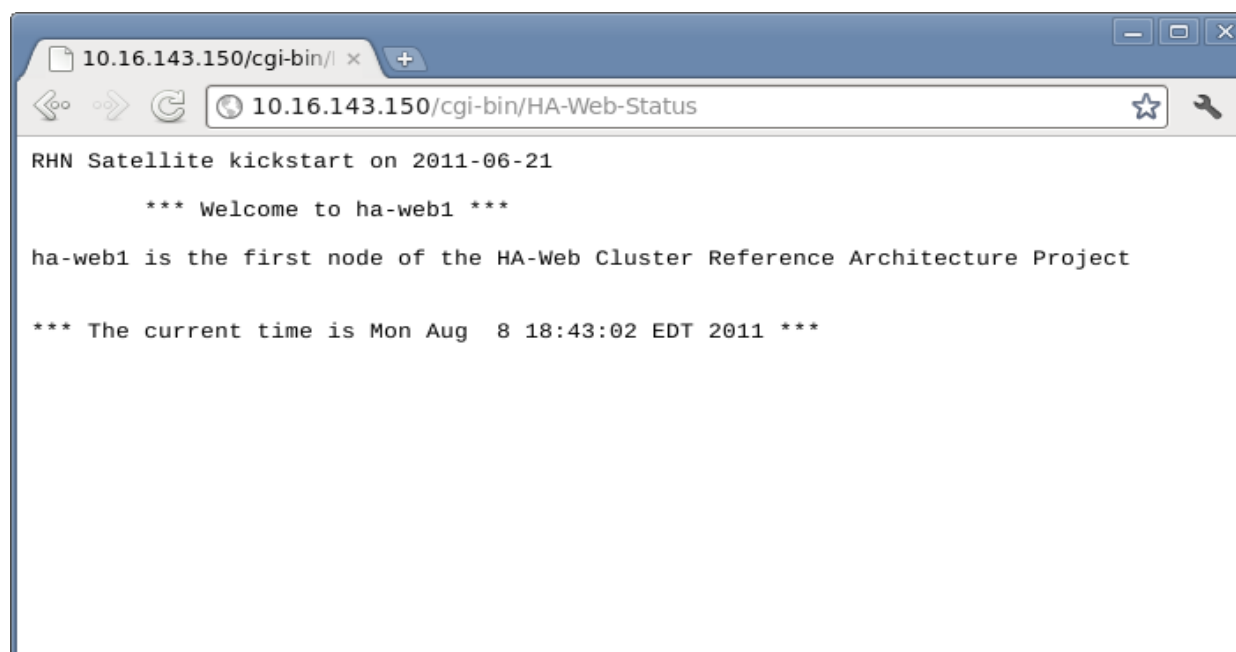


Figure 5.3-2: Web Service Before Relocate



2. Relocate the cluster web service to another cluster node by first selecting *ha-web2-ci* from the node drop-down list (**Figure 5.3-3: Web Service Relocate**). Activate the change by selecting the start icon to the right:

Nodes | Fence Devices | Failover Domains | Resources | **Service Groups** | Configure

+ Add Start Restart Disable Delete

!	Name	Status	Autostart	Failover Domain
<input type="checkbox"/>	ha-web-service	Running on ha-web1-ci	<input checked="" type="checkbox"/>	ha-web-failover

ha-web-service ▶ ↺ ■ ×

Status Running on ha-web1-ci

Figure 5.3-3: Web Service Relocate

3. The **Service Groups** tab is updated to reflect the change in service location (**Figure 5.3-4: Web Service Relocated**):

Nodes | Fence Devices | Failover Domains | Resources | **Service Groups** | Configure

+ Add Start Restart Disable Delete

!	Name	Status	Autostart	Failover Domain
<input type="checkbox"/>	ha-web-service	Running on ha-web2-ci	<input checked="" type="checkbox"/>	ha-web-failover

Select an item to view details

Figure 5.3-4: Web Service Relocated



Refreshing the cluster status page confirms the web service is now running on cluster node *ha-web2* (**Figure 5.3-5: Web Service Status After Relocate**):

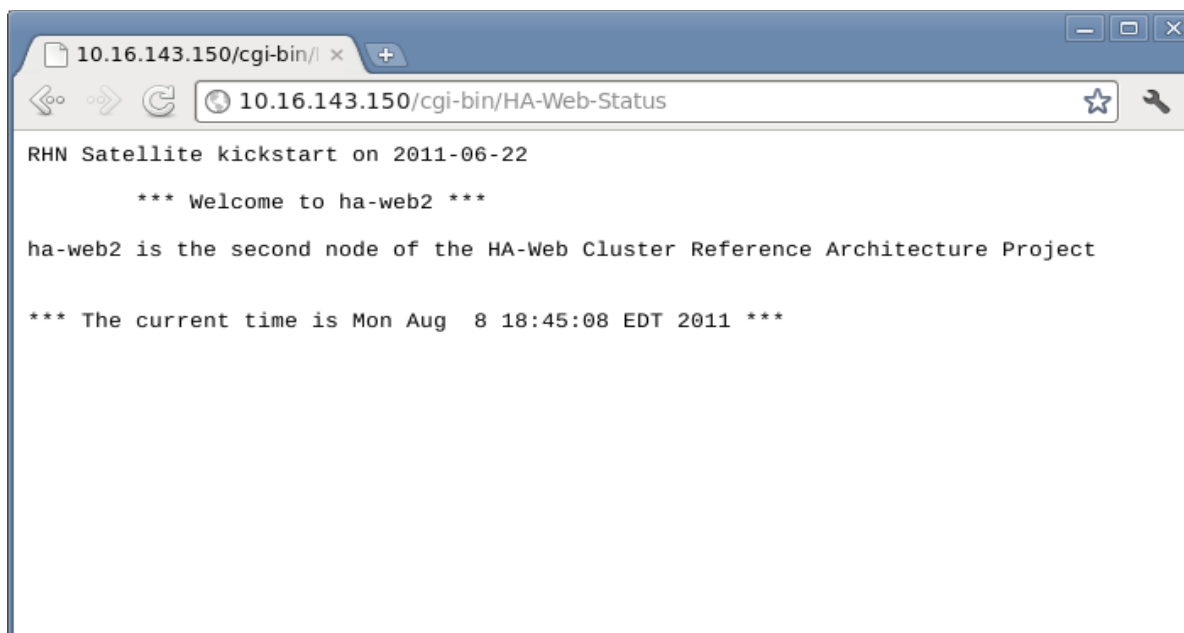


Figure 5.3-5: Web Service Status After Relocate



CCS (Method 2)

As an alternative, cluster web services can be relocated from one cluster node to another from the command line. Any cluster node can be used to relocate a service. The following steps demonstrate how to relocate the cluster web service (*ha-web-service*) from node *ha-web2-ci* to node *ha-web1-ci* using the cluster service administration (**clusvcadm**) tool on node *ha-web3-ci*.

1. Verify the cluster status and current location of the web service from node *ha-web3-ci*:

```
# clustat
Cluster Status for ha-web-cluster @ Tue Jul 26 19:30:33 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, rgmanager
ha-web2-ci           2   Online, rgmanager
ha-web3-ci           3   Online, Local, rgmanager

Service Name          Owner (Last)      State
-----
service:ha-web-service  ha-web2-ci       started
```

2. Relocate the service from node *ha-web2-ci* to node *ha-web1-ci*. This can be run this from any cluster node but in this case node *ha-web3-ci* is used:

```
# clusvcadm -r ha-web-service -m ha-web1-ci
Trying to relocate service:ha-web-service to ha-web1-ci...Success
service:ha-web-service is now running on ha-web1-ci
```

3. Verify the cluster status and new location of the web service from node (*ha-web3-ci*):

```
# clustat
Cluster Status for ha-web-cluster @ Tue Jul 26 19:43:24 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, rgmanager
ha-web2-ci           2   Online, rgmanager
ha-web3-ci           3   Online, Local, rgmanager

Service Name          Owner (Last)      State
-----
service:ha-web-service  ha-web1-ci       started
```



5.4 Fencing Cluster Nodes

Fencing a node can be done to test the fencing method in use. Only one cluster node at a time should be fenced to prevent potential loss of quorum. In the following steps, IPMI LAN fencing is tested by fencing node1 (*ha-web1-ci*) and forcing the web service to relocate. Since the fencing briefly powers down node1 all fencing and status commands should be run from either node2 (*ha-web2-ci*) or node3 (*ha-web3-ci*). The fencing of nodes can only be performed from the command line (CLI).

1. Verify the current status of the web service (*ha-web-service*) from node3 (*ha-web3-ci*):

```
# clustat
Cluster Status for ha-web-cluster @ Wed Jul 27 11:18:00 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, rgmanager
ha-web2-ci           2   Online, rgmanager
ha-web3-ci           3   Online, Local, rgmanager

Service Name          Owner (Last)      State
-----
service:ha-web-service  ha-web1-ci       started
```

2. Issue the fencing command from node3 (*ha-web3-ci*). Optionally the verbose mode option (*-vv*) can be specified:

```
# fence_node -vv ha-web1-ci
fence ha-web1-ci dev 0.0 agent fence_ipmilan result: success
agent args: nodename=ha-web1-ci agent=fence_ipmilan auth=password
ipaddr=10.16.143.231 lanplus=on login=root passwd=password power_wait=5
timeout=20
fence ha-web1-ci success
```

3. Check the status as node1 (*ha-web1-ci*) is powered down, the web service (*ha-web-service*) is relocated to node2 (*ha-web2-ci*) and node1 (*ha-web1-ci*) has fully restarted:

```
# clustat
Cluster Status for ha-web-cluster @ Wed Jul 27 11:18:26 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Offline
ha-web2-ci           2   Online, rgmanager
ha-web3-ci           3   Online, Local, rgmanager

Service Name          Owner (Last)      State
-----
service:ha-web-service  ha-web2-ci       starting
```



The web service (*ha-web-service*) has been relocated to node2 (*ha-web2-ci*), has *started* and is available, but node1 (*ha-web1-ci*) is still booting and *Offline*:

```
# clustat
Cluster Status for ha-web-cluster @ Wed Jul 27 11:18:54 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Offline
ha-web2-ci           2   Online, rgmanager
ha-web3-ci           3   Online, Local, rgmanager

Service Name         Owner (Last)      State
-----
service:ha-web-service  ha-web2-ci      started
```

After cluster node1 (*ha-web1-ci*) has completed booting, verify it is a member of the cluster:

```
# clustat
Cluster Status for ha-web-cluster @ Wed Jul 27 11:22:20 2011
Member Status: Quorate

Member Name          ID   Status
-----
ha-web1-ci           1   Online, rgmanager
ha-web2-ci           2   Online, rgmanager
ha-web3-ci           3   Online, Local, rgmanager

Service Name         Owner (Last)      State
-----
service:ha-web-service  ha-web2-ci      started
```



5.5 Importing a Cluster

Importing a cluster is used when a configuration exists but the Conga (**luci**) management interface has not imported the configuration into the internal **luci** database. In situations where CCS was used to build a cluster, **luci** initially is not aware of the cluster until the configuration has been imported. Perform the following steps to import a cluster into the **luci** interface:

1. Open the **luci** web interface and login using the *root* account. Under **Manage Clusters** select **Add** (**Figure 5.5-1: Cluster Import**):

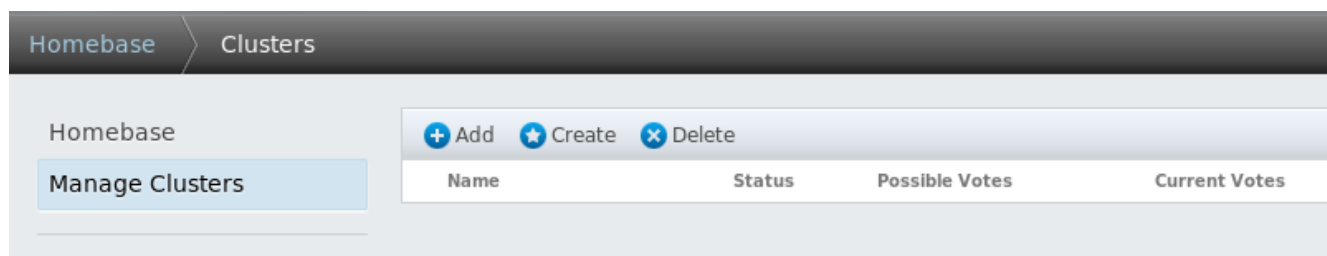


Figure 5.5-1: Cluster Import

2. The **Add Existing Cluster** dialog box opens. Populate the fields as shown below in **Figure 5.5-2: Add Existing Cluster**. Select **Add Cluster** when complete.

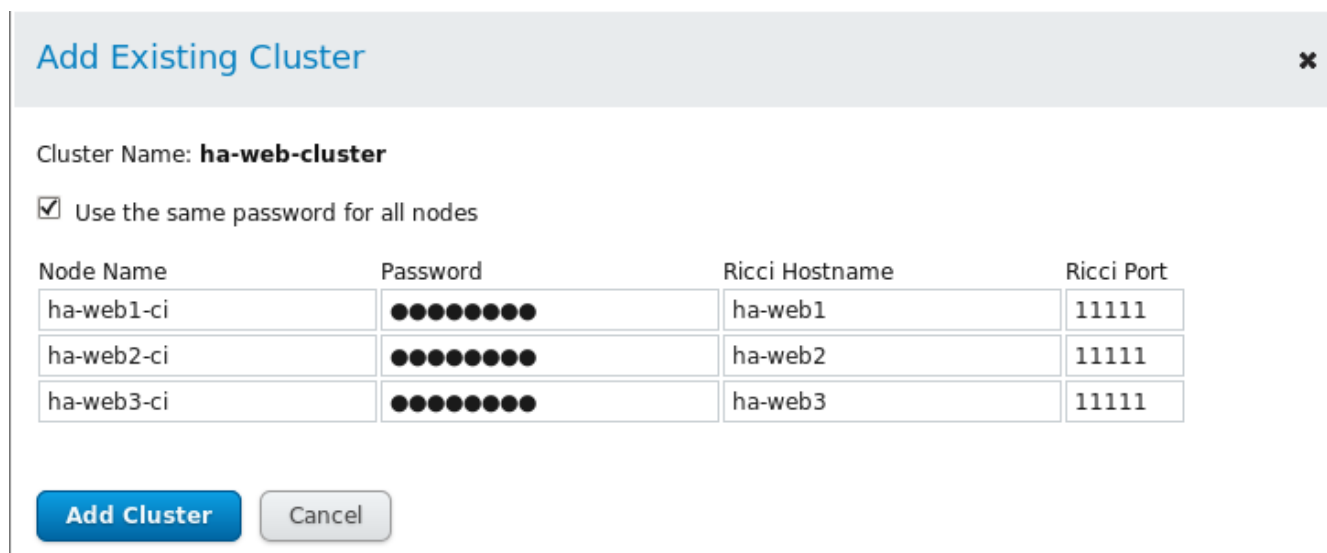


Figure 5.5-2: Add Existing Cluster



3. The Homebase confirms the import was successful (**Figure 5.5-3: Cluster Import Successful**):
4. The cluster is now imported and can be managed through the **luci** interface (**Figure 5.5-4: Cluster Node Status**):

The screenshot shows the Homebase interface for managing clusters. The breadcrumb trail is Homebase > Clusters > ha-web-cluster. On the left, there is a sidebar with 'Homebase' and 'Manage Clusters' sections. Under 'Manage Clusters', a green dot indicates the 'ha-web-cluster' is active. The main content area has tabs for 'Nodes', 'Fence Devices', 'Failover Domains', 'Resources', and 'Service Groups'. Below the tabs are action buttons: '+ Add', 'Reboot', 'Join Cluster', 'Leave Cluster', and 'Delete'. A table lists the nodes:

	! Node Name	Node ID	Votes	Status
<input type="checkbox"/>	ha-web1-ci	1	1	Cluster Member
<input type="checkbox"/>	ha-web2-ci	2	1	Cluster Member
<input type="checkbox"/>	ha-web3-ci	3	1	Cluster Member

Figure 5.5-4: Cluster Node Status

The screenshot shows the Homebase interface for managing clusters. The breadcrumb trail is Homebase > Clusters > ha-web-cluster. On the left, there is a sidebar with 'Homebase' and 'Manage Clusters' sections. Under 'Manage Clusters', a green dot indicates the 'ha-web-cluster' is active. The main content area has tabs for 'Nodes', 'Fence Devices', 'Failover Domains', 'Resources', and 'Service Groups'. Below the tabs are action buttons: '+ Add', 'Reboot', 'Join Cluster', 'Leave Cluster', and 'Delete'. A table lists the nodes:

CLUSTER SUMMARY			
Name	Status	Nodes Joined	
ha-web-cluster	Quorate	3 of 3	

Figure 5.5-3: Cluster Import Successful



6 Conclusion

Red Hat's High Availability Add-On for Red Hat Enterprise Linux 6 is Red Hat's premier clustering solution. The High Availability Add-On provides reliability, availability and scalability (RAS) to critical production services by eliminating single points of failure and providing automatic *failover* of those services in the event of a cluster node failure or error condition.

This reference architecture detailed the deployment, configuration and management of a highly available web service using the High Availability Add-On for Red Hat Enterprise Linux. A three node cluster was deployed and configured to use a highly available LVM (HA-LVM) volume for the cluster web service contents. A dedicated management station was deployed to manage and configure the cluster. This configuration can be scaled out by adding more cluster nodes or configured to run other cluster services as required to best meet the needs of specific computing environments.

Cluster creation and management was demonstrated using both Conga and the **luci** graphical user interface (GUI), and the Cluster Configuration System (CCS) command line interface (CLI). Both methods can be used interchangeably to create, configure and manage a cluster.

The most common cluster management tasks (adding, removing nodes, relocating cluster services, fencing nodes, importing an existing cluster) were also demonstrated.



Appendix A: References

1. Red Hat Enterprise Linux 6 Installation Guide
Installing Red Hat Enterprise Linux 6 for all architectures
Edition 1.0
http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/pdf/Installation_Guide/Red_Hat_Enterprise_Linux-6-Installation_Guide-en-US.pdf
2. Red Hat Enterprise Linux 6 Deployment Guide
Deployment, Configuration and Administration of Red Hat Enterprise Linux 6
Edition 1.0
http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/pdf/Deployment_Guide/Red_Hat_Enterprise_Linux-6-Deployment_Guide-en-US.pdf
3. Red Hat Enterprise Linux 6 Virtualization Guide
Guide to Virtualization on Red Hat Enterprise Linux 6
Edition 3.2
http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Virtualization/index.html
4. Red Hat Enterprise Linux 6 Cluster Administration
Configuring and Managing the High Availability Add-On
http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/pdf/Cluster_Administration/Red_Hat_Enterprise_Linux-6-Cluster_Administration-en-US.pdf
5. Red Hat Enterprise Linux 6 Logical Volume Manager Administration
LVM Administrator Guide
http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/pdf/Logical_Volume_Manager_Administration/Red_Hat_Enterprise_Linux-6-Logical_Volume_Manager_Administration-en-US.pdf
6. Red Hat Enterprise Linux 6 DM Multipath
DM Multipath Configuration and Administration
http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/pdf/DM_Multipath/Red_Hat_Enterprise_Linux-6-DM_Multipath-en-US.pdf
7. “What is a Highly Available LVM (HA-LVM) configuration and how do I implement it?”
Red Hat Knowledge Base Article - KB3068
<https://access.redhat.com/kb/docs/DOC-3068>
8. “Using SCSI Persistent Reservations with Red Hat Enterprise Linux 6 with the High Availability Add-On”
Red Hat Knowledge Base Article – KB43079
<https://access.redhat.com/kb/docs/DOC-43079>



Appendix B: Red Hat Enterprise Linux 6 – Satellite Configuration Details

The deployment of Red Hat Enterprise Linux 6 on the clusters nodes was done using a local Satellite server connected to Red Hat Network (RHN). The following steps describe configuring the Satellite server in preparation of the cluster node deployments.

1. Open the Satellite server web interface and login (**Figure 1: Satellite Overview**):

The screenshot shows the Red Hat Network Satellite web interface. At the top, there is a header with the language set to English, links to Knowledgebase and Documentation, and user information for 'admin' at 'Red Hat, Inc.' with links for Preferences and Sign Out. Below the header is a search bar and a navigation menu with options: Overview, Systems, Errata, Channels, Configuration, Schedule, Users, Admin, and Help. The main content area is titled 'Overview' and contains several sections:

- Tasks:** A list of tasks including 'Manage Entitlements and Subscriptions: My Organization | RHN Satellite-Wide', 'Register Systems', 'Manage Activation Keys', 'Manage Kickstarts', 'Manage Configuration Files', 'Manage RHN Satellite Organizations', and 'Configure RHN Satellite'.
- Inactive Systems:** A table listing inactive systems with their names and the time they have been inactive.

System Name	Inactive Time
cf-cloudforms1.cloud.lab.eng.bos.redhat.com	1 Day(s)
cf-cloudforms1.cloud.lab.eng.bos.redhat.com	1 Day(s)
cf-cloudforms2.cloud.lab.eng.bos.redhat.com	1 Day(s)
cf-cloudengine.cloud.lab.eng.bos.redhat.com	2 Day(s)
cf-cloudforms5.cloud.lab.eng.bos.redhat.com	2 Day(s)

At the bottom of the Inactive Systems section, there is a link: [View All Inactive Systems \(55\)](#).

Figure 1: Satellite Overview



2. Create an Activation Key. Under the **Systems** tab select **Activation Keys** (**Figure 2: Activation Keys**):

The screenshot shows the Red Hat Network Satellite web interface. At the top, there is a navigation bar with the following tabs: Overview, Systems (selected), Errata, Channels, Configuration, Schedule, Users, Admin, and Help. To the right of the navigation bar, there is a search box with the text "Systems" and a "Search" button. Below the navigation bar, there is a red banner with the text "NO SYSTEMS SELECTED" and two buttons: "MANAGE" and "CLEAR". On the left side, there is a sidebar menu with the following items: Overview, Systems, System Groups, System Set Manager, Advanced Search, Activation Keys (selected), Stored Profiles, Custom System Info, and Kickstart. The main content area is titled "Activation Keys" and contains the following text: "Activation Keys are used to register systems. Systems registered with an activation key will inherit the characteristics defined by that key." Below this, there is a section titled "Universal Default" with the following text: "If a universal default activation key is set for your organization, then systems registered to your organization will inherit the properties of that key by default without the need to explicitly specify that key during registration. You do not currently have a universal default activation key set. To set a key as the universal default, please visit the details page of that key and check off the 'Universal Default?' checkbox." At the bottom, there is a section titled "All Activation Keys" with the following text: "The following activation keys have been created for use by your organization."

Figure 2: Activation Keys

Select **create new key**.



- The **Create Activation Key** screen is displayed (**Figure 3: Activation Key Details**). Complete the fields as shown for **Description**, **Key**, **Base Channels**, **Add-On Entitlements**. Leave all other fields as defaults:

Details Child Channels Packages Configuration Groups Activated Systems

Activation Key Details

Systems registered with this activation key will inherit the settings listed below.

Description	<input type="text" value="ha-web-cluster-nodes-key"/> Tip: Use this to describe what kind of settings this key will reflect on systems that use it. If left blank, this field will be filled in 'None'.
Key:	1- <input type="text" value="ha-web-cluster-nodes"/> Tip: Leave blank for automatic key generation. Note that the prefix is an indication of the RHN Satellite organization the key is associated with.
Usage:	<input type="text"/> Tip: Leave blank for unlimited use.
Base Channels:	<input type="text" value="Red Hat Enterprise Linux Server (v. 6 for 64-bit x86_64)"/> Tip: Choose "RHN Satellite Default" to allow systems to register to the default Red Hat provided channel that corresponds to their installed version of Red Hat Enterprise Linux. You may also choose particular Red Hat provided channels or custom base channels here, but please note if a system using this key is not compatible with the selected channel, it will fall back to its Red Hat default channel.
Add-On Entitlements:	<input checked="" type="checkbox"/> Monitoring <input checked="" type="checkbox"/> Provisioning <input type="checkbox"/> Virtualization <input type="checkbox"/> Virtualization Platform
Configuration File Deployment:	<input type="checkbox"/> Tip: Deploy configuration files to systems when they are registered with this activation key.
Universal Default:	<input type="checkbox"/> Tip: Only one universal default activation key may be set for this organization. By setting this key as universal default, you will remove universal default status from the current universal default key if it exists. If this key is set as universal default, then newly-registered systems to your organization will inherit the properties of this key.

Figure 3: Activation Key Details

Select **Create Activation Key** when done.



4. Select Child Channels. Under the **Child Channels** tab select the channels required to deploy the cluster nodes (**Figure 4: Child Channels**):
 - RHEL Server High Availability (v. 6 for 64-bit x86_64)
 - RHN Tools for RHEL (v. 6 for 64-bit x86_64)

ha-web-cluster-nodes-key delete key

Details **Child Channels** Packages Configuration Groups Activated Systems

Any system registered using this activation key will be subscribed to the selected child channels.

The following child channels of **Red Hat Enterprise Linux Server (v. 6 for 64-bit x86_64)** can be associated with this activation key.

- RHEL Server High Availability (v. 6 for 64-bit x86_64)
- RHEL Server Optional (v. 6 64-bit x86_64)
- RHEL Server Resilient Storage (v. 6 for 64-bit x86_64)
- RHEL Server Supplementary (v. 6 64-bit x86_64)
- RHEL V2VWIN (v. 6 for 64-bit x86_64)
- RHN Tools for RHEL (v. 6 for 64-bit x86_64)

Update Key

Figure 4: Child Channels

Select **Update Key** when done.



5. Create a Kickstart profile. Under the Systems tab select **Kickstart (Figure 5: Create New Kickstart Profile)**:

The screenshot shows the Red Hat Network Satellite interface. The top navigation bar includes 'Overview', 'Systems', 'Errata', 'Channels', 'Configuration', 'Schedule', 'Users', 'Admin', and 'Help'. The 'Systems' tab is active, and the page title is 'Kickstart Overview'. The main content area is divided into three sections:

- Kickstart Summary:** A table showing the number of profiles for different operating systems.
- Kickstart Actions:** A list of links for managing profiles.
- Systems Currently Kickstarting:** A table showing the status and last change time for systems currently being kickstarted.

Kickstarting Systems	Status	Last Change
Unregistered System	Kickstart complete.	38 days ago
Unregistered System	Kickstart complete.	151 days ago
Unregistered System	Kickstart complete.	6 days ago
Unregistered System	Kickstart complete.	13 days ago

Figure 5: Create New Kickstart Profile

Select **create new kickstart profile**.



6. The Create Kickstart Profile page opens (**Figure 6: Create Kickstart Profile - Step 1**). Enter **Label**, **Base Channel** and **Kickstartable Tree**:

- Overview
- Systems
- System Groups
- System Set Manager
- Advanced Search
- Activation Keys
- Stored Profiles
- Custom System Info
- Kickstart**
- Profiles
- Bare Metal
- GPG and SSL Keys
- Distributions
- File Preservation
- Kickstart Snippets

Step 1: Create Kickstart Profile

A kickstart file is a simple text file containing a list of items, each identified by a keyword, that answers the questions an installer needs in order to successfully install Red Hat Enterprise Linux. A kickstart profile includes a kickstart file, as well as other saved options such as the version of Red Hat Enterprise Linux to be installed and the location of the installation files.

Label*:	<input type="text" value="ha-web-cluster-nodes"/>
Base Channel*:	<input type="text" value="Red Hat Enterprise Linux Server (v. 6 for 64-bit x86_64)"/>
Kickstartable Tree*:	<input type="text" value="ks-rhel-x86_64-server-6-6.1"/>
Virtualization Type:	<input type="text" value="None"/>

Figure 6: Create Kickstart Profile - Step 1

Click **Next** when done.



7. The Distribution File Location page (**Figure 7: Create Kickstart Profile - Step 2**) opens. Accept all defaults:

Step 2: Distribution File Location

In order to install Red Hat Enterprise Linux during kickstart, the Red Hat Enterprise Linux installation files must be available to the kickstarting system via network connection. You can make these available either via http or ftp.

Default Download Location: /ks/dist/ks-rhel-x86_64-server-6-6.1

Custom Download Location:

Figure 7: Create Kickstart Profile - Step 2

Select **Next** to continue.



8. The Root Password page opens (**Figure 8: Create Kickstart Profile - Step 3**). Enter the root password for the cluster nodes:

Step 3: Root Password

The kickstart profile will need to have a root password for the system(s) to be kickstarted in order to complete the kickstart process.

New Root Password*:

Verify New Root Password*:

[Previous](#) [Finish](#)

Figure 8: Create Kickstart Profile - Step 3

Select **Finish** when done.



9. Enable SELinux. Under the **Kickstart details** tab select **Advanced Options**. Next select the *ha-web-cluster-nodes* profile (**Figure 9: Kickstart Profile**):

Overview
Systems
System Groups
System Set Manager
Advanced Search
Activation Keys
Stored Profiles
Custom System Info
Kickstart
Profiles
Bare Metal
GPG and SSL Keys
Distributions
File Preservation
Kickstart Snippets

Kickstart: ha-web-cluster-nodes

[clone kickstart](#) [delete kickstart](#)

Kickstart Details System Details Software Activation Keys Scripts Kickstart File

Details Operating System Variables Advanced Options Bare Metal Kickstart

Advanced Options

You may wish to modify the following options for special-case kickstart scenarios. We recommend consulting the Red Hat Enterprise Linux System Administration Guide for detailed documentation of all of these options.

auth*:

autopart:

autostep:

bootloader*:

Figure 9: Kickstart Profile

Scroll down to the option **selinux:** and enable it (**Figure 10: SELinux Enforcing**):

selinux:

Figure 10: SELinux Enforcing

Select **Update Kickstart** when done.



10. Associate the Activation Key with the kickstart profile. Under the **Activation Keys** tab select the check box next to *ha-web-cluster-nodes-key* (**Figure 11: Associate Activation Key to Kickstart Profile**):

The screenshot shows the RHN Satellite web interface. The browser address bar indicates the URL: `https://ra-ns1.cloud.lab.eng.bos.redhat.com/rhn/kickstart/ActivationKeysList.do?`. The page title is "Kickstart: ha-web-cluster-nodes". The "Activation Keys" tab is selected, showing a list of 10 activation keys. The key "ha-web-cluster-nodes-key" is selected, indicated by a checked checkbox. Below the table, there are "Update List" and "Select All" buttons. A note and a warning are also present.

Description	Key	Usage
<input type="checkbox"/> RHEL 6	1-9abcdef8abcdef7abcdef	(unlimited)
<input type="checkbox"/> cf-rhel61	1-cf-rhel61	(unlimited)
<input type="checkbox"/> cf-rhel61-rhn	1-cf-rhel61-rhn	(unlimited)
<input type="checkbox"/> cf-rhel61-rhn-new	1-cf-rhel61-rhn-new	(unlimited)
<input type="checkbox"/> cf-rhev	1-cfrhev	(unlimited)
<input type="checkbox"/> cluster-nodes	1-ha-cluster-nodes	(unlimited)
<input type="checkbox"/> ha-luci-key	1-haluci	(unlimited)
<input checked="" type="checkbox"/> ha-web-cluster-nodes-key	1-ha-web-cluster-nodes	(unlimited)
<input type="checkbox"/> ha-web-cluster-nodes-key2	1-80fe4280c31442d43549c823355b2a78	(unlimited)
<input type="checkbox"/> rhel61	1-rhel61	(unlimited)

Figure 11: Associate Activation Key to Kickstart Profile

Select **Update Activation Key**. This completes the configuration of the Satellite server in preparation of the cluster node deployments.



Appendix C: Fibre Channel Storage Provisioning

An **HA-LVM** volume (*LVM2*) is used for storing the content of the cluster web service. The shared volume is created on an *HP StorageWorks MSA2324fc* Fibre Channel storage array. The array contains a single controller (*Ports A1, A2*) and an *MSA70* expansion shelf providing a total of 48 physical drives. The steps below describe how to provision a 20GB volume (**ha-web-data-01**) within a new virtual disk (*VD1*) from the command line.

Step 1. Login to the MSA storage array

```
# ssh -Y -l manage ra-msa20
```

Step 2. View the available virtual disks (*vdisk*), physical disks and volumes

```
# show vdisk
# show disk
# show volumes
```

Step 3. Create a virtual disk to hold the volume

```
# create vdisk level raid6 disks 1.1-12 VD1
```

Step 4. Create a volume within the virtual disk

```
# create volume vdisk VD1 size 20GB access no-access lun 1 ha-web-data-01
```

Step 5. View the current host connection mappings

```
# show hosts
```

Host ID	Name	Discovered	Mapped	Profile
50060B0000C28608	rhelh-01_1	Yes	No	Standard
50060B0000C2860A	rhelh-01_2	Yes	No	Standard
50060B0000C2863E	bkup_host1	Yes	No	Standard
50060B0000C2863C	bkup_host0	Yes	No	Standard
50060B0000C28614	rhevh-01_1	Yes	No	Standard
50060B0000C28644		Yes	No	Standard
50060B0000C28616	rhevh-01_2	Yes	No	Standard
50060B0000C28646		Yes	No	Standard
50060B0000C28600	mgmt1_host0	Yes	No	Standard
50060B0000C28602	mgmt1_host1	Yes	No	Standard
50060B0000C28640		Yes	No	Standard
50060B0000C28642		Yes	No	Standard
50060B0000C2861A		Yes	No	Standard
50060B0000C28634		Yes	No	Standard
50060B0000C28636		Yes	No	Standard
50060B0000C2862C		Yes	No	Standard
50060B0000C2862E		Yes	No	Standard
50060B0000C28628		Yes	No	Standard
50060B0000C2862A		Yes	No	Standard
50060B0000C28612		Yes	No	Standard



50060B0000C28610	Yes	No	Standard
21FD00051E597C97	Yes	No	Standard
50060B0000C2863A test1_host1	No	No	Standard
50060B0000C28638 test1_host0	No	No	Standard
5001438000C3E622 monet_host0	No	No	Standard
5001438000C3E67E monet_host1	No	No	Standard
50060B0000C28632 rhevh-02_2	No	No	Standard
50060B0000C28630 rhevh-02_1	No	No	Standard
50060B0000C28626 rhelh-02_2	No	No	Standard
50060B0000C28624 rhelh-02_1	No	No	Standard
50060B0000C28606 mgmt2_host1	No	No	Standard
50060B0000C28604 mgmt2_host0	No	No	Standard

Step 6. Identify what ports each host is using

ha-web1:

```
# cat /sys/class/fc_host/host0/port_name
0x50060b0000c28628
# cat /sys/class/fc_host/host1/port_name
0x50060b0000c2862A
```

ha-web2:

```
# cat /sys/class/fc_host/host0/port_name
0x50060b0000c2862C
# cat /sys/class/fc_host/host1/port_name
0x50060b0000c2862E
```

ha-web3:

```
# cat /sys/class/fc_host/host0/port_name
0x50060b0000c28634
# cat /sys/class/fc_host/host1/port_name
0x50060b0000c28636
```

Note: If the Fibre Channel storage array has two controllers attached to the SAN fabric then each host has four port connections instead of the two shown here.

Step 7. Configure the host-name id's to the ports and verify

```
# set host-name id 50060B0000C28628 ha-web1-host0 # Controller A Port 1
# set host-name id 50060B0000C2862A ha-web1-host1 # Controller A Port 2
# set host-name id 50060B0000C2862C ha-web2-host0 # Controller A Port 1
# set host-name id 50060B0000C2862E ha-web2-host1 # Controller A Port 2
# set host-name id 50060B0000C28634 ha-web3-host0 # Controller A Port 1
# set host-name id 50060B0000C28636 ha-web3-host1 # Controller A Port 2
# show hosts
Host ID          Name              Discovered Mapped Profile
-----
50060B0000C28608 rhelh-01_1       Yes        No        Standard
50060B0000C2860A rhelh-01_2       Yes        No        Standard
50060B0000C2863E bkup_host1       Yes        No        Standard
50060B0000C2863C bkup_host0       Yes        No        Standard
50060B0000C28614 rhevh-01_1       Yes        No        Standard
50060B0000C28644                Yes        No        Standard
50060B0000C28616 rhevh-01_2       Yes        No        Standard
```



50060B0000C28646		Yes	No	Standard
50060B0000C28600	mgmt1_host0	Yes	No	Standard
50060B0000C28602	mgmt1_host1	Yes	No	Standard
50060B0000C28640		Yes	No	Standard
50060B0000C28642		Yes	No	Standard
50060B0000C2861A		Yes	No	Standard
50060B0000C28634	ha-web3-host0	Yes	No	Standard
50060B0000C28636	ha-web3-host1	Yes	No	Standard
50060B0000C2862C	ha-web2-host0	Yes	No	Standard
50060B0000C2862E	ha-web2-host1	Yes	No	Standard
50060B0000C28628	ha-web1-host0	Yes	No	Standard
50060B0000C2862A	ha-web1-host1	Yes	No	Standard
50060B0000C28612		Yes	No	Standard
50060B0000C28610		Yes	No	Standard
21FD00051E597C97		Yes	No	Standard

Step 8. View the volumes

```
# show volumes
Vdisk Name          Size   Serial Number          WR Policy
-----
VD1   ha-web-data-01 19.9GB 00c0ffdab7f400005ce8f54d01000000  write-back
-----
```

Step 9. Restrict volume access to the three cluster nodes

```
# map volume access read-write lun 1 ports A1,A2 \
  host ha-web1-host0, ha-web1-host1, \
       ha-web2-host0, ha-web2-host1, \
       ha-web3-host0, ha-web3-host1 \
       ha-web-data-01
Info: Command completed successfully. - Mapping succeeded.
Host ha-web1-host0 was mapped for volume ha-web-data-01 with LUN 1.
Info: Command completed successfully. - Mapping succeeded.
Host ha-web1-host1 was mapped for volume ha-web-data-01 with LUN 1.
Info: Command completed successfully. - Mapping succeeded.
Host ha-web2-host0 was mapped for volume ha-web-data-01 with LUN 1.
Info: Command completed successfully. - Mapping succeeded.
Host ha-web2-host1 was mapped for volume ha-web-data-01 with LUN 1.
Info: Command completed successfully. - Mapping succeeded.
Host ha-web3-host0 was mapped for volume ha-web-data-01 with LUN 1.
Info: Command completed successfully. - Mapping succeeded.
Host ha-web3-host1 was mapped for volume ha-web-data-01 with LUN 1.
Success: Command completed successfully. - The volume(s) were mapped
successfully.
```

Step 10. Verify volume and host mappings

```
# show volume-map
Info: Retrieving data...
Volume View [Serial Number (00c0ffdab7f400005ce8f54d01000000) Name (ha-web-
data-01) ] Mapping:
  Ports LUN   Access          Host-Port-Identifer Nickname          Profile
-----
  A1,A2 1       read-write      50060B0000C28628    ha-web1-host0    Standard
  A1,A2 1       read-write      50060B0000C2862A    ha-web1-host1    Standard
```



```

A1,A2 1      read-write  50060B0000C2862C    ha-web2-host0 Standard
A1,A2 1      read-write  50060B0000C2862E    ha-web2-host1 Standard
A1,A2 1      read-write  50060B0000C28634    ha-web3-host0 Standard
A1,A2 1      read-write  50060B0000C28636    ha-web3-host1 Standard
                not-mapped  all other hosts          Standard
# show host-map
Host View [ID (50060B0000C28634) Name (ha-web3-host0) Profile (Standard) ]
Mapping:
  Name                Serial Number                LUN  Access  Ports
  -----
  ha-web-data-01 00c0ff00dab7f400005ce8f54d01000000 1    read-write  A1,A2

Host View [ID (50060B0000C28636) Name (ha-web3-host1) Profile (Standard) ]
Mapping:
  Name                Serial Number                LUN  Access  Ports
  -----
  ha-web-data-01 00c0ff00dab7f400005ce8f54d01000000 1    read-write  A1,A2

Host View [ID (50060B0000C2862C) Name (ha-web2-host0) Profile (Standard) ]
Mapping:
  Name                Serial Number                LUN  Access  Ports
  -----
  ha-web-data-01 00c0ff00dab7f400005ce8f54d01000000 1    read-write  A1,A2

Host View [ID (50060B0000C2862E) Name (ha-web2-host1) Profile (Standard) ]
Mapping:
  Name                Serial Number                LUN  Access  Ports
  -----
  ha-web-data-01 00c0ff00dab7f400005ce8f54d01000000 1    read-write  A1,A2

```

Step 11. From each of the cluster nodes, determine which device files are configured for the 20 GB Fibre Channel disk - `/dev/sdb`, `/dev/sdc`, `/dev/sdd`, `/dev/sde`

```

# fdisk -l 2>/dev/null | grep "^Disk /dev/sd"
Disk /dev/sda: 146.8 GB, 146778685440 bytes
Disk /dev/sdb: 20.0 GB, 19999997952 bytes
Disk /dev/sdc: 20.0 GB, 19999997952 bytes
Disk /dev/sdd: 20.0 GB, 19999997952 bytes
Disk /dev/sde: 20.0 GB, 19999997952 bytes

```

Step 12. Verify the World Wide ID's (WWID) match for each device. The WWID's must be the same across each cluster node

```

# /lib/udev/scsi_id --whitelisted --device=/dev/sdb
3600c0ff000dab7f45ce8f54d01000000
# /lib/udev/scsi_id --whitelisted --device=/dev/sdc
3600c0ff000dab7f45ce8f54d01000000
# /lib/udev/scsi_id --whitelisted --device=/dev/sdd
3600c0ff000dab7f45ce8f54d01000000
# /lib/udev/scsi_id --whitelisted --device=/dev/sde
3600c0ff000dab7f45ce8f54d01000000

```



Appendix D: Cluster Configuration File (cluster.conf)

```
<?xml version="1.0"?>
<cluster config_version="67" name="ha-web-cluster">
  <clusternodes>
    <clusternode name="ha-web1-ci" nodeid="1">
      <fence>
        <method name="Primary">
          <device name="IPMI-ha-web1-ci"/>
        </method>
      </fence>
    </clusternode>
    <clusternode name="ha-web2-ci" nodeid="2">
      <fence>
        <method name="Primary">
          <device name="IPMI-ha-web2-ci"/>
        </method>
      </fence>
    </clusternode>
    <clusternode name="ha-web3-ci" nodeid="3">
      <fence>
        <method name="Primary">
          <device name="IPMI-ha-web3-ci"/>
        </method>
      </fence>
    </clusternode>
  </clusternodes>
  <fencedevices>
    <fencedevice agent="fence_ipmilan" auth="password"
      ipaddr="10.16.143.231" lanplus="on" login="root"
      name="IPMI-ha-web1-ci" passwd="password"
      power_wait="5" timeout="20"/>
    <fencedevice agent="fence_ipmilan" auth="password"
      ipaddr="10.16.143.232" lanplus="on" login="root"
      name="IPMI-ha-web2-ci" passwd="password"
      power_wait="5" timeout="20"/>
    <fencedevice agent="fence_ipmilan" auth="password"
      ipaddr="10.16.143.233" lanplus="on" login="root"
      name="IPMI-ha-web3-ci" passwd="password"
      power_wait="5" timeout="20"/>
  </fencedevices>
  <rm>
    <failoverdomains>
      <failoverdomain name="ha-web-failover" nofailback="0"
        ordered="1" restricted="0">
        <failoverdomainnode name="ha-web1-ci" priority="1"/>
        <failoverdomainnode name="ha-web2-ci" priority="2"/>
        <failoverdomainnode name="ha-web3-ci" priority="3"/>
      </failoverdomain>
    </failoverdomains>
  </rm>
</resources>
```



```
<ip address="10.16.143.150" monitor_link="on" sleeptime="10"/>
<lvm lv_name="ha-web-lvol1" name="ha-web-HA-LVM"
  self_fence="on" vg_name="HA-Web-VG"/>
<fs device="/dev/HA-Web-VG/ha-web-lvol1" fsid="56432"
  fstype="ext4" mountpoint="/ha/ha-web"
  name="ha-web-filesystem" self_fence="on"/>
<script file="/etc/init.d/httpd" name="ha-web-apache-script"/>
</resources>
<service domain="ha-web-failover"
  max_restarts="3" name="ha-web-service"
  recovery="restart" restart_expire_time="3600">
  <ip ref="10.16.143.150"/>
  <lvm ref="ha-web-HA-LVM"/>
  <fs ref="ha-web-filesystem"/>
  <script ref="ha-web-apache-script"/>
</service>
</rm>
</cluster>
```



Appendix E: HA-Web Status Script

The HA-Web-Status script is used to verify that the web service and required resources are functioning correctly. The script is also useful for testing the web service after the service has been relocated or a node failover has occurred. Configure the script on the HA-LVM volume as follows:

1. Login to the node that currently has the HA-LVM volume mounted as `/ha/ha-web`
2. Create the `.cgi-bin` directory to hold the script:

```
# mkdir -p /ha/ha-web/.cgi-bin
```

3. Create the file `/ha/ha-web/.cgi-bin/HA-Web-Status` with the following contents:

```
#!/bin/bash
printf "Content-type: text/plain\n"
/bin/cat /etc/motd
printf "*** The current time is ` /bin/date ` ***\n\n"
```

4. Set the file permissions on the `.cgi-bin` directory and file so the script can be run:

```
# chmod 755 /ha/ha-web/.cgi-bin
# chmod 755 /ha/ha-web/.cgi-bin/HA-Web-Status
```

5. On each cluster node, edit `/etc/httpd/conf/httpd.conf` and change **ScriptAlias** to the `/ha/ha-web/.cgi-bin` directory on the HA-LVM volume:

```
ScriptAlias /cgi-bin/ "/ha/ha-web/.cgi-bin/"
```

Also set the `.cgi-bin` **Directory** entry:

```
<Directory "/ha/ha-web/.cgi-bin/">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>
```




6. Test the script from the command line:

```
# /ha/ha-web/.cgi-bin/HA-Web-Status
Content-type: text/plain

RHN Satellite kickstart on 2011-06-21

    *** Welcome to ha-web1 ***

ha-web1 is the first node of the HA-Web Cluster Reference Architecture
Project

*** The current time is Tue Aug  9 16:45:16 EDT 2011 ***
```

7. Once the cluster has been created and the web service configured, the script can be run from the web service. Open a browser window onto the web service (<http://10.16.143.150/cgi-bin/HA-Web-Status>) and running the test script (**Figure 1: HA-Web-Status**):

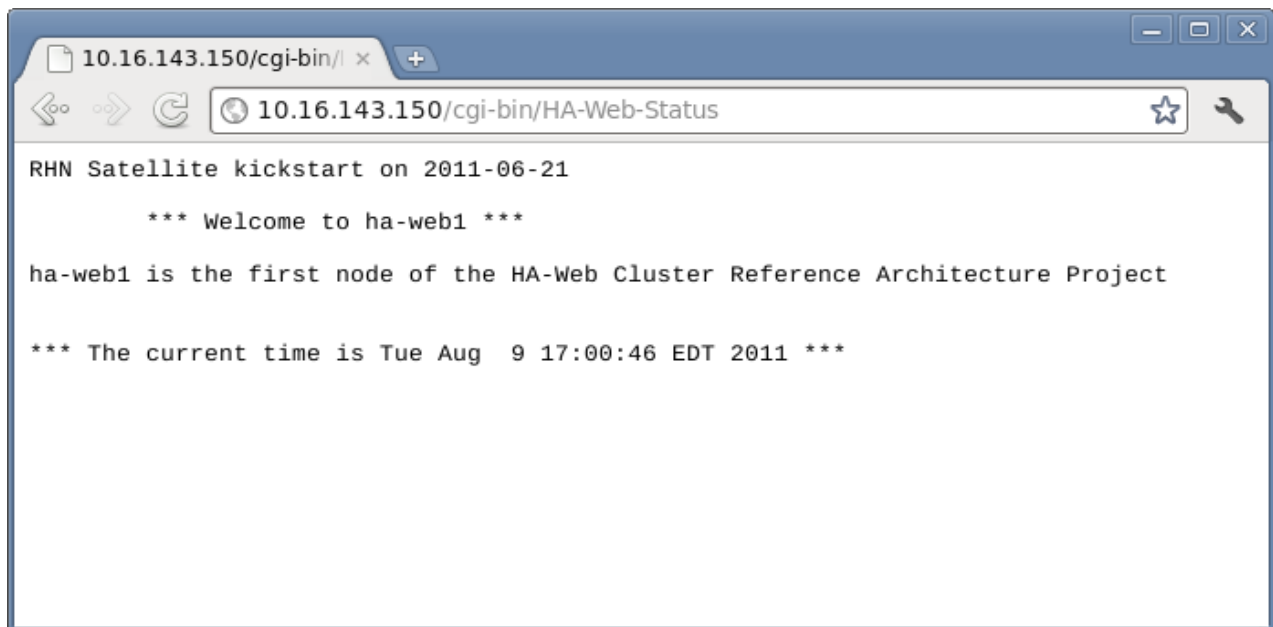


Figure 1: HA-Web-Status



Appendix F: Cluster Configuration Matrix

	ha-web1	ha-web2	ha-web3
Nodes			
Node Name IP Address (cluster interconnect)	ha-web1-ci 192.168.1.151	ha-web2-ci 192.168.1.152	ha-web3-ci 192.168.1.153
Ricci Hostname IP Address (public interface)	ha-web1 10.16.143.151	ha-web2 10.16.143.152	ha-web3 10.16.143.153
Fencing			
Fence Type	IPMI Lan	IPMI Lan	IPMI Lan
Fence Device Name	IPMI-ha-web1-ci	IPMI-ha-web2-ci	IPMI-ha-web3-ci
Fence Device IP Address	10.16.143.231	10.16.143.232	10.16.143.233
Fence Method Name	Primary	Primary	Primary
Fence Instance	IPMI-ha-web1-ci	IPMI-ha-web2-ci	IPMI-ha-web3-ci
Failover			
Failover Domain	ha-web-failover	ha-web-failover	ha-web-failover
Failover Type	Prioritized	Prioritized	Prioritized
Failover Priority	1	2	3
Resources			
IP Address	10.16.143.150	10.16.143.150	10.16.143.150
HA-LVM Volume Group Logical Volume	ha-web-HA-LVM HA-Web-VG ha-web-lvol1	ha-web-HA-LVM HA-Web-VG ha-web-lvol1	ha-web-HA-LVM HA-Web-VG ha-web-lvol1
Filesystem Type Mount point Device	ha-web-filesystem ext4 /ha/ha-web /dev/HA-Web-VG/ha-web-lvol1	ha-web-filesystem ext4 /ha/ha-web /dev/HA-Web-VG/ha-web-lvol1	ha-web-filesystem ext4 /ha/ha-web /dev/HA-Web-VG/ha-web-lvol1
Script Path	ha-web-apache-script /etc/init.d/httpd	ha-web-apache-script /etc/init.d/httpd	ha-web-apache-script /etc/init.d/httpd
Service Group			
Service	ha-web-service	ha-web-service	ha-web-service

Appendix F: Cluster Configuration Matrix



Appendix G: Deployment Checklists

Task	Task Description	Location	Details
<i>Deployment Tasks – Management Server</i>			
1	Install Red Hat Enterprise Linux 6	<i>ha-web-mgmt</i>	Section 4.2.1
2	Configure Networks	<i>ha-web-mgmt</i>	Section 4.2.2
3	Configure Firewall	<i>ha-web-mgmt</i>	Section 4.2.3
4	Install Cluster Management Software	<i>ha-web-mgmt</i>	Section 4.2.4
<i>Deployment Tasks – Cluster Nodes</i>			
5	Install Red Hat Enterprise Linux 6	<i>ha-web1</i> <i>ha-web2</i> <i>ha-web3</i>	Section 4.3.1
6	Configure Networks and Bonding	<i>ha-web1</i> <i>ha-web2</i> <i>ha-web3</i>	Section 4.3.2
7	Configure Firewall	<i>ha-web1</i> <i>ha-web2</i> <i>ha-web3</i>	Section 4.3.3
8	Install Cluster Node Software	<i>ha-web1</i> <i>ha-web2</i> <i>ha-web3</i>	Section 4.3.4
9	Configure Storage	<i>ha-web1</i> <i>ha-web2</i> <i>ha-web3</i>	Section 4.3.5
10	Configure Web Server	<i>ha-web1</i> <i>ha-web2</i> <i>ha-web3</i>	Section 4.3.6
11	Configure SELinux Security Parameters	<i>ha-web1</i> <i>ha-web2</i> <i>ha-web3</i>	Section 4.3.7

Appendix G: Deployment Checklist



Task	Task Description	Location	Details
Cluster Creation Tasks – Conga (Method 1)			
12	Create Cluster	<i>ha-web-mgmt</i>	Section 4.4.1
13	Add Fence Devices	<i>ha-web-mgmt</i>	Section 4.4.2
14	Add Failover Domain	<i>ha-web-mgmt</i>	Section 4.4.3
15	Add Resources	<i>ha-web-mgmt</i>	Section 4.4.4
16	Add Service Group	<i>ha-web-mgmt</i>	Section 4.4.5
17	Verify Cluster Web Service	<i>ha-web-mgmt</i>	Section 4.4.6
Cluster Creation Tasks – ccs (Method 2)			
12	Create Cluster	<i>ha-web-mgmt</i>	Section 4.5.1
13	Add Nodes	<i>ha-web-mgmt</i>	Section 4.5.2
14	Add Fence Devices	<i>ha-web-mgmt</i>	Section 4.5.3
15	Add Failover Domain	<i>ha-web-mgmt</i>	Section 4.5.4
16	Add Resources	<i>ha-web-mgmt</i>	Section 4.5.5
17	Add Service Group	<i>ha-web-mgmt</i>	Section 4.5.6
18	Activate Cluster	<i>ha-web-mgmt</i>	Section 4.5.7
19	Verify Cluster Web Service	<i>ha-web-mgmt</i>	Section 4.5.8

Appendix G: Deployment Checklist (continued)