SUSE Linux Enterprise High Availability Extension

11 www.novell.com

April 17, 2009 | High Availability Guide

High Availability Guide

Contents

About This Guide

SUSE® Linux Enterprise High Availability Extension is an integrated suite of open source clustering technologies that enables you to implement highly available physical and virtual Linux clusters. For quick and efficient configuration and administration, the High Availability Extension includes both a graphical user interface (GUI) and a command line interface (CLI).

This guide is intended for administrators who need to set up, configure, and maintain High Availability (HA) clusters. Both approaches (GUI and CLI) are covered in detail to help the administrators choose the appropriate tool that matches their needs for performing the key tasks.

The guide is subdivided into the following parts:

Installation and Setup
: Before starting to install and configure your cluster, make yourself familiar with the cluster fundamentals and architecture, get an overview of the key features and benefits, and what changed since the last release. Learn which hardware and software requirements must be met and which preparations to take before executing the next steps. Perform the installation and basic setup of your HA cluster using YaST.

Configuration and Administration
: Add, configure and manage resources, using either the GUI or the `crm` command line interface. Learn how to make use of load balancing and fencing. In case you consider writing your own resource agents or modifying existing ones, get some background information on how to create different types of resource agents.

Storage and Data Replication
: SUSE Linux Enterprise High Availability Extension ships with a cluster-aware file system (Oracle Cluster File System, OCFS2) and volume manager (clustered Logical Volume Manager, cLVM). For replication of your data, the High Availability Extension also delivers DRBD (Distributed Replicated Block Device) which you can use to mirror the data of a high availably service from the active node of a cluster to its standby node.

Troubleshooting and Reference
: Managing your own cluster requires you to perform a certain amount of troubleshooting. Learn about the most common problems and how to fix them. Find a compre-

hensive reference of the command line tools the High Availability Extension offers for administering your own cluster. Also lists the most important facts and figures about cluster resources and resource agents.

Many chapters in this manual contain links to additional documentation resources. This includes additional documentation that is available on the system as well as documentation available on the Internet.

For an overview of the documentation available for your product and the latest documentation updates, refer to `http://www.novell.com/documentation`.

1 Feedback

Several feedback channels are available:

- To report bugs for a product component or to submit enhancement requests, please use `https://bugzilla.novell.com/`. If you are new to Bugzilla, you might find the *Bug Writing FAQs* helpful, available from the Novell Bugzilla home page.
- We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comments feature at the bottom of each page of the online documentation and enter your comments there.

2 Documentation Conventions

The following typographical conventions are used in this manual:

- `/etc/passwd`: directory names and filenames
- *`placeholder`*: replace *`placeholder`* with the actual value
- `PATH`: the environment variable PATH
- `ls`, `--help`: commands, options, and parameters

- `user`: users or groups
- Alt, Alt + F1: a key to press or a key combination; keys are shown in uppercase as on a keyboard
- *File*, *File* > *Save As*: menu items, buttons
- This paragraph is only relevant for the specified architectures. The arrows mark the beginning and the end of the text block.

  This paragraph is only relevant for the specified architectures. The arrows mark the beginning and the end of the text block.
- *Dancing Penguins* (Chapter *Penguins*, ↑Another Manual): This is a reference to a chapter in another manual.

Part I. Installation and Setup

1 Conceptual Overview

SUSE® Linux Enterprise High Availability Extension is an integrated suite of open source clustering technologies that enables you to implement highly available physical and virtual Linux clusters, and to eliminate single points of failure. It ensures high availability and manageability of critical network resources including data, applications, and services. Thus, it helps you maintain business continuity, protect data integrity, and reduce unplanned downtime for your mission-critical Linux workloads.

It ships with essential monitoring, messaging, and cluster resource management functionality, supporting failover, failback, and migration (load balancing) of individually managed cluster resources. The High Availability Extension is available as add-on to SUSE Linux Enterprise Server 11.

1.1 Product Features

SUSE® Linux Enterprise High Availability Extension helps you ensure and manage the availability of your network resources. The following list highlights some of the key features:

Support for a Wide Range of Clustering Scenarios
: Including active/active and active/passive (N+1, N+M, N to 1, N to M) scenarios, as well as hybrid physical and virtual clusters (allowing virtual servers to be clustered with physical servers to improve services availability and resource utilization).

Multi-node active cluster, containing up to 16 Linux servers. Any server in the cluster can restart resources (applications, services, IP addresses, and file systems) from a failed server in the cluster.

Flexible Solution
: The High Availability Extension ships with OpenAIS messaging and membership layer and Pacemaker Cluster Resource Manager. Using Pacemaker, administrators can continuously monitor the health of their resources, manage dependencies, and automatically stop and start services based on highly configurable rules and policies. The High Availability Extension allows you to tailor a cluster to the specific applications and hardware infrastructure that fit your organization. Time-dependent configuration enables services to automatically migrate back to repaired nodes at specified times.

Storage and Data Replication
: With the High Availability Extension you can dynamically assign and reassign server storage as needed. It supports Fibre Channel or iSCSI storage area networks (SANs). Shared disk systems are also supported, but they are not a requirement. SUSE Linux Enterprise High Availability Extension also comes with a cluster-aware file system (Oracle Cluster File System, OCFS2) and volume manager (clustered Logical Volume Manager, cLVM). For replication of your data, the High Availability Extension also delivers DRBD (Distributed Replicated Block Device) which you can use to mirror the data of a high availably service from the active node of a cluster to its standby node.

Support for Virtualized Environments
: SUSE Linux Enterprise High Availability Extension supports the mixed clustering of both physical and virtual Linux servers. SUSE Linux Enterprise Server 11 ships with Xen, an open source virtualization hypervisor. The cluster resource manager in the High Availability Extension is able to recognize, monitor and manage services running within virtual servers created with Xen, as well as services running in physical servers. Guest systems can be managed as services by the cluster.

Resource Agents
: SUSE Linux Enterprise High Availability Extension includes a huge number of resource agents to manage a resources such as Apache, IPv4, IPv6 and many more. It also ships with resource agents for popular third party applications such as IBM WebSphere Application Server. For a list of Open Cluster Framework (OCF) resource agents included with your product, refer to Chapter 15, *HA OCF Agents*

(page 189). The most up to date list is available online at www.novell.com/products/highavailability.

User-friendly Administration
: For easy configuration and administration, the High Availability Extension ships with both a graphical user interface (like YaST and the Linux HA Management Client) and a powerful unified command line interface. Both approaches provide a single point of administration for effectively monitoring and administrating your cluster. Learn how to do so in the following chapters.

1.2 Product Benefits

The High Availability Extension allows you to configure up to 16 Linux servers into a high-availability cluster (HA cluster), where resources can be dynamically switched or moved to any server in the cluster. Resources can be configured to automatically migrate in the event of a server failure, or they can be moved manually to troubleshoot hardware or balance the workload.

The High Availability Extension provides high availability from commodity components. Lower costs are obtained through the consolidation of applications and operations onto a cluster. The High Availability Extension also allows you to centrally manage the complete cluster and to adjust resources to meet changing workload requirements (thus, manually “load balance” the cluster). Allowing clusters of more than two nodes also provides savings by allowing several nodes to share a “hot spare”.

An equally important benefit is the potential reduction of unplanned service outages as well as planned outages for software and hardware maintenance and upgrades.

Reasons that you would want to implement a cluster include:

- Increased availability
- Improved performance
- Low cost of operation
- Scalability
- Disaster recovery

- Data protection
- Server consolidation
- Storage consolidation

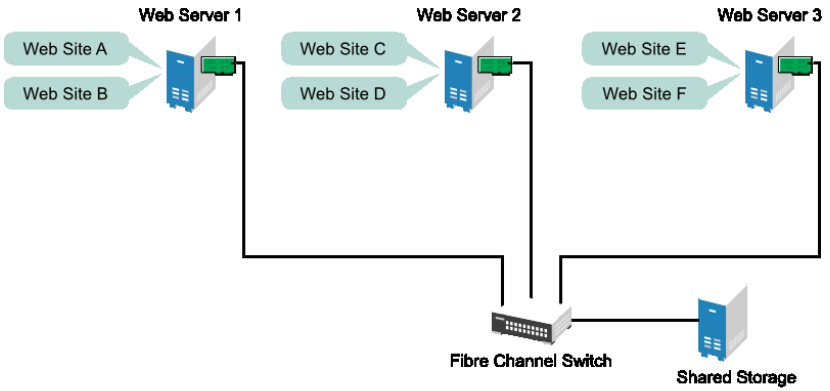
Shared disk fault tolerance can be obtained by implementing RAID on the shared disk subsystem.

The following scenario illustrates some of the benefits the High Availability Extension can provide.

Example Cluster Scenario

Suppose you have configured a three-server cluster, with a Web server installed on each of the three servers in the cluster. Each of the servers in the cluster hosts two Web sites. All the data, graphics, and Web page content for each Web site are stored on a shared disk subsystem connected to each of the servers in the cluster. The following figure depicts how this setup might look.

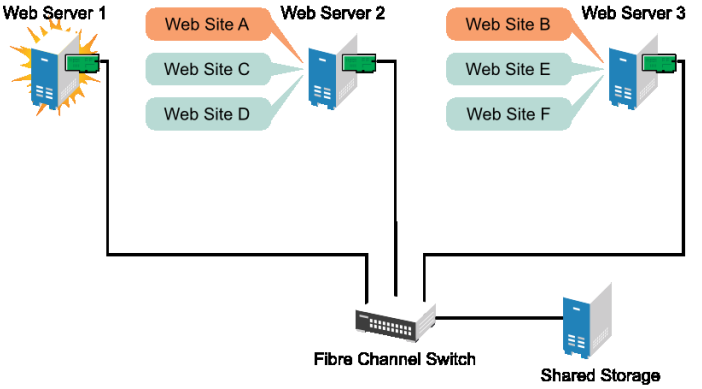
Figure 1.1 *Three-Server Cluster*



During normal cluster operation, each server is in constant communication with the other servers in the cluster and performs periodic polling of all registered resources to detect failure.

Suppose Web Server 1 experiences hardware or software problems and the users depending on Web Server 1 for Internet access, e-mail, and information lose their connections. The following figure shows how resources are moved when Web Server 1 fails.

Figure 1.2 *Three-Server Cluster after One Server Fails*



Web Site A moves to Web Server 2 and Web Site B moves to Web Server 3. IP addresses and certificates also move to Web Server 2 and Web Server 3.

When you configured the cluster, you decided where the Web sites hosted on each Web server would go should a failure occur. In the previous example, you configured Web Site A to move to Web Server 2 and Web Site B to move to Web Server 3. This way, the workload once handled by Web Server 1 continues to be available and is evenly distributed between any surviving cluster members.

When Web Server 1 failed, the High Availability Extension software

- Detected a failure and verified with STONITH that Web Server 1 was really dead
- Remounted the shared data directories that were formerly mounted on Web server 1 on Web Server 2 and Web Server 3.
- Restarted applications that were running on Web Server 1 on Web Server 2 and Web Server 3
- Transferred IP addresses to Web Server 2 and Web Server 3

In this example, the failover process happened quickly and users regained access to Web site information within seconds, and in most cases, without needing to log in again.

Now suppose the problems with Web Server 1 are resolved, and Web Server 1 is returned to a normal operating state. Web Site A and Web Site B can either automatically fail back (move back) to Web Server 1, or they can stay where they are. This is dependent on how you configured the resources for them. Migrating the services back to Web Server 1 will incur some down-time, so the High Availability Extension also allows you to defer the migration until a period when it will cause little or no service interruption. There are advantages and disadvantages to both alternatives.

The High Availability Extension also provides resource migration capabilities. You can move applications, Web sites, etc. to other servers in your cluster as required for system management.

For example, you could have manually moved Web Site A or Web Site B from Web Server 1 to either of the other servers in the cluster. You might want to do this to upgrade or perform scheduled maintenance on Web Server 1, or just to increase performance or accessibility of the Web sites.

1.3 Cluster Configurations

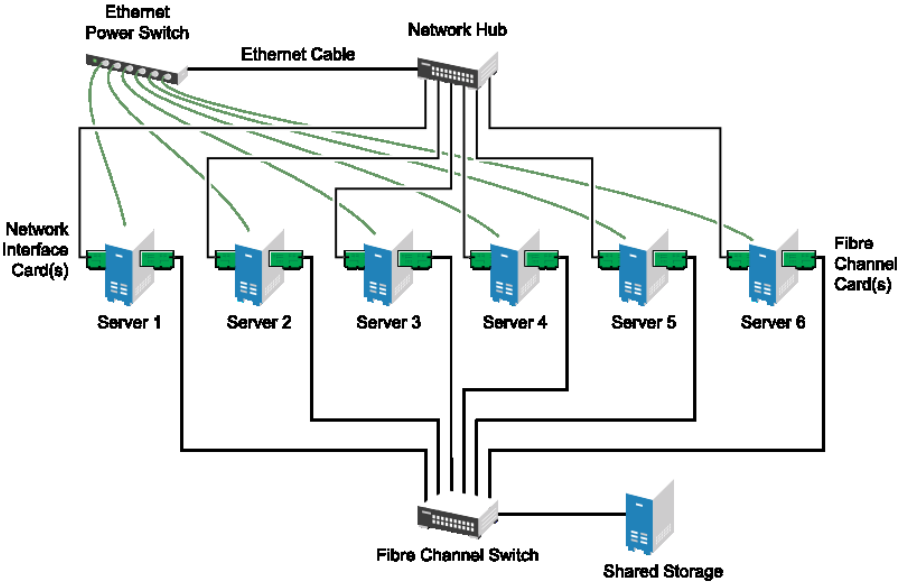
Cluster configurations with the High Availability Extension might or might not include a shared disk subsystem. The shared disk subsystem can be connected via high-speed Fibre Channel cards, cables, and switches, or it can be configured to use iSCSI. If a server fails, another designated server in the cluster automatically mounts the shared disk directories previously mounted on the failed server. This gives network users continuous access to the directories on the shared disk subsystem.

IMPORTANT: Shared Disk Subsystem with cLVM

When using a shared disk subsystem with cLVM, that subsystem must be connected to all servers in the cluster from which it needs to be accessed.

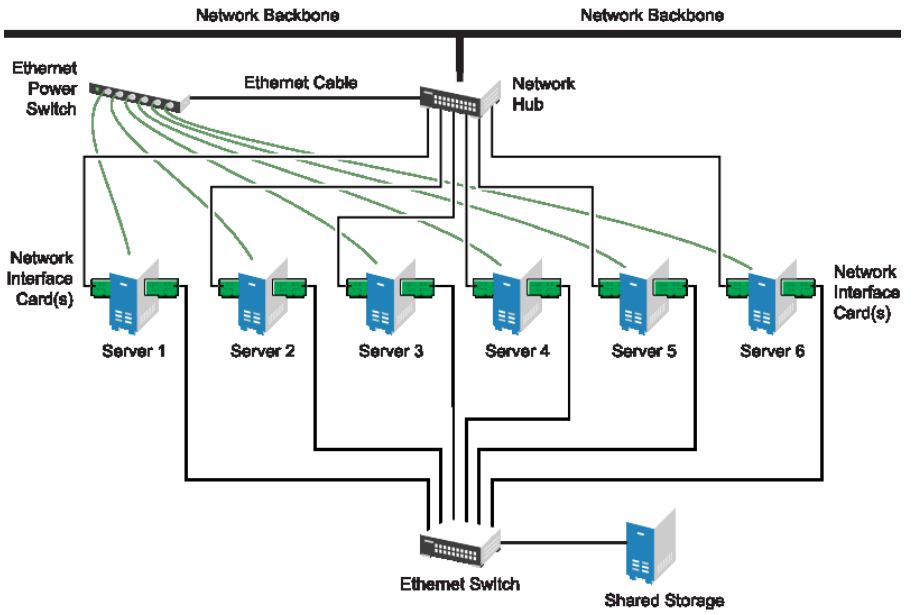
Typical resources might include data, applications, and services. The following figure shows how a typical Fibre Channel cluster configuration might look.

Figure 1.3 *Typical Fibre Channel Cluster Configuration*



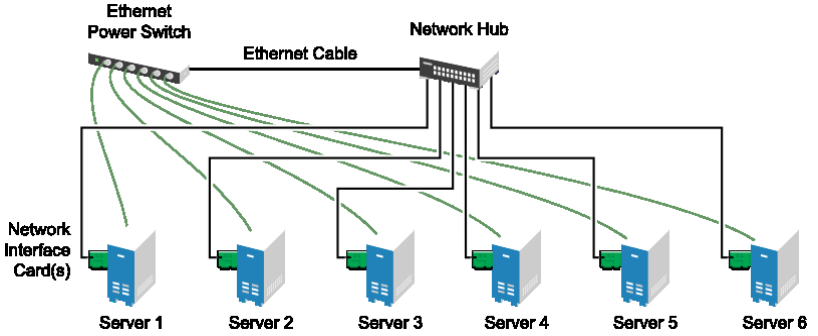
Although Fibre Channel provides the best performance, you can also configure your cluster to use iSCSI. iSCSI is an alternative to Fibre Channel that can be used to create a low-cost Storage Area Network (SAN). The following figure shows how a typical iSCSI cluster configuration might look.

Figure 1.4 *Typical iSCSI Cluster Configuration*



Although most clusters include a shared disk subsystem, it is also possible to create a cluster without a share disk subsystem. The following figure shows how a cluster without a shared disk subsystem might look.

Figure 1.5 *Typical Cluster Configuration Without Shared Storage*



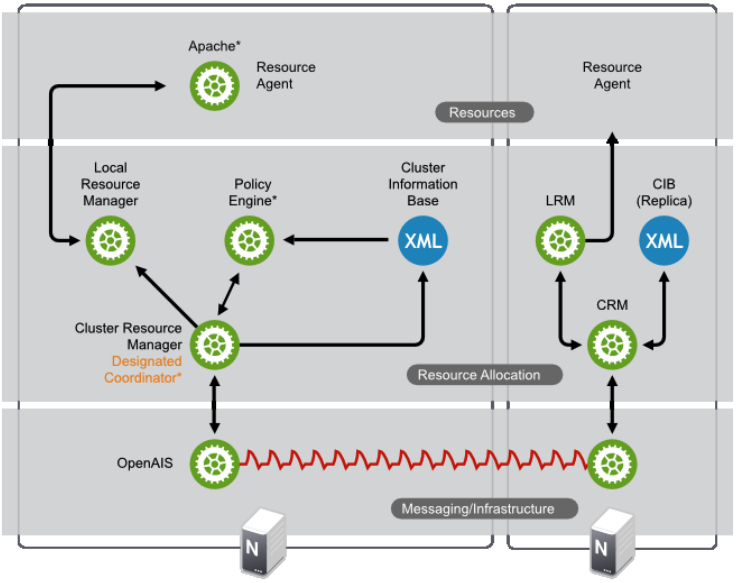
1.4 Architecture

This section provides a brief overview of the High Availability Extension architecture. It identifies and provides information on the architectural components, and describes how those components interoperate.

1.4.1 Architecture Layers

The High Availability Extension has a layered architecture. Figure 1.6, “Architecture” (page 11) illustrates the different layers and their associated components.

Figure 1.6 *Architecture*



Messaging and Infrastructure Layer

The primary or first layer is the messaging/infrastructure layer, also known as the OpenAIS layer. This layer contains components that send out the messages containing "I'm alive" signals, as well as other information. The program of the High Availability Extension resides in the messaging/infrastructure layer.

Resource Allocation Layer

The next layer is the resource allocation layer. This layer is the most complex, and consists of the following components:

Cluster Resource Manager (CRM)

Every action taken in the resource allocation layer passes through the Cluster Resource Manager. If other components of the resource allocation layer (or components which are in a higher layer) need to communicate, they do so through the local CRM.

On every node, the CRM maintains the Cluster Information Base (CIB) (page 12), containing definitions of all cluster options, nodes, resources their relationship and current status. One CRM in the cluster is elected as the Designated Coordinator (DC), meaning that it has the master CIB. All other CIBs in the cluster are a replicas of the master CIB. Normal read and write operations on the CIB are serialized through the master CIB. The DC is the only entity in the cluster that can decide that a cluster-wide change needs to be performed, such as fencing a node or moving resources around.

Cluster Information Base (CIB)

The Cluster Information Base is an in-memory XML representation of the entire cluster configuration and current status. It contains definitions of all cluster options, nodes, resources, constraints and the relationship to each other. The CIB also synchronizes updates to all cluster nodes. There is one master CIB in the cluster, maintained by the DC. All the other nodes contain a CIB replica.

Policy Engine (PE)

Whenever the Designated Coordinator needs to make a cluster-wide change (react to a new CIB), the Policy Engine calculates the next state of the cluster based on the current state and the configuration. The PE also produces a transition graph

containing a list of (resource) actions and dependencies to achieve the next cluster state. The PE runs on every node to speed up DC failover.

Local Resource Manager (LRM)
: The LRM calls the local Resource Agents (see Section “Resource Layer” (page 13)) on behalf of the CRM. It can thus perform start / stop / monitor operations and report the result to the CRM. It also hides the difference between the supported script standards for Resource Agents (OCF, LSB, Heartbeat Version 1). The LRM is the authoritative source for all resource related information on its local node.

Resource Layer

The highest layer is the Resource Layer. The Resource Layer includes one or more Resource Agents (RA). Resource Agents are programs, usually shell scripts, that have been written to start, stop, and monitor a certain kind of service (a resource). Resource Agents are called only by the LRM. Third parties can include their own agents in a defined location in the file system and thus provide out-of-the-box cluster integration for their own software.

1.4.2 Process Flow

SUSE Linux Enterprise High Availability Extension uses Pacemaker as CRM. The CRM is implemented as daemon (`crmd`) that has an instance on each cluster node. Pacemaker centralizes all cluster decision making by electing one of the crmd instances to act as a master. Should the elected crmd process (or the node it is on) fail, a new one is established.

A CIB, reflecting the cluster’s configuration and current state of all resources in the cluster is kept on each node. The contents of the CIB are automatically kept in sync across the entire cluster.

Many actions performed in the cluster will cause a cluster-wide change. These actions can include things like adding or removing a cluster resource or changing resource constraints. It is important to understand what happens in the cluster when you perform such an action.

For example, suppose you want to add a cluster IP address resource. To do this, you can use one of the command line tools or the GUI to modify the CIB. It is not required to perform the actions on the DC, you can use either tool on any node in the cluster and

they will be relayed to the DC. The DC will then replicate the CIB change to all cluster nodes.

Based on the information in the CIB, the PE then computes the ideal state of the cluster and how it should be achieved and feeds a list of instructions to the DC. The DC sends commands out via the messaging/infrastructure layer which are received by the crmd peers on other nodes. Each crmd uses it LRM (implemented as lrmd) to perform resource modifications. The lrmd is non-cluster aware and interacts directly with resource agents (scripts).

The peer nodes all report the results of their operations back to the DC. Once the DC concludes that all necessary operations are successfully performed in the cluster, the cluster will go back to the idle state and wait for further events. If any operation was not carried out as planned, the PE is invoked again with the new information recorded in the CIB.

In some cases, it may be necessary to power off nodes in order to protect shared data or complete resource recovery. For this Pacemaker comes with a fencing subsystem, stonithd. STONITH is an acronym for "Shoot The Other Node In The Head" and is usually implemented with a remote power switch. In Pacemaker, STONITH devices are modeled as resources (and configured in the CIB) to enable them to be easily monitored for failure, however stonithd takes care of understanding the STONITH topology such that its clients simply request a node be fenced and it does the rest.

1.5 What's New?

With SUSE Linux Enterprise Server 11, the cluster stack has changed from Heartbeat to OpenAIS. OpenAIS implements an industry standard API, the Application Interface Specification (AIS), published by the Service Availability Forum. The cluster resource manager from SUSE Linux Enterprise Server 10 has been retained but has been significantly enhanced, ported to OpenAIS and is now known as Pacemaker.

For more details what changed in the High Availability components from SUSE® Linux Enterprise Server 10 SP2 to SUSE Linux Enterprise High Availability Extension 11, refer to the following sections.

1.5.1 New Features and Functions Added

Migration Threshold and Failure Timeouts
: The High Availability Extension now comes with the concept of a migration threshold and a failure timeout. You can define a number of failures for resources, after which they will migrate to a new node. By default, the node will no longer be allowed to run the failed resource until the administrator manually resets the resource's failcount. However it is also possible to expire them by setting the resource's `failure-timeout` option.

Resource and Operation Defaults
: You can now set global defaults for resource options and operations.

Support for Offline Configuration Changes
: Often it is desirable to preview the effects of a series of changes before updating the configuration atomically. You can now create a "shadow" copy of the configuration that can be edited with the command line interface, before committing it and thus changing the active cluster configuration atomically.

Reusing Rules, Options and Sets of Operations
: Rules, instance_attributes, meta_attributes and sets of operations can be defined once and referenced in multiple places.

Using XPath Expressions for Certain Operations in the CIB
: The CIB now accepts XPath-based `create`, `modify`, `delete` operations. For more information, refer to the `cibadmin` help text.

Multi-dimensional Collocation and Ordering Constraints
: For creating a set of collocated resources, previously you could either define a resource group (which could not always accurately express the design) or you could define each relationship as an individual constraint—causing a constraint explosion as the number of resources and combinations grew. Now you can also use an alternate form of collocation constraints by defining `resource_sets`.

Connection to the CIB From Non-cluster Machines
: Provided Pacemaker is installed on a machine, it is possible to connect to the cluster even if the machine itself is not a part of it.

Triggering Recurring Actions at Known Times

By default, recurring actions are scheduled relative to when the resource started, but this is not always desirable. To specify a date/time that the operation should be relative to, set the operation's interval-origin. The cluster uses this point to calculate the correct start-delay such that the operation will occur at origin + (interval * N).

1.5.2 Changed Features and Functions

Naming Conventions for Resource and Custer Options

All resource and cluster options now use dashes (-) instead of underscores (_). For example, the `master_max` meta option has been renamed to `master-max`.

Renaming of `master_slave` Resource

The `master_slave` resource has been renamed to `master`. Master resources are a special type of clone that can operate in one of two modes.

Container Tag for Attributes

The `attributes` container tag has been removed.

Operation Field for Prerequisites

The `pre-req` operation field has been renamed `requires`.

Interval for Operations

All operations must have an interval. For start/stop actions the interval must be set to `0` (zero).

Attributes for Collocation and Ordering Constraints

The attributes of collocation and ordering constraints were renamed for clarity.

Cluster Options for Migration Due to Failure

The `resource-failure-stickiness` cluster option has been replaced by the `migration-threshold` cluster option. See also Migration Threshold and Failure Timeouts (page 15).

Arguments for Command Line Tools

The arguments for command-line tools have been made consistent. See also Naming Conventions for Resource and Custer Options (page 16).

Validating and Parsing XML
: The cluster configuration is written in XML. Instead of a Document Type Definition (DTD), now a more powerful RELAX NG schema is used to define the pattern for the structure and content. `libxml2` is used as parser.

`id` Fields
: `id` fields are now XML IDs which have the following limitations:

- IDs cannot contain colons.
- IDs cannot begin with a number.
- IDs must be globally unique (not just unique for that tag).

References to Other Objects
: Some fields (such as those in constraints that refer to resources) are IDREFs. This means that they must reference existing resources or objects in order for the configuration to be valid. Removing an object which is referenced elsewhere will therefor fail.

1.5.3 Removed Features and Functions

Setting Resource Meta Options
: It is no longer possible to set resource meta options as top-level attributes. Use meta attributes instead.

Setting Global Defaults
: Resource and operation defaults are no longer read from `crm_config`.

2 Getting Started

In the following, learn about the system requirements and which preparations to take before installing the High Availability Extension. Find a short overview of the basic steps to install and set up a cluster.

2.1 Hardware Requirements

The following list specifies hardware requirements for a cluster based on SUSE® Linux Enterprise High Availability Extension. These requirements represent the minimum hardware configuration. Additional hardware might be necessary, depending on how you intend to use your cluster.

- 1 to 16 Linux servers with software as specified in Section 2.2, “Software Requirements” (page 20). The servers do not require identical hardware (memory, disk space, etc.).
- At least two TCP/IP communication media. Cluster nodes use multicast for communication so the network equipment has to support multicasting. The communication media should support a data rate of 100 Mbit/s or higher. Preferably, the Ethernet channels should be bonded. bonded.
- Optional: A shared disk subsystem connected to all servers in the cluster from where it needs to be accessed.
- A STONITH mechanism. STONITH is an acronym for “Shoot the other node in the head”. A STONITH device is a power switch which the cluster uses to reset

nodes that are thought to be dead or misbehaving. Resetting non-heartbeating nodes is the only reliable way to ensure that no data corruption is performed by nodes that hang and only appear to be dead.

For more information, refer to Chapter 8, *Fencing and STONITH* (page 81).

2.2 Software Requirements

Ensure that the following software requirements are met:

- SUSE® Linux Enterprise Server 11 with all available online updates installed on all nodes that will be part of the cluster.
- SUSE Linux Enterprise High Availability Extension 11 including all available online updates installed on all nodes that will be part of the cluster.

2.3 Shared Disk System Requirements

A shared disk system (Storage Area Network, or SAN) is recommended for your cluster if you want data to be highly available. If a shared disk subsystem is used, ensure the following:

- The shared disk system is properly set up and functional according to the manufacturer's instructions.
- The disks contained in the shared disk system should be configured to use mirroring or RAID to add fault tolerance to the shared disk system. Hardware-based RAID is recommended. Host-based software RAID is not supported for all configurations.
- If you are using iSCSI for shared disk system access, ensure that you have properly configured iSCSI initiators and targets.
- When using DRBD to implement a mirroring RAID system that distributes data across two machines, make sure to only access the replicated device. Use the same (bonded) NICs that the rest of the cluster uses to leverage the redundancy provided there.

2.4 Preparations

Prior to installation, execute the following preparatory steps:

- Configure hostname resolution and use static host information by by editing the `/etc/hosts` file on each server in the cluster. For more information, refer to *SUSE Linux Enterprise Server Administration Guide*, chapter *Basic Networking* , section *Configuring Hostname and DNS* , available at `http://www.novell.com/documentation`.

  It is essential that members of the cluster are able to find each other by name. If the names are not available, internal cluster communication will fail.

- Configure time synchronization by making cluster nodes synchronize to a time server outside the cluster . For more information, refer to *SUSE Linux Enterprise Server Administration Guide*, chapter *Time Synchronization with NTP* , available at `http://www.novell.com/documentation`.

  The cluster nodes will use the time server as their time synchronization source.

2.5 Overview: Installing and Setting Up a Cluster

After the preparations are done, the following steps are necessary to install and set up a cluster with SUSE® Linux Enterprise High Availability Extension:

1. Installing SUSE® Linux Enterprise Server 11 and SUSE® Linux Enterprise High Availability Extension 11 as add-on on top of SUSE Linux Enterprise Server. For detailed information, see Section 3.1, “Installing the High Availability Extension” (page 23).

2. Configuring OpenAIS. For detailed information, see Section 3.2, “Initial Cluster Setup” (page 24).

3. Starting OpenAIS and monitoring the cluster status. For detailed information, see Section 3.3, “Bringing the Cluster Online” (page 27).

4. Adding and configuring cluster resources, either with a graphical user interface (GUI) or from command line. For detailed information, see Chapter 4, *Configuring Cluster Resources with the GUI* (page 31) or Chapter 5, *Configuring Cluster Resources From Command Line* (page 59).

   To protect your data from possible corruption by means of fencing and STONITH, make sure to configure STONITH devices as resources. For detailed information, see Chapter 8, *Fencing and STONITH* (page 81).

   You might also need to create file systems on a shared disk (Storage Area Network, SAN) if they do not already exist and, if necessary, configure those file systems as cluster resources.

   Both cluster-aware (OCFS 2) and non-cluster-aware file systems can be configured with the High Availability Extension. If needed, you can also make use of data replication with DRBD. For detailed information, see Part III, "Storage and Data Replication" (page 91).

Installation and Basic Setup with YaST

3

There are several ways to install the software needed for High Availability clusters: either from a command line, using `zypper`, or with YaST which provides a graphical user interface. After installing the software on all nodes that will be part of your cluster, the next step is to initially configure the cluster so that the nodes can communicate with each others. This can either be done manually (by editing a configuration file) or with the YaST cluster module.

NOTE: Installing the Software Packages

The software packages needed for High Availability clusters are not automatically copied to the cluster nodes. Install SUSE® Linux Enterprise Server 11 and SUSE® Linux Enterprise High Availability Extension 11 on all nodes that will be part of your cluster.

3.1 Installing the High Availability Extension

The packages needed for configuring and managing a cluster with the High Availability Extension are included in the `High Availability` installation pattern. This patterns is only available after SUSE® Linux Enterprise High Availability Extension has been installed as add-on. For information on how to install add-on products, refer to the SUSE Linux Enterprise 11 *Deployment Guide*, chapter *Installing Add-On Products* .

1. Start YaST and select *Software* > *Software Management* to open the YaST package manager.
2. From the *Filter* list, select *Patterns* and activate the *High Availability* pattern in the pattern list.
3. Click *Accept* to start the installation of the packages.

3.2 Initial Cluster Setup

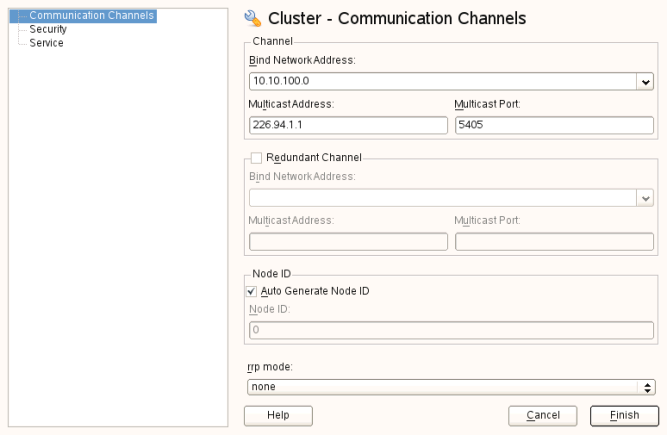
After having installed the HA packages, you can configure the initial cluster setup with YaST. This includes the communication channels between the nodes, security aspects like using encrypted communication and starting OpenAIS as service.

For the communication channels, you need to define a bind network address (`bindnetaddr`), a multicast address (`mcastaddr`) and a multicast port (`mcastport`). The `bindnetaddr` is the network address to bind to. To ease sharing configuration files across the cluster, OpenAIS uses network interface netmask to mask only the address bits that are used for routing the network. The `mcastaddr` can be a IPv4 or IPv6 multicast address. The `mcastport` is the UDP port specified for `mcastaddr`.

The nodes in the cluster will know each other from using the same multicast address and the same port number. For different clusters, use a different multicast address.

Procedure 3.1 *Configuring the Cluster*

1. Start YaST and select *Miscellaneous* > *Cluster* or run `yast2 cluster` on a command line to start the initial cluster configuration dialog.
2. In the *Communication Channel* category, configure the channels used for communication between the cluster nodes. This information is written to the `/etc/ais/openais.conf` configuration file.

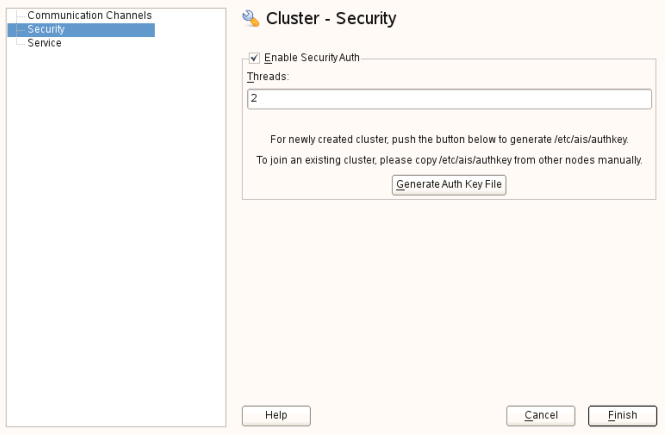


Define the *Bind Network Address*, the *Multicast Address* and the *Multicast Port* to use for all cluster nodes.

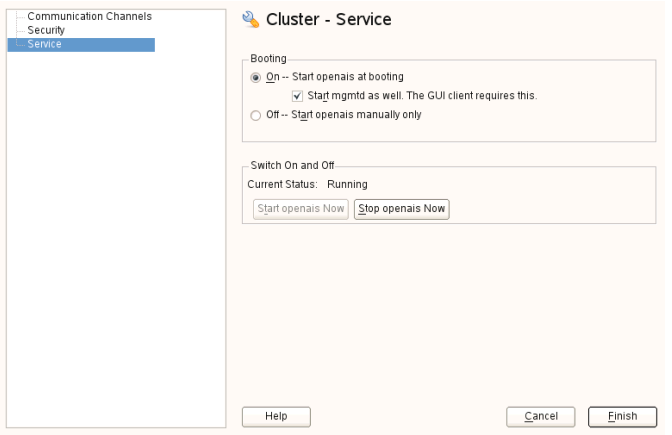
3 Specify a unique *Node ID* for every cluster node. It is recommended to start at `1`.

4 In the *Security* category, define the authentication settings for the cluster. If *Enable Security Authentication* is activated, HMAC/SHA1 authentication is used for communication between the cluster nodes.

This authentication method requires a shared secret, which is used to protect and authenticate messages. The authentication key (password) you specify will be used on all nodes in the cluster. For a newly created cluster, click *Generate Auth Key File* to create an authentication key that is written to `/etc/ais/authkey`.



5 In the *Service* category, choose whether you want to start OpenAIS on this cluster server each time it is booted.



If you select *Off*, you must start OpenAIS manually each time this cluster server is booted. To start OpenAIS manually, use the `rcopenais start` command.

To start OpenAIS immediately, click *Start OpenAIS Now*.

6 If all options are set according to your wishes, click *Finish*. YaST then automatically also adjusts the firewall settings and opens the UDP port used for multicast.

7 After the initial configuration is done, you need to transfer the configuration to the other nodes in the cluster. The easiest way to do so is to copy the `/etc/ais/openais.conf` file to the other nodes in the cluster. As each node needs to have a unique node ID, make sure to adjust the node ID accordingly after copying the file.

8 If you want to use encrypted communication, also copy the `/etc/ais/authkey` to the other nodes in the cluster.

3.3 Bringing the Cluster Online

After the basic configuration, you can bring the stack online and check the status.

1 Run the following command on each of the cluster nodes to start OpenAIS:

```
rcopenais start
```

2 On one of the nodes, check the cluster status with the following command:

```
crm_mon
```

If all nodes are online, the output should be similar to the following:

```
============
Last updated: Thu Feb  5 18:30:33 2009
Current DC: d42 (d42)
Version: 1.0.1-node: b7ffe2729e3003ac8ff740bebc003cf237dfa854
3 Nodes configured.
0 Resources configured.
============

Node: d230 (d230): online
Node: d42 (d42): online
Node: e246 (e246): online
```

After the basic configuration is done and the nodes are online, you can now start to configure cluster resources, either with the `crm` command line tool or with a graphical user interface. For more information, refer to Chapter 4, *Configuring Cluster Resources with the GUI* (page 31) or Chapter 5, *Configuring Cluster Resources From Command Line* (page 59).

Part II. Configuration and Administration

Configuring Cluster Resources with the GUI

4

The main purpose of an HA cluster is to manage user services. Typical examples of user services are an Apache web server, or a database. From the user's point of view, the services do something specific when ordered to do so. To the cluster, however, they are just resources which may be started or stopped—the nature of the service is irrelevant to the cluster.

As a cluster administrator, you need to create cluster resources for every resource or application you run on servers in your cluster. Cluster resources can include Web sites, e-mail servers, databases, file systems, virtual machines, and any other server-based applications or services you want to make available to users at all times.

To create cluster resources, either use the graphical user interface (the Linux HA Management Client) or the `crm` command line utility. For the command line approach, refer to Chapter 5, *Configuring Cluster Resources From Command Line* (page 59).

This chapter introduces the Linux HA Management Client and then covers several topics you need when configuring a cluster: creating resources, configuring constraints, specifying failover nodes and failback nodes, configuring resource monitoring, starting or removing resources, configuring resource groups or clone resources, and migrating resources manually.

The graphical user interface for configuring cluster resources is included in the `pacemaker-pygui` package.

4.1 Linux HA Management Client

When starting the Linux HA Management Client you need to connect to a cluster.

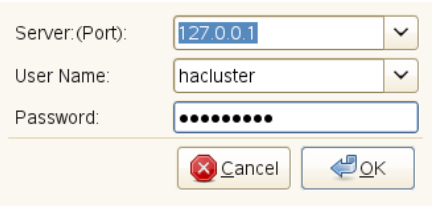
NOTE: Password for the `hacluster` User

The installation creates a linux user named `hacluster`. Prior to using the Linux HA Management Client, you must set the password for the `hacluster` user. To do this, become `root`, enter `passwd hacluster` at the command line and enter a password for the `hacluster` user.

Do this on every node you will connect to with the Linux HA Management Client.

To start the Linux HA Management Client, enter `crm_gui` at the command line. To connect to the cluster, select *Connection* > *Login*. By default, the *Server* field shows the localhost's IP address and `hacluster` as *User Name*. Enter the user's password to continue.

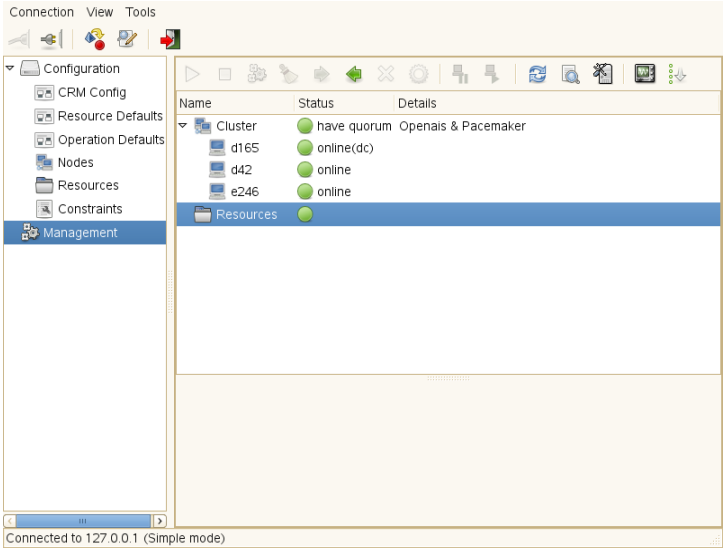
Figure 4.1 *Connecting to the Cluster*



If you are running the Linux HA Management Client remotely, enter the IP address of a cluster node as *Server*. As *User Name*, you can also use any other user belonging to the `haclient` group to connect ot the cluster.

After being connected, the main window opens:

Figure 4.2 *Linux HA Management Client - Main Window*



The Linux HA Management Client lets you add and modify resources, constraints, configurations etc. It also provides functionalities for managing cluster components like starting, stopping or migrating resources, cleaning up resources, or setting nodes to `standby`. Additionally, you can easily view, edit, import and export the XML structures of the CIB by selecting any of the *Configuration* subitems and selecting *Show > XML Mode*.

In the following, find some examples how to create and manage cluster resources with the Linux HA Management Client.

4.2 Creating Cluster Resources

You can create the following types of resources:

Primitive
: A primitive resource, the most basic type of a resource.

Group

Groups contain a set of resources that need to be located together, start sequentially and stop in the reverse order. For more information, refer to Section 4.10, “Configuring a Cluster Resource Group” (page 49).

Clone

Clones are resources that can be active on multiple hosts. Any resource can be cloned, provided the respective resource agent supports it. For more information, refer to Section 4.11, “Configuring a Clone Resource” (page 54).

Master

Masters are a special type of a clone resources, masters can have multiple modes. Masters must contain exactly one group or one regular resource.

Procedure 4.1 *Adding Primitive Resources*

1. Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, “Linux HA Management Client” (page 32).

2. In the left pane, select *Resources* and click *Add* > *Primitive*.

3. In the next dialog, set the following parameters for the resource:

   3a Enter a unique `ID` for the resource.

   3b From the *Class* list, select the resource agent class you want to use for that resource: *heartbeat*, *lsb*, *ocf* or *stonith*. For more information, see Section 14.1, “Supported Resource Agent Classes” (page 181).

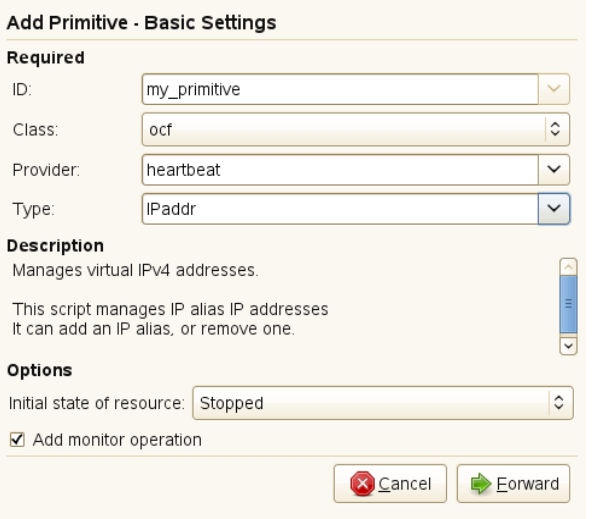
   3c If you selected *ocf* as class, specify also the *Provider* of your OCF resource agent. The OCF specification allows multiple vendors to supply the same resource agent.

   3d From the *Type* list, select the resource agent you want to use (for example, *IPaddr* or *Filesystem*). A short description for this resource agent is displayed below.

   The selection you get in the *Type* list depends on the *Class* (and for OCF resources also on the *Provider*) you have chosen.

   3e Below *Options*, set the *Initial state of resource*.

3f Activate *Add monitor operation* if you want the cluster to monitor if the resource is still healthy.



4 Click *Forward*. The next window shows a summary of the parameters that you have already defined for that resource. All required *Instance Attributes* for that resource are listed. You need to edit them in order to set them to appropriate values. You may also need to add more attributes, depending on your deployment and settings. For details how to do so, refer to Adding or Modifying Meta and Instance Attributes (page 36).

5 If all parameters are set according to your wishes, click *Apply* to finish the configuration of that resource. The configuration dialog is closed and the main window shows the newly added resource.

You can add or modify the following parameters for primitive resources at any time:

Meta Attributes

Meta attributes are options you can add for a resource. They tell the CRM how to treat a specific resource. For an overview of the available meta attributes, their values and defaults, refer to Section 14.3, “Resource Options” (page 185).

Instance Attributes

Instance attributes are parameters for certain resource classes that determine how they behave and which instance of a service they control. For more information, refer to Section 14.5, “Instance Attributes” (page 187).

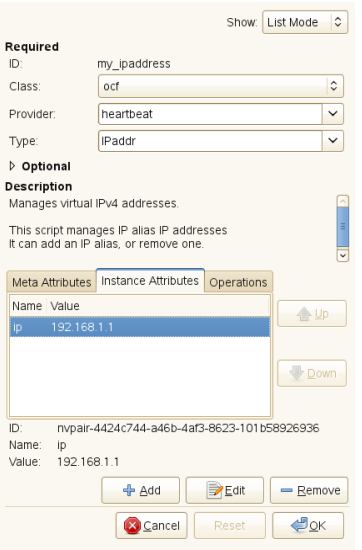
Operations

The monitor operations added for a resource. These instruct the cluster to make sure that the resource is still healthy. Monitor operations can be added for all classes of resource agents. You can also set particular parameters, such as `Timeout` for `start` or `stop` operations. For more information, refer to Section 4.7, “Configuring Resource Monitoring” (page 46).

Procedure 4.2 *Adding or Modifying Meta and Instance Attributes*

1. In the Linux HA Management Client main window, click *Resources* in the left pane to see the resources already configured for the cluster.

2. In the right pane, select the resource to modify and click *Edit* (or double-click the resource). The next window shows the basic resource parameters and the *Meta Attributes*, *Instance Attributes* or *Operations* already defined for that resource.



3 To add a new meta attribute or instance attribute, select the respective tab and click *Add*.

4 Select the *Name* of the attribute you want to add. A short *Description* is displayed.

5 If needed, specify an attribute *Value*. Otherwise the default value of that attribute will be used.

6 Click *OK* to confirm your changes. The newly added or modified attribute appears on the tab.

7 If all parameters are set according to your wishes, click *OK* to finish the configuration of that resource. The configuration dialog is closed and the main window shows the modified resource.

TIP: XML Source Code

The Linux HA Management Client allows you to view the XML that is generated from the parameters that you have defined for a specific resource or for all resources. Select *Show* > *XML Mode* in the top right corner of the resource configuration dialog or in the *Resources* view of the main window.

The editor displaying the XML code allows you to *Import* or *Export* the XML elements or to manually edit the XML code.

4.3 Creating STONITH Resources

To configure fencing, you need to configure one or more STONITH resources.

Procedure 4.3 *Adding a STONITH Resource*

1 Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, “Linux HA Management Client” (page 32).

2 In the left pane, select *Resources* and click *Add* > *Primitive*.

3 In the next dialog, set the following parameters for the resource:

3a Enter a unique `ID` for the resource.

3b From the *Class* list, select the resource agent class *stonith*.

3c From the *Type* list, select the STONITH plug-in for controlling your STONITH device. A short description for this plug-in is displayed below.

3d Below *Options*, set the *Initial state of resource*.

3e Activate *Add monitor operation* if you want the cluster to monitor the fencing device. For more information, refer to Section 8.4, “Monitoring Fencing Devices” (page 88).

4 Click *Forward*. The next window shows a summary of the parameters that you have already defined for that resource. All required *Instance Attributes* for the selected STONITH plug-in are listed. You need to edit them in order to set them to appropriate values. You may also need to add more attributes or monitor operations, depending on your deployment and settings. For details how to do so, refer to Adding or Modifying Meta and Instance Attributes (page 36) and Section 4.7, “Configuring Resource Monitoring” (page 46).

5 If all parameters are set according to your wishes, click *Apply* to finish the configuration of that resource. The configuration dialog is closed and the main window shows the newly added resource.

To complete your fencing configuration add constraints, or use clones or both. For more details, refer to Chapter 8, *Fencing and STONITH* (page 81).

4.4 Configuring Resource Constraints

Having all the resources configured is only part of the job. Even if the cluster knows all needed resources, it might still not be able to handle them correctly. Resource constraints let you specify which cluster nodes resources can run on, what order resources will load, and what other resources a specific resource is dependent on.

There are three different kinds of constraints available:

Resource Location
: Locational constraints that define on which nodes a resource may be run, may not be run or is preferred to be run.

Resource Collocation
: Collocational constraints that tell the cluster which resources may or may not run together on a node.

Resource Order
: Ordering constraints to define the sequence of actions.

When defining constraints, you also need to deal with scores. Scores of all kinds are integral to how the cluster works. Practically everything from migrating a resource to deciding which resource to stop in a degraded cluster is achieved by manipulating scores in some way. Scores are calculated on a per-resource basis and any node with a negative score for a resource cannot run that resource. After calculating the scores for a resource, the cluster then chooses the node with the highest one. `INFINITY` is currently defined as `1,000,000`. Additions or subtractions with it follows the following 3 basic rules:

- Any value + INFINITY = INFINITY
- Any value - INFINITY = -INFINITY
- INFINITY - INFINITY = -INFINITY

When defining resource constraints, you also specify a score for each constraint. The score indicates the value you are assigning to this resource constraint. Constraints with higher scores are applied before those with lower scores. By creating additional location constraints with different scores for a given resource, you can specify an order for the nodes that a resource will fail over to.

Procedure 4.4 *Adding or Modifying Locational Constraints*

1. Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, “Linux HA Management Client” (page 32).
2. In the Linux HA Management Client main window, click *Constraints* in the left pane to see the constraints already configured for the cluster.
3. In the left pane, select *Constraints* and click *Add*.
4. Select *Resource Location* and click *OK*.
5. Enter a unique *ID* for the constraint. When modifying existing constraints, the ID is already defined and is displayed in the configuration dialog.

6 Select the *Resource* for which to define the constraint. The list shows the IDs of all resources that have been configured for the cluster.

7 Set the *Score* for the constraint. Positive values indicate the resource can run on the *Node* you specify below. Negative values indicate the resource can not run on this node. Values of `+/- INFINITY` change "can" to `must`.

8 Select the *Node* for the constraint.

9 If you leave the *Node* and the *Score* field empty, you can also add rules by clicking *Add* > *Rule*. To add a lifetime, just click *Add* > *Lifetime*.

10 If all parameters are set according to your wishes, click *OK* to finish the configuration of the constraint. The configuration dialog is closed and the main window shows the newly added or modified constraint.

Procedure 4.5 *Adding or Modifying Collocational Constraints*

1 Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, "Linux HA Management Client" (page 32).

2 In the Linux HA Management Client main window, click *Constraints* in the left pane to see the constraints already configured for the cluster.

3 In the left pane, select *Constraints* and click *Add*.

4 Select *Resource Collocation* and click *OK*.

5 Enter a unique *ID* for the constraint. When modifying existing constraints, the ID is already defined and is displayed in the configuration dialog.

6 Select the *Resource* which is the collocation source. The list shows the IDs of all resources that have been configured for the cluster.

If the constraint cannot be satisfied, the cluster may decide not to allow the resource to run at all.

7 If you leave both the *Resource* and the *With Resource* field empty, you can also add a resource set by clicking *Add* > *Resource Set*. To add a lifetime, just click *Add* > *Lifetime*.

8 In *With Resource*, define the collocation target. The cluster will decide where to put this resource first and then decide where to put the resource in the *Resource* field.

9 Define a *Score* to determine the location relationship between both resources. Positive values indicate the resources should run on the same node. Negative values indicate the resources should not run on the same node. Values of + / – `INFINITY` change `should` to `must`. The score will be combined with other factors to decide where to put the node.

10 If needed, specify further parameters, like *Resource Role*.

Depending on the parameters and options you choose, a short *Description* explains the effect of the collocational constraint you are configuring.

11 If all parameters are set according to your wishes, click *OK* to finish the configuration of the constraint. The configuration dialog is closed and the main window shows the newly added or modified constraint.

Procedure 4.6 *Adding or Modifying Ordering Constraints*

1 Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, “Linux HA Management Client” (page 32).

2 In the Linux HA Management Client main window, click *Constraints* in the left pane to see the constraints already configured for the cluster.

3 In the left pane, select *Constraints* and click *Add*.

4 Select *Resource Order* and click *OK*.

5 Enter a unique *ID* for the constraint. When modifying existing constraints, the ID is already defined and is displayed in the configuration dialog.

6 With *First*, define the resource that must be started before the *Then* resource is allowed to.

7 With *Then* define the resource that will start after the *First* resource.

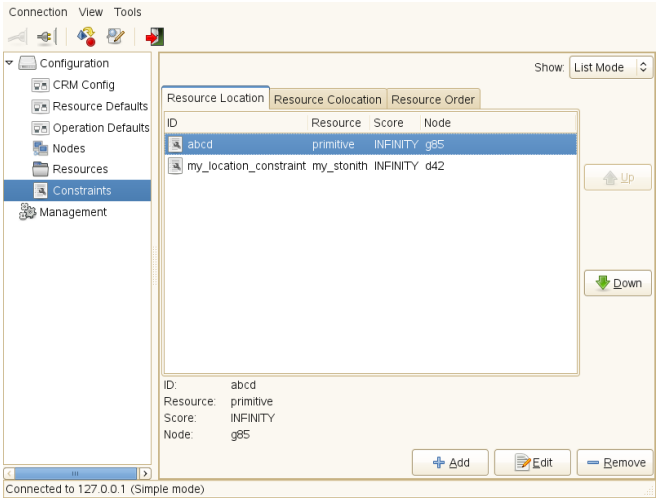
8 If needed, define further parameters, for example *Score* (if greater than zero, the constraint is mandatory; otherwise it is only a suggestion) or *Symmetrical* (if `true`, stop the resources in the reverse order).

   Depending on the parameters and options you choose, a short *Description* explains the effect of the ordering constraint you are configuring.

9 If all parameters are set according to your wishes, click *OK* to finish the configuration of the constraint. The configuration dialog is closed and the main window shows the newly added or modified constraint.

You can access and modify all constraints that you have configured in the *Constraints* view of the Linux HA Management Client.

Figure 4.3 *Linux HA Management Client - Constraints*



For more information on configuring constraints and detailed background information about the basic concepts of ordering and collocation, refer to the following documents available at http://clusterlabs.org/wiki/Documentation:

- *Configuration 1.0 Explained* , chapter *Resource Constraints*
- *Collocation Explained*
- *Ordering Explained*

4.5 Specifying Resource Failover Nodes

A resource will be automatically restarted if it fails. If that cannot be achieved on the current node, or it fails N times on the current node, it will try to fail over to another node. You can define a number of failures for resources (a `migration-threshold`), after which they will migrate to a new node. If you have more than two nodes in your cluster, the node a particular resource fails over to is chosen by the High Availability software.

If you want to choose which node a resource will fail over to, you must do the following:

1. Configure a location constraint for that resource as described in Adding or Modifying Locational Constraints (page 39).
2. Add the `migration-threshold` meta attribute to that resource as described in Adding or Modifying Meta and Instance Attributes (page 36) and enter a *Value* for the migration-threshold. The value should be positive and less that INFINITY.
3. If you want to automatically expire the failcount for a resource, add the `failure-timeout` meta attribute to that resource as described in Adding or Modifying Meta and Instance Attributes (page 36) and enter a *Value* for the failure-timeout.
4. If you want to specify additional failover nodes with preferences for a resource, create additional location constraints.

For example, let us assume you have configured a location constraint for resource `r1` to preferably run on `node1`. If it fails there, `migration-threshold` is checked and compared to the failcount. If failcount >= migration-threshold then the resource is migrated to the node with the next best preference.

By default, once the threshold has been reached, the node will no longer be allowed to run the failed resource until the administrator manually resets the resource's failcount (after fixing the failure cause).

However, it is possible to expire the failcounts by setting the resource's failure-timeout option. So a setting of `migration-threshold=2` and `failure-timeout=60s` would cause the resource to migrate to a new node after two failures and potentially allow it to move back (depending on the stickiness and constraint scores) after one minute.

There are two exceptions to the migration threshold concept, occurring when a resource either fails to start or fails to stop: Start failures set the failcount to INFINITY and thus always cause an immediate migration. Stop failures cause fencing (when `stonith-enabled` is set to `true` which is the default). In case there is no STONITH resource defined (or `stonith-enabled` is set to `false`), the resource will not migrate at all.

To clean up the failcount for a resource with the Linux HA Management Client, select *Management* in the left pane, select the respective resource in the right pane and click *Cleanup Resource* in the toolbar. This executes the commands `crm_resource -C` and `crm_failcount -D` for the specified resource on the specified node. For more information, see also crm_resource(8) (page 153) and crm_failcount(8) (page 143).

4.6 Specifying Resource Failback Nodes (Resource Stickiness)

A resource might fail back to its original node when that node is back online and in the cluster. If you want to prevent a resource from failing back to the node it was running on prior to a failover, or if you want to specify a different node for the resource to fail back to, you must change its resource stickiness value. You can specify resource stickiness when you are creating a resource, or after.

Consider the following when specifying a resource stickiness value:

Value is `0`:
: This is the default. The resource will be placed optimally in the system. This may mean that it is moved when a "better" or less loaded node becomes available. This option is almost equivalent to automatic failback, except that the resource may be moved to a node that is not the one it was previously active on.

Value is greater than `0`:
: The resource will prefer to remain in its current location, but may be moved if a more suitable node is available. Higher values indicate a stronger preference for a resource to stay where it is.

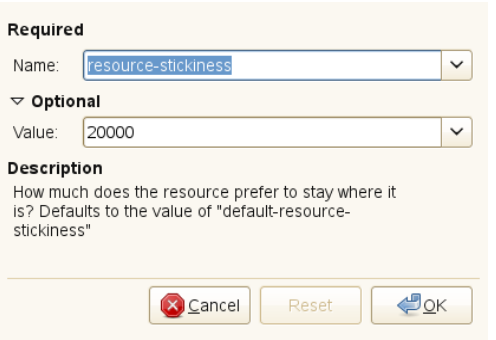
Value is less than `0`:
: The resource prefers to move away from its current location. Higher absolute values indicate a stronger preference for a resource to be moved.

Value is `INFINITY`:
: The resource will always remain in its current location unless forced off because the node is no longer eligible to run the resource (node shutdown, node standby, reaching the `migration-threshold`, or configuration change). This option is almost equivalent to completely disabling automatic failback .

Value is `-INFINITY`:
: The resource will always move away from its current location.

Procedure 4.7 *Specifying Resource Stickiness*

1. Add the `resource-stickiness` meta attribute to the resource as described in Adding or Modifying Meta and Instance Attributes (page 36).



2. As *Value* for the resource-stickiness, specify a value between `-INFINITY` and `INFINITY`.

4.7 Configuring Resource Monitoring

Although the High Availability Extension can detect a node failure, it also has the ability to detect when an individual resource on a node has failed. If you want to ensure that a resource is running, you must configure resource monitoring for that resource. Resource monitoring consists of specifying a timeout and/or start delay value, and an interval. The interval tells the CRM how often it should check the resource status.

Procedure 4.8 *Adding or Modifying Monitor Operations*

1. Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, “Linux HA Management Client” (page 32).

2. In the Linux HA Management Client main window, click *Resources* in the left pane to see the resources already configured for the cluster.

3. In the right pane, select the resource to modify and click *Edit*. The next window shows the basic resource parameters and the meta attributes, instance attributes and operations already defined for that resource.

4 To add a new monitor operation, select the respective tab and click *Add*.

   To modify an existing operation, select the respective entry and click *Edit*.

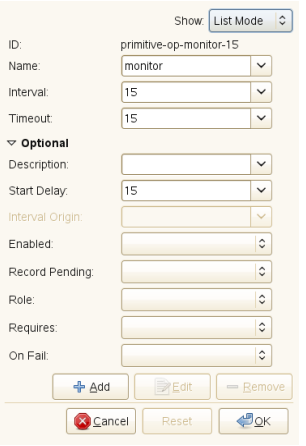
5 Enter a unique *ID* for the monitor operation. When modifying existing monitor operations, the ID is already defined and is displayed in the configuration dialog.

6 In *Name*, select the action to perform, for example `monitor`, `start`, or *stop*.

7 In the *Interval* field, enter a value in seconds.

8 In the *Timeout* field, enter a value in seconds. After the specified timeout period, the operation will be treated as `failed`. The PE will decide what to do or execute what you specified in the *On Fail* field of the monitor operation.

9 If needed, set optional parameters, like *On Fail* (what do if this action ever fails?) or *Requires* (what conditions need to be satisfied before this action occurs?).



10 If all parameters are set according to your wishes, click *OK* to finish the configuration of that resource. The configuration dialog is closed and the main window shows the modified resource.

If you do not configure resource monitoring, resource failures after a successful start will not be communicated, and the cluster will always show the resource as healthy.

If the resource monitor detects a failure, the following actions will take place:

- Log file messages will be generated, according to the configuration specified in the `logging` section of `/etc/ais/openais.conf` (by default, written to syslog, usually `/var/log/messages`).
- The failure will be reflected in the Linux HA Management Client, the `crm_mon` tools, and in the CIB status section. To view them in the Linux HA Management Client, click *Management* in the left pane, then in the right pane, select the resource whose details you want to see.
- The cluster will initiate noticeable recovery actions which may include stopping the resource to repair the failed state and restarting the resource locally or on another node. The resource also may not be restarted at all, depending on the configuration and state of the cluster.

4.8 Starting a New Cluster Resource

NOTE: Starting Resources

When configuring a resource with the High Availability Extension, the same resource should not be started or stopped manually (outside of the cluster). The High Availability Extension software is responsible for all service start or stop actions.

If a resource's initial state was set to `stopped` when being created (`target-role` meta attribute has the value `stopped`), it does not start automatically after being created. To start a new cluster resource with the Linux HA Management Client, select *Management* in the left pane. In the right pane, right-click the resource and select *Start* (or start it from the toolbar).

4.9 Removing a Cluster Resource

To remove a cluster resource with the Linux HA Management Client, switch to the *Resources* view in the left pane, then select the respective resource and click *Remove*.

NOTE: Removing Referenced Resources

Cluster resources cannot be removed if their ID is referenced by any constraint. If you cannot delete a resource, check where the resource ID is referenced and remove the resource from the constraint first.

4.10 Configuring a Cluster Resource Group

Some cluster resources are dependent on other components or resources, and require that each component or resource are started in a specific order and run together on the same server. To simplify this configuration we support the concept of groups.

Groups have the following properties:

Starting and Stopping Resources
: Resources are started in the order they appear in and stopped in the reverse order which they appear in.

Dependency
: If a resource in the group cannot run anywhere, then none of the resources located after that resource in the group is allowed to run.

Group Contents
: Groups may only contain a collection of primitive cluster resources. To refer to the child of a group resource, use the child's ID instead of the group's.

Constraints
: Although it is possible to reference the group's children in constraints, it is usually preferable to use the group's name instead.

Stickiness
: Stickiness is additive in groups. Every *active* member of the group will contribute its stickiness value to the group's total. So if the default `resource-stickiness` is `100` and a group has seven members, five of which are active, then the group as a whole will prefer its current location with a score of `500`.

Resource Monitoring
: To enable resource monitoring for a group, you must configure monitoring separately for each resource in the group that you want monitored.

NOTE: Empty Groups

Groups must contain at least one resource, otherwise the configuration is not valid.

Procedure 4.9 *Adding a Resource Group*

1. Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, "Linux HA Management Client" (page 32).

2. In the left pane, select *Resources* and click *Add* > *Group*.

3. Enter a unique `ID` for the group.

4. Below *Options*, set the *Initial state of resource* and click *Forward*.

5. In the next step, you can add primitives as sub-resources for the group. These are created similar as described in Adding Primitive Resources (page 34).

6. If all parameters are set according to your wishes, click *Apply* to finish the configuration of the primitive.

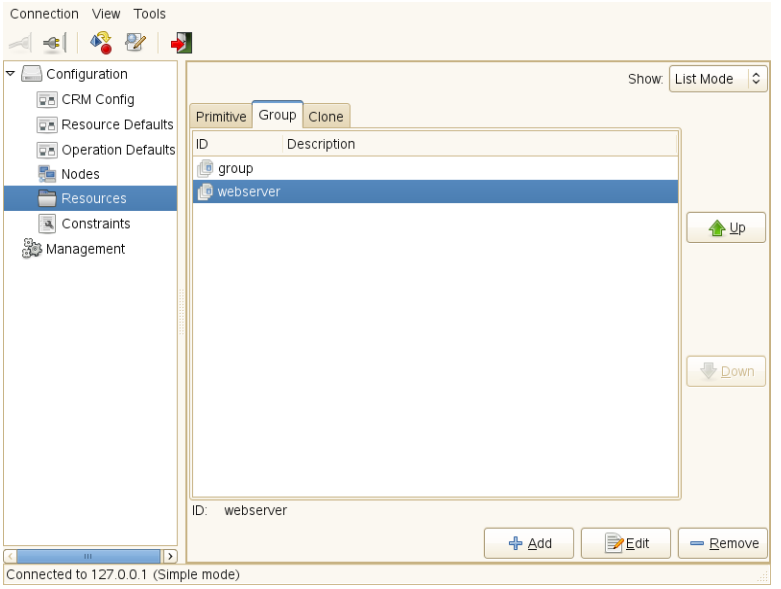
7. In the next window, you can continue adding sub-resources for the group by choosing *Primitive* again and clicking *OK*.

   When you do not want to add more primitives to the group, click *Cancel* instead. The next window shows a summary of the parameters that you have already defined for that group. The *Meta Attributes* and *Primitives* of the group are listed. The position of the resources in the *Primitive* tab represents the order in which the resources are started in the cluster.

8 As the order of resources in a group is important, use the *Up* and *Down* buttons to sort or resort the *Primitives* in the group.

9 If all parameters are set according to your wishes, click *OK* to finish the configuration of that group. The configuration dialog is closed and the main window shows the newly created or modified group.

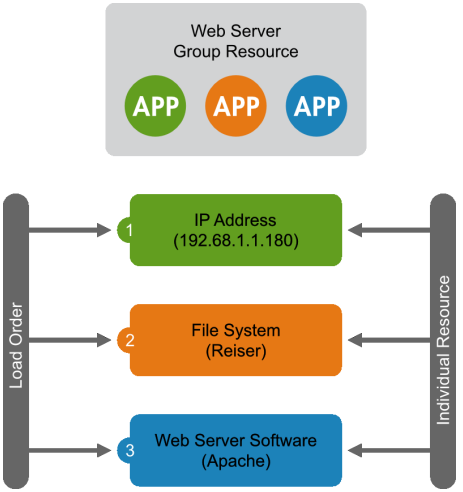
Figure 4.4 *Linux HA Management Client - Groups*



Example 4.1 *Resource Group for a Web Server*

An example for a resource group is a Web server that requires an IP address and a file system. In this case each component is a separate cluster resource that is combined into a cluster resource group. The resource group would then run on a server or servers, and in case of a software or hardware malfunction, fail over to another server in the cluster the same as an individual cluster resource.

Figure 4.5 *Group Resource*



In Adding a Resource Group (page 50), you learned how to create a resource group. Let us assume you already have created a resource group like explained above. Adding Resources to an Existing Group (page 52) shows you how to modify the group to match Example 4.1, “Resource Group for a Web Server” (page 51).

Procedure 4.10 *Adding Resources to an Existing Group*

1. Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, “Linux HA Management Client” (page 32).

2. In the left pane, switch to the *Resources* view and in the right pane, select the group to modify and click *Edit*. The next window shows the basic group parameters and the meta attributes and primitives already defined for that resource.

3. Click the *Primitives* tab and click *Add*.

4. In the next dialog, set the following parameters to add an IP address as sub-resource of the group:

   4a Enter a unique `ID`, for example, `my_ipaddress`.

   4b From the *Class* list, select *ocf* as resource agent class.

4c As *Provider* of your OCF resource agent, select *heartbeat*.

4d From the *Type* list, select *IPaddr* as resource agent.

4e Click *Forward*.

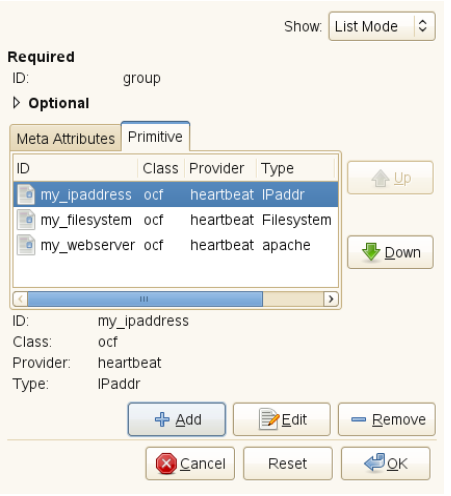
4f In the *Instance Attribute* tab, select the *IP* entry and click *Edit* (or double-click the *IP* entry).

4g As *Value*, enter the desired IP address, for example, `192.168.1.1`.

4h Click *OK* and *Apply*. The group configuration dialog shows the newly added primitive.

5 Add the next sub-resources (file system and Web server) by clicking *Add* again.

6 Set the respective parameters for each of the sub-resources similar to steps Step 4a (page 52) to Step 4h (page 53), until you have configured all sub-resources for the group.



As we configured the sub-resources already in the order in that they need to be started in the cluster, the order on the *Primitives* tab is already correct.

7 In case you need to change the resource order for a group, use the *Up* and *Down* buttons to resort the resources on the *Primitive* tab.

8 To remove a resource from the group, select the resource on the *Primitives* tab and click *Remove*.

9 Click *OK* to finish the configuration of that group. The configuration dialog is closed and the main window shows the modified group.

4.11 Configuring a Clone Resource

You may want certain resources to run simultaneously on multiple nodes in your cluster. To do this you must configure a resource as a clone. Examples of resources that might be configured as clones include STONITH and cluster file systems like OCFS2. You can clone any resource provided it is supported by the resource's Resource Agent. Clone resources may even be configured differently depending on which nodes they are hosted.

There are three types of resource clones:

Anonymous Clones
: These are the simplest type of clones. They behave identically anywhere they are running. Because of this, there can only be one instance of an anonymous clone active per machine.

Globally Unique Clones
: These resources are distinct entities. An instance of the clone running on one node is not equivalent to another instance on another node; nor would any two instances on the same node be equivalent.

Stateful Clones
: Active instances of these resources are divided into two states, active and passive. These are also sometimes referred to as primary and secondary, or master and slave. Stateful clones can be either anonymous or globally unique.

Procedure 4.11 *Adding or Modifying Clones*

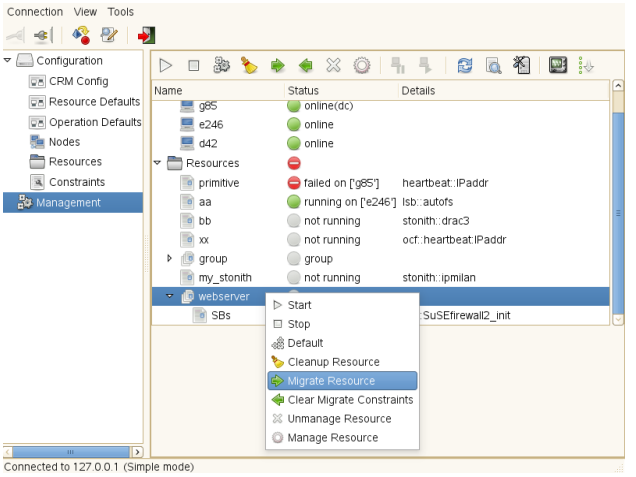
1. Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, “Linux HA Management Client” (page 32).
2. In the left pane, select *Resources* and click *Add* > *Clone*.
3. Enter a unique `ID` for the clone.
4. Below *Options*, set the *Initial state of resource*.
5. Activate the respective options you want to set for your clone and click *Forward*.
6. In the next step, you can either add a *Primitive* or a *Group* as sub-resources for the clone. These are created similar as described in Adding Primitive Resources (page 34) or Adding a Resource Group (page 50).
7. If all parameters in the clone configuration dialog are set according to your wishes, click *Apply* to finish the configuration of the clone.

4.12 Migrating a Cluster Resource

As mentioned in Section 4.5, “Specifying Resource Failover Nodes” (page 43), the cluster will fail over (migrate) resources automatically in case of software or hardware failures—according to certain parameters you can define (for example, migration threshold or resource stickiness). Apart from that, you can also manually migrate a resource to another node in the cluster resources manually.

Procedure 4.12 *Manually Migrating Resources*

1. Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, “Linux HA Management Client” (page 32).
2. Switch to the *Management* view in the left pane, then right-click the respective resource in the right pane and select *Migrate Resource*.



3 In the new window, select the node to which to move the resource to in *To Node*. This creates a location constraint with an `INFINITY` score for the destination node.

4 If you want to migrate the resource only temporarily, activate *Duration* and enter the time frame for which the resource should migrate to the new node. After the expiration of the duration, the resource *can* move back to its original location or it may stay where it is (depending on resource stickiness).

5 In cases where the resource cannot be migrated (if the resource's stickiness and constraint scores total more than `INFINITY` on the current node), activate the *Force* option. This forces the resource to move by creating a rule for the current location and a score of `-INFINITY`.

NOTE

This will prevent the resource from running on this node until the constraint is removed with *Clear Migrate Constraints* or the duration expires.

6 Click *OK* to confirm the migration.

To allow the resource to move back again, switch to the *Management*, right-click the resource view and select *Clear Migrate Constraints*. This uses the `crm_resource -U` command. The resource *can* move back to its original location or it may stay where it is (depending on resource stickiness). For more information, refer to crm_resource(8) (page 153) or *Configuration 1.0 Explained* , section *Resource Migration* , available from http://clusterlabs.org/wiki/Documentation.

4.13 For More Information

http://clusterlabs.org/
: Home page of Pacemaker, the cluster resource manager shipped with the High Availability Extension.

http://linux-ha.org
: Home page of the The High Availability Linux Project.

http://clusterlabs.org/wiki/Documentation
: *CRM Command Line Interface* : Introduction to the `crm` command line tool.

http://clusterlabs.org/wiki/Documentation
: *Configuration 1.0 Explained* : Explains the concepts used to configure Pacemaker. Contains comprehensive and very detailed information for reference.

Configuring Cluster Resources From Command Line

5

Like in Chapter 4 (page 31), a cluster resource must be created for every resource or application you run on the servers in your cluster. Cluster resources can include Web sites, e-mail servers, databases, file systems, virtual machines, and any other server-based applications or services you want to make available to users at all times.

You can either use the graphical HA Management Client utility, or the `crm` command line utility to create resources. This chapter introduces the several `crm` utilities.

5.1 Command Line Tools

After the installation there are several tools to administer a cluster. Usually you need the `crm` command only. This command has several subcommands. Run `crm help` to get an overview of all available commands. It has a thorough help system with embedded examples.

The `crm` tool has management abilities (the subcommands `resources` and `node`), and are used for configuration (`cib`, `configure`). Management subcommands take effect *immediately*, but configuration needs a final `commit`.

5.2 Debugging Your Configuration Changes

Before loading the changes back into the cluster, it is recommended to view your changes with `ptest`. The `ptest` can show a diagram of actions which would be induced by the changes to be committed. You need the `graphiz` package to display the diagrams. The following example is a transcript, adding a monitor operation:

```
# crm
crm(live)# configure
crm(live)configure# show fence-node2
primitive fence-node2 stonith:apcsmart \
        params hostlist="node2"
crm(live)configure# monitor fence-node2 120m:60s
crm(live)configure# show changed
primitive fence-node2 stonith:apcsmart \
        params hostlist="node2" \
        op monitor interval="120m" timeout="60s"
crm(live)configure# ptest
crm(live)configure# commit
```

5.3 Creating Cluster Resources

There are three types of RAs (Resource Agents) available with the cluster. First, there are legacy Heartbeat 1 scripts. High availability can make use of LSB initialization scripts. Finally, the cluster has its own set of OCF (Open Cluster Framework) agents. This documentation concentrates on LSB scripts and OCF agents.

To create a cluster resource use the `crm` tool. To add a new resource to the cluster, the general procedure is as follows:

1. Open a shell and become `root`.
2. Run `crm` to open the internal shell of `crm`. The prompt changes to `crm(live)#`.
3. Configure a primitive IP address:

   ```
   crm(live)# configure
   crm(live)configure# primitive myIP ocf:heartbeat:IPaddr \
        params ip=127.0.0.99 op monitor intervall=60s
   ```

The previous command configures a “primitive” with the name `myIP`. You need the class (here `ocf`), provider (`heartbeat`), and type (`IPaddr`). Furthermore this primitive expects some parameters like the IP address. You have to change the address to your setup.

4 Display and review the changes you have made:

```
crm(live)configure# show
```

To see the XML structure, use the following:

```
crm(live)configure# show xml
```

5 Commit your changes to take effect:

```
crm(live)configure# commit
```

5.3.1 LSB Initialization Scripts

All LSB scripts are commonly found in the directory `/etc/init.d`. They must have several actions implemented, which are at least `start`, `stop`, `restart`, `reload`, `force-reload`, and `status` as explained in http://www.linux-foundation.org/spec/refspecs/LSB_1.3.0/gLSB/gLSB/iniscrptact.html.

The configuration of those services is not standardized. If you intend to use an LSB script with High Availability, make sure that you understand how the respective script is configured. Often you can find some documentation to this in the documentation of the respective package in `/usr/share/doc/packages/`*PACKAGENAME*.

NOTE: Do Not Touch Services Used by High Availability

When used by High Availability, the service should not be touched by other means. This means that it should not be started or stopped on boot, reboot, or manually. However, if you want to check if the service is configured properly, start it manually, but make sure that it is stopped again before High Availability takes over.

Before using an LSB resource, make sure that the configuration of this resource is present and identical on all cluster nodes. The configuration is not managed by High Availability. You must take care of that yourself.

5.3.2 OCF Resource Agents

All OCF agents are located in `/usr/lib/ocf/resource.d/heartbeat/`. These are small programs that have a functionality similar to that of LSB scripts. However, the configuration is always done with environment variables. All OCF Resource Agents are required to have at least the actions `start`, `stop`, `status`, `monitor`, and `meta-data`. The `meta-data` action retrieves information about how to configure the agent. For example, if you want to know more about the `IPaddr` agent, use the command:

```
OCF_ROOT=/usr/lib/ocf /usr/lib/ocf/resource.d/heartbeat/IPaddr meta-data
```

The output is lengthy information in a simple XML format. You can validate the output with the `ra-api-1.dtd` DTD. Basically this XML format has three sections—first several common descriptions, second all the available parameters, and last the available actions for this agent.

This output is meant to be machine-readable, not very readable. For this reason, the `crm` tool contains the `ra` command to get different information about resource agents:

```
# crm
crm(live)# ra
crm(live)ra#
```

The command `classes` gives you a list of all classes and providers:

```
crm(live)ra# classes
stonith
lsb
ocf / lvm2 ocfs2 heartbeat pacemaker
heartbeat
```

To get an overview about all available resource agents for a class (and provider) use `list`:

```
crm(live)ra# list ocf
AudibleAlarm       ClusterMon         Delay              Dummy
Filesystem         ICP                IPaddr             IPaddr2
IPsrcaddr          IPv6addr           LVM                LinuxSCSI
MailTo             ManageRAID         ManageVE           Pure-FTPd
Raid1              Route              SAPDatabase        SAPInstance
SendArp            ServeRAID          SphinxSearchDaemon Squid
...
```

More information about a resource agent can be viewed with `meta`:

```
crm(live)ra# meta Filesystem ocf heartbeat
Filesystem resource agent (ocf:heartbeat:Filesystem)

Resource script for Filesystem. It manages a Filesystem on a shared storage
medium.

Parameters (* denotes required, [] the default):
...
```

You can leave the viewer by pressing Q. Find a configuration example at Chapter 6, *Setting Up a Simple Testing Resource* (page 75).

5.3.3 Example Configuration for an NFS Server

To set up the NFS server, three resources are needed: a file system resource, a drbd resource, and a group of an NFS server and an IP address. The following subsection shows you how to do it.

Setting Up a File System Resource

The `filesystem` resource is configured as an OCF primitive resource. It has the task to mount and unmount a device to a directory on start and stop requests. In this case, the device is `/dev/drbd0` and the directory to use as mount point is `/srv/failover`. The file system used is `xfs`.

Use the following commands in the `crm` shell to configure a filesystem resource:

```
crm(live)# configure
crm(live)configure# primitive filesystem_resource \
    ocf:heartbeat:Filesystem \
    params device=/dev/drbd0 directory=/srv/failover fstype=xfs
```

Configuring drbd

Before starting with the drbd High Availability configuration, set up a `drbd` device manually. Basically this is configuring drbd in `/etc/drbd.conf` and letting it synchronize. The exact procedure for configuring drbd is described in the *Storage Administration Guide*. For now, assume that you configured a resource `r0` that may be accessed at the device `/dev/drbd0` on both of your cluster nodes.

The drbd resource is an OCF master slave resource. This can be found in the description of the metadata of the drbd RA. However, more important is that there are the actions promote and demote in the actions section of the metadata. These are mandatory for master slave resources and commonly not available to other resources.

For High Availability, master slave resources may have multiple masters on different nodes. It is even possible to have a master and slave on the same node. Therefore, configure this resource in a way that there is exactly one master and one slave, each running on different nodes. Do this with the meta attributes of the master resource. Master slave resources are a special kind of clone resources in High Availability. Every master and every slave counts as a clone.

Use the following commands in the crm shell to configure a master slave resource:

```
crm(live)# configure
crm(live)configure# primitive drbd_r0 ocf:heartbeat:drbd params
crm(live)configure# ms drbd_resource drbd_r0 \
   meta clone_max=2 clone_node_max=1 master_max=1 master_node_max=1 notify=true
crm(live)configure# commit
```

NFS Server and IP Address

To make the NFS server always available at the same IP address, use an additional IP address as well as the ones the machines use for their normal operation. This IP address is then assigned to the active NFS server in addition to the system's IP address.

The NFS server and the IP address of the NFS server should always be active on the same machine. In this case, the start sequence is not very important. They may even be started at the same time. These are the typical requirements for a group resource.

Before starting the High Availability RA configuration, configure the NFS server with YaST. Do not let the system start the NFS server. Just set up the configuration file. If you want to do that manually, see the manual page exports(5) (man 5 exports). The configuration file is /etc/exports. The NFS server is configured as an LSB resource.

Configure the IP address completely with the High Availability RA configuration. No additional modification is necessary in the system. The IP address RA is an OCF RA.

```
crm(live)# configure
crm(live)configure# primitive nfs_resource lsb:nfsserver
crm(live)configure# primitive ip_resource ocf:heartbeat:IPaddr \
      params ip=10.10.0.1
```

```
crm(live)configure# group nfs_group nfs_resource ip_resource
crm(live)configure# commit
crm(live)configure# end
crm(live)# quit
```

5.4 Creating a STONITH Resource

From the `crm` perspective, a STONITH device is just another resource. To create a STONITH resource, proceed as follows:

1. Run the `crm` command as system administrator. The prompt changes to `crm(live)`.

2. Get a list of all STONITH types with the following command:

```
crm(live)# ra list stonith
apcmaster               apcsmart                baytech
cyclades                drac3                   external/drac5
external/hmchttp        external/ibmrsa         external/ibmrsa-telnet
external/ipmi           external/kdumpcheck     external/rackpdu
external/riloe          external/sbd            external/ssh
external/vmware         external/xen0           external/xen0-ha
ibmhmc                  ipmilan                 meatware
null                    nw_rpc100s              rcd_serial
rps10                   ssh                     suicide
```

3. Choose a STONITH type from the above list and view the list of possible options. Use the following command (press Q to close the viewer):

```
crm(live)# ra meta external/ipmi stonith
IPMI STONITH external device (stonith:external/ipmi)

IPMI-based host reset

Parameters (* denotes required, [] the default):
...
```

4. Create the STONITH resource with the class `stonith`, the type you have chosen in Step 3, and the respective parameters if needed, for example:

```
crm(live)# configure
crm(live)configure# primitive my-stonith stonith:external/ipmi \
   meta target-role=Stopped \
   operations my_stonith-operations \
     op monitor start-delay=15 timeout=15 hostlist='' \
                pduip='' community=''
```

5.5 Configuring Resource Constraints

Having all the resources configured is only one part of the job. Even if the cluster knows all needed resources, it might still not be able to handle them correctly. For example, it would not make sense to try to mount the file system on the slave node of drbd (in fact, this would fail with drbd). To inform the cluster about these things, define constraints.

In High Availability, there are three different kinds of constraints available:

- Locational constraints that define on which nodes a resource may be run (in the `crm` shell with the command `location`).
- Collocational constraints that tell the cluster which resources may or may not run together on a node (`colocation`).
- Ordering constraints to define the sequence of actions (`order`).

5.5.1 Locational Constraints

This type of constraint may be added multiple times for each resource. All `rsc_location` constraints are evaluated for a given resource. A simple example that increases the probability to run a resource with the ID `fs1-loc` on the node with the name `earth` to 100 would be the following:

```
crm(live)configure# location fs1-loc fs1 100: earth
```

5.5.2 Collocational Constraints

The `colocation` command is used to define what resources should run on the same or on different hosts. Usually it is very common to use the following sequence:

```
crm(live)configure# order rsc1 rsc2
crm(live)configure# colocation rsc2 rsc1
```

It is only possible to set a score of either +INFINITY or -INFINITY, defining resources that must always or must never run on the same node. For example, to run the two resources with the IDs `filesystem_resource` and `nfs_group` always on the same host, use the following constraint:

```
crm(live)configure# colocation nfs_on_filesystem inf: nfs_group
filesystem_resource
```

For a master slave configuration, it is necessary to know if the current node is a master in addition to running the resource locally. This can be checked with an additional `to_role` or `from_role` attribute.

5.5.3 Ordering Constraints

Sometimes it is necessary to provide an order in which services must start. For example, you cannot mount a file system before the device is available to a system. Ordering constraints can be used to start or stop a service right before or after a different resource meets a special condition, such as being started, stopped, or promoted to master. Use the following commands in the `crm` shell to configure an an ordering constraint:

```
crm(live)configure# order nfs_after_filesystem mandatory: group_nfs
filesystem_resource
```

5.5.4 Constraints for the Example Configuration

The example used for this chapter would not work as expected without additional constraints. It is essential that all resources run on the same machine as the master of the drbd resource. Another thing that is critical is that the drbd resource must be master before any other resource starts. Trying to mount the drbd device when drbd is not master simply fails. The constraints that must be fulfilled look like the following:

- The file system must always be on the same node as the master of the drbd resource.

  ```
  crm(live)configure# colocation filesystem_on_master inf: \
      filesystem_resource drbd_resource:Master
  ```

- The NFS server as well as the IP address must be on the same node as the file system.

  ```
  crm(live)configure# colocation nfs_with_fs inf: \
     nfs_group filesystem_resource
  ```

- The NFS server as well as the IP address start after the file system is mounted:

```
crm(live)configure# order nfs_second mandatory: \
   filesystem_resource nfs_group
```

- The file system must be mounted on a node after the drbd resource is promoted to master on this node.

```
crm(live)configure# order drbd_first inf: \
    drbd_resource:promote filesystem_resource
```

5.6 Specifying Resource Failover Nodes

To determine a resource failover, use the meta attribute migration-threshold. For example:

```
crm(live)configure# location r1-node1 r1 100: node1
```

Normally r1 prefers to run on node1. If it fails there, migration-threshold is checked and compared to the failcount. If failcount >= migration-threshold then it is migrated to the node with the next best preference.

Start failures set the failcount to INFINITY depends on the `start-failure-is-fatal` option. Stop failures cause fencing. If there's no STONITH defined, then the resource will not migrate at all.

5.7 Specifying Resource Failback Nodes (Resource Stickiness)

A rsc may failback after it has been migrated due to the number of failures only when the administrator resets the failcount or the failures have been expired (see failure-timeout meta attribute).

```
crm resource failcount RSC delete NODE
```

5.8 Configuring Resource Monitoring

To monitor a resource, there are two possibilities: either define monitor operation with the `op` keyword or use the `monitor` command. The following example configures an Apache resource and monitors it for every 30 minutes with the `op` keyword:

```
crm(live)configure# primitive apache apache \
  params ... \
  op monitor interval=60s timeout=30s
```

The same can be done with:

```
crm(live)configure# primitive apache apache \
   params ...
crm(live)configure# monitor apache 60s:30s
```

5.9 Starting a New Cluster Resource

To start a new cluster resource you need the respective identifier. Proceed as follows:

1. Run the `crm` command as system administrator. The prompt changes to `crm(live)`.
2. Search for the respective resource with the command `status`.
3. Start the resource with:

   ```
   crm(live)# resource start ID
   ```

5.10 Removing a Cluster Resource

To remove a cluster resource you need the respective identifier. Proceed as follows:

1. Run the `crm` command as system administrator. The prompt changes to `crm(live)`.
2. Run the following command to get a list of your resources:

   ```
   crm(live)# resource status
   ```

For example, the output can look like this (whereas myIP is the respective identifier of your resource):

```
myIP    (ocf::IPaddr:heartbeat) ...
```

3 Delete the resource with the respective identifier (which implies a `commit` too):

```
crm(live)# configure delete YOUR_ID
```

4 Comit the changes:

```
crm(live)# configure commit
```

5.11 Configuring a Cluster Resource Group

One of the most common elements of a cluster is a set of resources that need to be located together, start sequentially and stop in the reverse order. To simplify this configuration we support the concept of groups. The following example creates two primitives (an IP address and an email resource):

1 Run the `crm` command as system administrator. The prompt changes to `crm(live)`.

2 Configure the primitives:

```
crm(live)# configure
crm(live)configure# primitive Public-IP ocf:IPaddr:heartbeat \
   params ip=1.2.3.4
crm(live)configure# primitive Email lsb:exim
```

3 Group the primitives with their respective identifiers in the correct order:

```
crm(live)configure# group shortcut Public-IP Email
```

5.12 Configuring a Clone Resource

Clones were initially conceived as a convenient way to start N instances of an IP resource and have them distributed throughout the cluster for load balancing. They have turned out to quite useful for a number of purposes including integrating with DLM, the fencing subsystem and OCFS2. You can clone any resouce provided the resource agent supports it.

These types of cloned resources exist:

Anonymous Resources
: Anonymous clones are the simplest type. These resources behave completely identical everywhere they are running. Because of this, there can only be one copy of an anonymous clone active per machine.

Multi-State Resources
: Multi-state resources are a specialization of clones. They allow the instances to be in on of two operating modes. These modes are called “master” and “slave” but can mean whatever you wish them to mean. The only limitation is that when an instance is started, it must come up in a slave state.

5.12.1 Creating Anonymous Clone Resources

To create an anonymouse clone resource, first create a primitive resource and then refer to it with the `clone` command. Do the following:

1. Run the `crm` command as system administrator. The prompt changes to `crm(live).`

2. Configure the primitive, for example:

```
crm(live)# configure
crm(live)configure# primitive Apache lsb:apache
```

3. Clone the primitive:

```
crm(live)configure# clone apache-clone Apache \
   meta globally-unique=false
```

5.12.2 Creating Stateful/Multi-State Clone Resources

To create an stateful clone resource, first create a primitive resource and then the master-slave resource.

1. Run the `crm` command as system administrator. The prompt changes to `crm(live).`
2. Configure the primitive. Change the intervals if needed:

```
crm(live)# configure
crm(live)configure# primitive myRSC ocf:myCorp:myAppl \
  operations foo \
    op monitor interval=60 \
    op monitor interval=61 role=Master
```

3. Create the master slave resource:

```
crm(live)configure# clone apache-clone Apache \
   meta globally-unique=false
```

5.13 Migrating a Cluster Resource

Although resources are configured to automatically fail over (or migrate) to other nodes in the cluster in the event of a hardware or software failure, you can also manually migrate a resource to another node in the cluster using either the Linux HA Management Client or the command line.

1. Run the `crm` command as system administrator. The prompt changes to `crm(live).`
2. To migrate a resource named `ipaddress1` to a cluster node named `node2`, enter:

```
crm(live)# resource
crm(live)resource# migrate ipaddress1 node2
```

5.14 Testing with Shadow Configuration

NOTE: For Experienced Administrators Only

Although the concept is easy, it is nevertheless recommended to use shadow configuration only, when you really need them and got some experience with High Availability.

A shadow configuration is used to test different configuration scenarios. If you have created several shadow configuration, you can test it one by one to see the effect of your changes.

The usual process looks like this:

1. User starts the `crm` tool.
2. You switch to the `configure` subcommand:

   ```
   crm(live)# configure
   crm(live)configure#
   ```

3. Now you can make your changes. However, when you figure out that they are risky or you want to apply them later, you can save them into a new shadow configuration:

   ```
   crm(live)configure# cib new myNewConfig
   INFO: myNewConfig shadow CIB created
   crm(myNewConfig)configure# commit
   ```

4. After you have created the shadow configuration, you can make your changes.
5. To switch back to the live cluster configuration, use this command:

   ```
   crm(myNewConfig)configure# cib use
   crm(live)configure#
   ```

5.15 For More Information

http://linux-ha.org
: Homepage of High Availability Linux

http://www.clusterlabs.org/mediawiki/images/8/8d/Crm_cli.pdf
: Gives you an introduction to the CRM CLI tool

http://www.clusterlabs.org/mediawiki/images/f/fb/Configuration_Explained.pdf
: Explains the Pacemaker configuration

6 Setting Up a Simple Testing Resource

After your cluster is installed and set up as described in Chapter 3, *Installation and Basic Setup with YaST* (page 23) and you have learned how to configure resources either with the GUI or from command line, this chapter provides a basic example for the configuration of a simple resource: an IP address. It demonstrates both approaches to do so, using either the Linux HA Management Client or the `crm` command line tool.

For the following example, we assume that your cluster consists of at least two nodes.

6.1 Configuring a Resource with the GUI

Creating a sample cluster resource and migrating it to another server can help you test to ensure your cluster is functioning properly. A simple resource to configure and migrate is an IP address.

Procedure 6.1 *Creating an IP Address Cluster Resource*

1. Start the Linux HA Management Client and log in to the cluster as described in Section 4.1, "Linux HA Management Client" (page 32).

2. In the left pane, switch to the *Resources* view and in the right pane, select the group to modify and click *Edit*. The next window shows the basic group parameters and the meta attributes and primitives already defined for that resource.

3. Click the *Primitives* tab and click *Add*.

4. In the next dialog, set the following parameters to add an IP address as sub-resource of the group:

   4a. Enter a unique `ID`, for example, `myIP`.

   4b. From the *Class* list, select *ocf* as resource agent class.

   4c. As *Provider* of your OCF resource agent, select *heartbeat*.

   4d. From the *Type* list, select *IPaddr* as resource agent.

   4e. Click *Forward*.

   4f. In the *Instance Attribute* tab, select the *IP* entry and click *Edit* (or double-click the *IP* entry).

   4g. As *Value*, enter the desired IP address, for example, `10.10.0.1` and click *OK*.

   4h. *Add* a new instance attribute and specify `nic` as *Name* and `eth0` as *Value*, then click *OK*.

   The name and value are dependent on your hardware configuration and what you chose for the media configuration during the installation of the High Availability Extension software.

5. If all parameters are set according to your wishes, click *OK* to finish the configuration of that resource. The configuration dialog is closed and the main window shows the modified resource.

To start the resource with the Linux HA Management Client, select *Management* in the left pane. In the right pane, right-click the resource and select *Start* (or start it from the toolbar).

To migrate the IP address resource to another node (`saturn`) proceed as follows:

Procedure 6.2 *Migrating Resources to Another Node*

1. Switch to the *Management* view in the left pane, then right-click the IP address resource in the right pane and select *Migrate Resource*.

2. In the new window, select `saturn` from the *To Node* drop-down list to move the selected resource to the node `saturn`.

3. If you want to migrate the resource only temporarily, activate *Duration* and enter the time frame for which the resource should migrate to the new node.

4. Click *OK* to confirm the migration.

6.2 Manual Configuration of a Resource

Resources are any type of service that a computer provides. Resources are known to High Availability when they may be controlled by RAs (Resource Agents), which are LSB scripts, OCF scripts, or legacy Heartbeat 1 resources. All resources can be configured with the `crm` command or as XML in the CIB (Cluster Information Base) in the `resources` section. For an overview of available resources, look at Chapter 15, *HA OCF Agents* (page 189).

To add an IP address `10.10.0.1` as a resource to the current configuration, use the `crm` command:

Procedure 6.3 *Creating an IP Address Cluster Resource*

1. Run the `crm` command as system administrator. The prompt changes to `crm(live).`

2. Switch to the `configure` subcommand:

```
crm(live)# configure
```

3. Create an IP address resource:

```
crm(live)configure# resource
primitive myIP ocf:heartbeat:IPaddr params ip=10.10.0.1
```

NOTE

When configuring a resource with High Availability, the same resource should not be initialized by `init`. High availability is be responsible for all service start or stop actions.

If the configuration was successful, a new resource appears in `crm_mon` that is started on a random node of your cluster.

To migrate a resource to another node, do the following:

Procedure 6.4 *Migrating Resources to Another Node*

1. Start a shell and become the user `root`.
2. Migrate your resource `myip` to node `saturn`:

```
crm resource migrate myIP saturn
```

Adding or Modifying Resource Agents

7

All tasks that should be managed by a cluster must be available as a resource. There are two major groups that should be distinguished: resource agents and STONITH agents. For both categories, you can add your own agents, extending the abilities of the cluster to your own needs.

7.1 STONITH Agents

A cluster sometimes detects that one of the nodes is misbehaving and needs to remove it. This is called *fencing* and is commonly done with a STONITH resource. All STONITH resources reside in `/usr/lib/stonith/plugins` on each node.

WARNING: SSH and STONITH Are Not Supported

It is impossible to know how SSH might react to other system problems. For this reason, SSH and STONITH agent are not supported for a production environment.

To get a list of all currently available STONITH devices (from the software side), use the command `stonith -L`.

Unfortunately, there is no documentation about writing STONITH agents yet. If you want to write new STONITH agents, consult the examples available in the source of the `heartbeat-common` package.

7.2 Writing OCF Resource Agents

All OCF RAs are available in `/usr/lib/ocf/resource.d/`, see Section 14.1, "Supported Resource Agent Classes" (page 181) for more information. To avoid name clashes, create a different subdirectory when creating new resource agents. For example, if you have a resource group `kitchen` with the resource `coffee_machine`, add this resource to the directory `/usr/lib/ocf/resource.d/kitchen/`. To access this RA, execute the command `crm`:

```
configure
primitive coffee_1 ocf:coffee_machine:kitchen ...
```

When implementing your own OCF RA, provide several actions for this agent. More details about writing OCF resource agents can be found at `http://www.linux-ha.org/OCFResourceAgent`. Find special information about several concepts of High Availability 2 at Chapter 1, *Conceptual Overview* (page 3).

8 Fencing and STONITH

Fencing is a very important concept in computer clusters for HA (High Availability). A cluster sometimes detects that one of the nodes is misbehaving and needs to remove it. This is called *fencing* and is commonly done with a STONITH resource. Fencing may be defined as a method to bring an HA cluster to a known state.

Every resource in a cluster has a state attached, for example: "resource r1 is started on node1". In an HA cluster, such a state implies that "resource r1 is stopped on all nodes but node1", because an HA cluster must make sure that every resource may be started on at most one node. Every node must report every change that happens to a resource. The cluster state is thus a collection of resource states and node states.

If, for whatever reason, a state of some node or resource cannot be established with certainty, fencing comes in. Even when the cluster does not know what is happening on some node, fencing can make sure that that node does not run any important resources.

8.1 Classes of Fencing

There are two classes of fencing: resource level and node level fencing. The latter is the primary subject of this chapter.

Resource Level Fencing
: Using resource level fencing the cluster can make sure that a node cannot access one or more resources. One typical example is a SAN, where a fencing operation changes rules on a SAN switch to deny access from the node.

The resource level fencing may be achieved using normal resources on which the resource you want to protect depends. Such a resource would simply refuse to start on this node and therefore resources which depend on will not run on the same node.

Node Level Fencing
: Node level fencing makes sure that a node does not run any resources at all. This is usually done in a very simple, yet brutal way: the node is simply reset using a power switch. This may ultimately be necessary because the node may not be responsive at all.

8.2 Node Level Fencing

In SUSE® Linux Enterprise High Availability Extension, the fencing implementation is STONITH (Shoot The Other Node in the Head). It provides the node level fencing. The High Availability Extension includes the `stonith` command line tool, an extensible interface for remotely powering down a node in the cluster. For an overview of the available options, run `stonith --help` or refer to the man page of `stonith` for more information.

8.2.1 STONITH Devices

To use node level fencing, you first of all need to have a fencing device. To get a list of STONITH devices which are supported by the High Availability Extension, run the following command as `root` on any of the nodes:

```
stonith -L
```

STONITH devices may be classified into the following categories:

Power Distribution Units (PDU)
: Power Distribution Units are an essential element in managing power capacity and functionality for critical network, server and data center equipment. They can provide remote load monitoring of connected equipment and individual outlet power control for remote power recycling.

Uninterruptible Power Supplies (UPS)
: An uninterruptible power supply provides emergency power to connected equipment by supplying power from a separate source when utility power is not available.

Blade Power Control Devices
: If you are running a cluster on a set of blades, then the power control device in the blade enclosure is the only candidate for fencing. Of course, this device must be capable of managing single blade computers.

Lights-out Devices
: The lights-out devices (IBM RSA, HP iLO, Dell DRAC) are becoming increasingly popular and in future they may even become standard equipment of of-the-shelf computers. However, they are inferior to UPS devices, because they share a power supply with their host (a cluster node). If a node stays without power, the device supposed to control it would be just as useless. In that case, the CRM will try to fence the node in vain and this will continue forever because all other resource operations would wait for the fencing/STONITH operation to succeed.

Testing Devices
: Testing devices are used exclusively for testing purposes. They are usually more gentle on the hardware. Once the cluster goes into production, they must be replaced with real fencing devices.

The choice of the STONITH device depends mainly on your budget and the kind of hardware you use.

8.2.2 STONITH Implementation

The STONITH implementation of SUSE® Linux Enterprise High Availability Extension consists of two components:

stonithd
: stonithd is a daemon which can be accessed by the local processes or over the network. It accepts commands which correspond to fencing operations: reset, power-off, and power-on. It can also check the status of the fencing device.

  The stonithd daemon runs on every node in the CRM HA cluster. The stonithd instance running on the DC node receives a fencing request from the CRM. It is up to this and other stonithd programs to carry out the desired fencing operation.

STONITH Plug-ins

For every supported fencing device there is a STONITH plug-in which is capable of controlling that device. A STONITH plug-in is the interface to the fencing device. All STONITH plug-ins reside in `/usr/lib/stonith/plugins` on each node. All STONITH plug-ins look the same to stonithd, but are quite different on the other side reflecting the nature of the fencing device.

Some plug-ins support more than one device. A typical example is `ipmilan` (or `external/ipmi`) which implements the IPMI protocol and can control any device which supports this protocol.

8.3 STONITH Configuration

To configure fencing, you need to configure one or more STONITH resources—the stonithd daemon requires no configuration. All configuration is stored in the CIB. A STONITH resource is a resource of class `stonith` (see Section 14.1, “Supported Resource Agent Classes” (page 181)). STONITH resources are a representation of STONITH plug-ins in the CIB. Apart from the fencing operations, the STONITH resources can be started, stopped and monitored, just like any other resource. Starting or stopping STONITH resources means enabling and disabling STONITH in this case. Starting and stopping are thus only administrative operations and do not translate to any operation on the fencing device itself. However, monitoring does translate to device status.

STONITH resources can be configured just like any other resource. For more information about configuring resources, see Section 4.3, “Creating STONITH Resources” (page 37) or Section 5.4, “Creating a STONITH Resource” (page 65).

The list of parameters (attributes) depends on the respective STONITH type. To view a list of parameters for a specific device, use the `stonith` command:

```
stonith -t stonith-device-type -n
```

For example, to view the parameters for the `ibmhmc` device type, enter the following:

```
stonith -t ibmhmc -n
```

To get a short help text for the device, use the `-h` option:

```
stonith -t stonith-device-type -h
```

8.3.1 Example STONITH Resource Configurations

In the following, find some example configurations written in the syntax of the `crm` command line tool. To apply them, put the sample in a text file (for example, `sample.txt`) and run:

```
crm < sample.txt
```

For more information about configuring resources with the `crm` command line tool, refer to Chapter 5, *Configuring Cluster Resources From Command Line* (page 59).

WARNING: Testing Configurations

Some of the examples below are for demonstration and testing purposes only. Do not use any of the `Testing Configuration` examples in real-life cluster scenarios.

Example 8.1 *Testing Configuration*

```
configure
primitive st-null stonith:null \
params hostlist="node1 node2"
clone fencing st-null
commit
```

Example 8.2 *Testing Configuration*

An alternative configuration:

```
configure
 primitive st-node1 stonith:null \
 params hostlist="node1"
 primitive st-node2 stonith:null \
 params hostlist="node2"
 location l-st-node1 st-node1 -inf: node1
 location l-st-node2 st-node2 -inf: node2
 commit
```

This configuration example is perfectly alright as far as the cluster software is concerned. The only difference to a real world configuration is that no fencing operation takes place.

Example 8.3 *Testing Configuration*

A more realistic example, but still only for testing, is the following external/ssh configuration:

```
configure
 primitive st-ssh stonith:external/ssh \
 params hostlist="node1 node2"
 clone fencing st-ssh
 commit
```

This one can also reset nodes. The configuration is remarkably similar to the first one which features the null STONITH device. In this example, clones are used. They are a CRM/Pacemaker feature. A clone is basically a shortcut: instead of defining `n` identical, yet differently named resources, a single cloned resource suffices. By far the most common use of clones is with STONITH resources if the STONITH device is accessible from all nodes.

Example 8.4 *Configuration of an IBM RSA Lights-out Device*

The real device configuration is not much different, though some devices may require more attributes. An IBM RSA lights-out device might be configured like this:

```
configure
primitive st-ibmrsa-1 stonith:external/ibmrsa-telnet \
params nodename=node1 ipaddr=192.168.0.101 \
userid=USERID passwd=PASSW0RD
primitive st-ibmrsa-2 stonith:external/ibmrsa-telnet \
params nodename=node2 ipaddr=192.168.0.102 \
userid=USERID passwd=PASSW0RD
location l-st-node1 st-ibmrsa-1 -inf: node1
location l-st-node2 st-ibmrsa-2 -inf: node2
commit
```

In this example, location constraints are used because of the following reason: There is always certain probability that the STONITH operation is going to fail. Hence, a STONITH operation on the node which is the executioner too is not reliable. If the node is reset, then it cannot send the notification about the fencing operation outcome. The only way to do that is to assume that the operation is going to succeed and send the notification beforehand. But if the operation fails, we are in trouble. Therefore, stonithd refuses to kill its host by convention.

Example 8.5 *Configuration of an UPS Fencing Device*

The configuration of a UPS kind of fencing device is similar to the examples above, the details are left as an exercise to the reader. All UPS devices employ the same mechanics for fencing, but how the device itself is accessed, is different. Old UPS devices (those that were considered professional), used to have just a serial port, typically connected at 1200baud using a special serial cable. Many new ones still have a serial port, but often they also sport a USB interface or an ethernet interface. The kind of connection you can use depends on what the plug-in supports.

For example, compare the `apcmaster` with the `apcsmart` device by using the `stonith -t` *`stonith-device-type`* `-n` command:

```
stonith -t apcmaster -h
```

returns the following information:

```
STONITH Device: apcmaster - APC MasterSwitch (via telnet)
NOTE: The APC MasterSwitch accepts only one (telnet)
connection/session a time. When one session is active,
subsequent attempts to connect to the MasterSwitch will fail.
For more information see http://www.apc.com/
List of valid parameter names for apcmaster STONITH device:
ipaddr
login
 password
```

With

```
stonith -t apcsmart -h
```

you get the following output:

```
STONITH Device: apcsmart - APC Smart UPS
(via serial port - NOT USB!).
Works with higher-end APC UPSes, like
Back-UPS Pro, Smart-UPS, Matrix-UPS, etc.
(Smart-UPS may have to be >= Smart-UPS 700?).
See http://www.networkupstools.org/protocols/apcsmart.html
for protocol compatibility details.
For more information see http://www.apc.com/
List of valid parameter names for apcsmart STONITH device:
ttydev
hostlist
```

The first plug-in supports APC UPS with a network port and telnet protocol. The second plug-in uses the APC SMART protocol over the serial line which is supported by many different APC UPS product lines.

8.3.2 Constraints Versus Clones

In Section 8.3.1, “Example STONITH Resource Configurations” (page 85) you learned that there are several ways to configure a STONITH resource: using constraints, or clones or both. The choice which construct to use for configuration depends on several factors (nature of the fencing device, number of host managed by the device, number of cluster nodes) and last but not least, also on personal preference.

In short: if clones are safe to use with your configuration and if they reduce the configuration, then use cloned STONITH resources.

8.4 Monitoring Fencing Devices

Just like any other resource, the STONITH class agents also support the monitor operation which is used for checking the status.

NOTE: Monitoring STONITH Resources

Monitoring STONITH resources is strongly recommended. Monitor them regularly, yet sparingly.

Fencing devices are an indispensable part of an HA cluster, but the less you need to exercise them, the better. Power management equipment is known to be rather fragile on the communication side. Some devices give up if there was too much broadcast traffic on the wire. Some cannot handle more than ten or so connections per minute. Some get confused or depressed if two clients try to connect at the same time. Most cannot handle more than one session at the time.

So checking the fencing devices once every couple of hours should be enough in most cases. The probability that within those few hours there will be a need for a fencing operation and that the power switch would fail is usually low.

For detailed information on how to configure monitor operations, refer to Adding or Modifying Meta and Instance Attributes (page 36) for the GUI approach or to Section 5.8, “Configuring Resource Monitoring” (page 69) for the command line approach.

8.5 Special Fencing Devices

Apart from plug-ins which handle real devices, some STONITH plug-ins are a bit out of line and deserve special attention.

`external/kdumpcheck`
: Sometimes, it is important to get a kernel core dump. This plug-in can be used to check if the dump is in progress. If that is the case, then it will return true, as if the node has been fenced, which is actually true given that it cannot run any resources at the time. `kdumpcheck` is typically used in concert with another, real, fencing device. See `/usr/share/doc/packages/heartbeat/stonith/README_kdumpcheck.txt` for more details.

`external/sbd`
: This is a self-fencing device. It reacts to a so-called “poison pill” which can be inserted into a shared disk. On shared storage connection loss, it also makes the node commit suicide. See `http://www.linux-ha.org/SBD_Fencing` for more details.

`meatware`
: `meatware` requires help from a human to operate. Whenever invoked, `meatware` logs a CRIT severity message which shows up on the node's console. The operator then needs to make sure that the node is down and issue a `meatclient(8)` command. This tells `meatware` that it can inform the cluster that it may consider the node dead. See `/usr/share/doc/packages/heartbeat/stonith/README.meatware` for more information.

`null`
: This is an imaginary device used in various testing scenarios. It always behaves and claims that it has shot a node, but never does anything. Do not use it unless you know what you are doing.

`suicide`
: This is a software-only device, which can reboot a node it is running on, using the `reboot` command. This requires action by the node's operating system and can fail under certain circumstances. Therefore avoid using this device whenever possible (but it can be used on one-node clusters).

`suicide` and `null` are the only exceptions to the “do not shoot my host” rule.

8.6 For More Information

`/usr/share/doc/packages/heartbeat/stonith/`
: In your installed system, this directory holds README files for many STONITH plug-ins and devices.

`http://linux-ha.org/STONITH`
: Information about STONITH at the home page of the The High Availability Linux Project.

`http://linux-ha.org/fencing`
: Information about fencing at the home page of the The High Availability Linux Project.

`http://linux-ha.org/ConfiguringStonithPlugins`
: Information about STONITH plug-ins at the home page of the The High Availability Linux Project.

`http://linux-ha.org/CIB/Idioms`
: Information about STONITH at the home page of the The High Availability Linux Project.

`http://clusterlabs.org/wiki/Documentation`, *Configuration 1.0 Explained*
: Explains the concepts used to configure Pacemaker. Contains comprehensive and very detailed information for reference.

`http://techthoughts.typepad.com/managing_computers/2007/10/split-brain-quo.html`
: Article explaining the concepts of split brain, quorum and fencing in HA clusters.

Part III. Storage and Data Replication

9 Oracle Cluster File System 2

Oracle Cluster File System 2 (OCFS2) is a general-purpose journaling file system that is fully integrated in the Linux 2.6 kernel and later. OCFS2 allows you to store application binary files, data files, and databases on devices on shared storage. All nodes in a cluster have concurrent read and write access to the file system. A user-space control daemon, managed via a clone resource, provides the integration with the HA stack, in particular OpenAIS and the DLM.

9.1 Features and Benefits

In SUSE Linux Enterprise Server 10 and later, OCFS2 can be used for example for the following storage solutions:

- General applications and workloads
- XEN image store in a cluster

  XEN virtual machines and virtual servers can be stored on OCFS2 volumes that are mounted by cluster servers to provide quick and easy portability of XEN virtual machines between servers.
- LAMP (Linux, Apache, MySQL, and PHP | Perl | Python) stacks

In addition, it is fully integrated with OpenAIS.

As a high-performance, symmetric, parallel cluster file system, OCFS2 supports the following functions:

- An application's files are available to all nodes in the cluster. Users simply install it once on an OCFS2 volume in the cluster.
- All nodes can concurrently read and write directly to storage via the standard file system interface, enabling easy management of applications that run across a cluster.
- File access is coordinated through the Distributed Lock Manager (DLM).

  DLM control is good for most cases, but an application's design might limit scalability if it contends with the DLM to coordinate file access.
- Storage backup functionality is available on all back-end storage. An image of the shared application files can be easily created, which can help provide effective disaster recovery.

OCFS2 also provides the following capabilities:

- Metadata caching.
- Metadata journaling.
- Cross-node file data consistency.
- Support for multiple-block sizes (each volume can have a different block size) up to 4 KB, for a maximum volume size of 16 TB.
- Support for up to 16 cluster nodes.
- Asynchronous and direct I/O support for database files for improved database performance.

9.2 Management Utilities and Commands

OCFS2 utilities are described in the following table. For information about syntax for these commands, see their man pages.

Table 9.1 *OCFS2 Utilities*

OCFS2 Utility	Description
debugfs.ocfs2	Examines the state of the OCFS file system for the purpose of debugging.
fsck.ocfs2	Checks the file system for errors and optionally repairs errors.
mkfs.ocfs2	Creates an OCFS2 file system on a device, usually a partition on a shared physical or logical disk.
mounted.ocfs2	Detects and lists all OCFS2 volumes on a clustered system. Detects and lists all nodes on the system that have mounted an OCFS2 device or lists all OCFS2 devices.
tunefs.ocfs2	Changes OCFS2 file system parameters, including the volume label, number of node slots, journal size for all node slots, and volume size.

9.3 OCFS2 Packages

The OCFS2 kernel module (`ocfs2`) is installed automatically in SUSE Linux Enterprise Server 11 HAE. To use OCFS2, use YaST (or the command line if you prefer) to install the `ocfs2-tools` and the matching `ocfs2-kmp-*` packages on each node in the cluster.

1. Log in as the `root` user or equivalent, then open the YaST Control Center.
2. Select *Software* > *Add-On Products*.
3. Select *Add* to make the SUSE Linux Enterprise High Availability Extension available.
4. Select *Run Software Manager* and choose *Filter* > *Patterns*.
5. Select the *High Availability* pattern for installation.

6 Click *Accept* and follow the on-screen instructions.

9.4 Creating an OCFS2 Volume

Follow the procedures in this section to configure your system to use OCFS2 and to create OCFS2 volumes.

9.4.1 Prerequisites

Before you begin, do the following:

- Prepare the block devices you plan to use for your OCFS2 volumes. Leave the devices as free space.

  We recommend that you store application files and data files on different OCFS2 volumes, but it is only mandatory to do so if your application volumes and data volumes have different requirements for mounting.

- Make sure that the `ocfs2-tools` package is installed. Use YaST or command line methods to install them if they are not. For YaST instructions, see Section 9.3, “OCFS2 Packages” (page 95).

9.4.2 Configuring OCFS2 Services

Before you can create OCFS2 volumes, you must configure OCFS2 services. .

Follow the procedure in this section for one node in the cluster.

1 Open a terminal window and log in as the `root` user or equivalent.

2 Add the distributed lock manager configuration.

  2a Start the crm shell and create a new scratch configuration:

```
crm
cib new stack-glue
```

2b Create the DLM service and have it run on all machines in the cluster:

```
configure
primitive dlm ocf:pacemaker:controld op monitor interval=120s
clone dlm-clone dlm meta globally-unique=false interleave=true
end
```

2c Verify the changes you made to the cluster before commiting them:

```
cib diff
configure verify
```

2d Upload the configuration to the cluster and quit the shell:

```
cib commit stack-glue
quit
```

3 Add the O2CB configuration with *crm*.

3a Start the crm shell and create a new scratch configuration:

```
crm
cib new oracle-glue
```

3b Configure Pacemaker to start the o2cb service on every node in the cluster.

```
configure
primitive o2cb ocf:ocfs2:o2cb op monitor interval=120s
clone o2cb-clone o2cb meta globally-unique=false interleave=true
```

3c Make sure that Pacemaker only starts the o2cb service on nodes that also have a copy of the dlm service already running:

```
colocation o2cb-with-dlm INFINITY: o2cb-clone dlm-clone
order start-o2cb-after-dlm mandatory: dlm-clone o2cb-clone
end
```

3d Upload the configuration to the cluster and quit the shell:

```
cib commit stack-glue
quit
```

9.4.3 Creating an OCFS2 Volume

Creating an OCFS2 file system and adding new nodes to the cluster should be performed on only one of the nodes in the cluster.

1. Open a terminal window and log in as the `root` user or equivalent.
2. Check if the cluster is online with the command `crm_mon`.
3. Create and format the volume using one of the following methods:

   - Use the `mkfs.ocfs2` utility. For information about the syntax for this command, refer to the `mkfs.ocfs2` man page.

     To create a new OCFS2 file system on `/dev/sdb1` that supports up to 16 cluster nodes, use

     ```
     mkfs.ocfs2 -N 16 /dev/sdb1
     ```

   See the following table for recommended settings.

OCFS2 Parameter	Description and Recommendation
Volume label	A descriptive name for the volume to make it uniquely identifiable when it is mounted on different nodes. Use the `tunefs.ocfs2` utility to modify the label as needed.
Cluster size	Cluster size is the smallest unit of space allocated to a file to hold the data. Options are 4, 8, 16, 32, 64, 128, 256, 512, and 1024 KB. Cluster size cannot be modified after the volume is formatted.

OCFS2 Parameter	Description and Recommendation
	Oracle recommends a cluster size of 128 KB or larger for database volumes. Oracle also recommends a cluster size of 32 or 64 KB for Oracle Home.
Number of node slots	The maximum number of nodes that can concurrently mount a volume. For each of the nodes, OCFS2 creates separate system files, such as the journals, for each of the nodes. Nodes that access the volume can be a combination of little-endian architectures (such as x86, x86-64, and ia64) and big-endian architectures (such as ppc64 and s390x). Node-specific files are referred to as local files. A node slot number is appended to the local file. For example: journal:0000 belongs to whatever node is assigned to slot number 0. Set each volume's maximum number of node slots when you create it, according to how many nodes that you expect to concurrently mount the volume. Use the `tunefs.ocfs2` utility to increase the number of node slots as needed; the value cannot be decreased.
Block size	The smallest unit of space addressable by the file system. Specify the block size when you create the volume. Options are 512 bytes (not recommended), 1 KB, 2 KB, or 4 KB (recommended for most volumes). Block size cannot be modified after the volume is formatted.

9.5 Mounting an OCFS2 Volume

1. Open a terminal window and log in as the `root` user or equivalent.
2. Check if the cluster in online with the command `crm_mon`.

3 Use one of the following methods to mount the volume.

WARNING: Manual Mounted OCFS2 Devices

If you mount the ocfs2 file system manually for testing purposes, you are required to umount the file system again before starting to use it by means of OpenAIS.

- In the `ocfs2console`, select a device in the Available Devices list, click *Mount*, specify the directory mount point and mount options (optional), then click *OK*.
- Mount the volume from the command line, using the `mount` command.
- Use the cluster manager to mount the file system. The ocf resource `Filesystem` can be used for this task. For more details, see Mounting the File system with the Cluster Manager (page 100).

On a successful mount, the device list in the `ocfs2console` shows the mount point along with the device.

Option	Description
`datavolume`	Ensures that the Oracle processes open the files with the o_direct flag.
`nointr`	No interruptions. Ensures the IO is not interrupted by signals.

To configure the file system resource, use the following procedure:

Procedure 9.1 *Mounting the File system with the Cluster Manager*

1 Start the crm shell and create a new scratch configuration:

```
crm
cib new filesystem
```

2 Configure Pacemaker to mount the file system on every node in the cluster.

```
configure
primitive fs ocf:heartbeat:Filesystem \
   params device="/dev/sdb1" directory="/mnt/shared" fstype="ocfs2" \
   op monitor interval=120s
clone fs-clone fs meta interleave="true" ordered="true"
```

3 Make sure that Pacemaker only starts the o2cb clone resource on nodes that also have a clone of the o2cb resource already running:

```
colocation fs-with-o2cb INFINITY: fs-clone o2cb-clone
order start-fs-after-o2cb mandatory: o2cb-clone fs-clone
end
```

4 Upload the configuration to the cluster and quit the shell:

```
cib commit filesystem
quit
```

9.6 Additional Information

For information about using OCFS2, see the *OCFS2 User Guide [http://oss.oracle.com/projects/ocfs2/documentation/]* on the OCFS2 project at Oracle [http://oss.oracle.com/projects/ocfs2/].

10 Cluster LVM

When managing shared storage on a cluster, every node must be informed about changes that are done to the storage subsystem. The Linux Volume Manager 2 (LVM2), which is widely used to manage local storage, has been extended to support transparent management of volume groups across the whole cluster. Clustered volume groups can be managed using the same commands as local storage.

To implement cLVM, a shared storage setup — such as provided by a Fire Channel, FCoE, SCSI or iSCSI SAN or DRBD — must be available.

Internally, cLVM uses the Distributed Lock Manager (DLM) component of the cluster stack to coordinate access to the LVM2 metadata. As DLM in turn integrates with the other components of the stack such as fencing, the integrity of the shared storage is protected at all times.

cLVM does not coordinate access to the shared data itself; to do so, you must configure OCFS2 or other cluster-aware applications on top of the cLVM-managed storage.

10.1 Configuration of cLVM

To setup a cluster-aware volume group, several tasks must be completed successfully:

1. Change the locking type of LVM2 to be cluster-aware.

   Edit the file `/etc/lvm/lvm.conf` and locate the line:

   ```
   locking_type = 1
   ```

Change the locking type to 3, and write the configuration to disk. Copy this configuration to all nodes.

2 Include the clvmd resource as a clone in the pacemaker configuration, and make it depend on the DLM clone resource. A typical snippet from the crm configuration shell would look like this:

```
primitive dlm ocf:pacemaker:controld
primitive clvm ocf:lvm2:clvmd \
        params daemon_timeout="30"
clone dlm-clone dlm \
        meta target-role="Started" interleave="true" ordered="true"
clone clvm-clone clvm \
        meta target-role="Started" interleave="true" ordered="true"
colocation colo-clvm inf: clvm-clone dlm-clone
order order-clvm inf: dlm-clone clvm-clone
...
```

Before proceeding, confirm that these resources have started successfully in your cluster. You may use crm_mon or the GUI to check the running services.

3 Prepare the physical volume for LVM with the command:

```
pvcreate <physical volume path>
```

4 Create a cluster aware volume group:

```
vgcreate --clustered y <volume group name> <physical volume path>
```

5 Create logical volumes as needed, for example:

```
lvcreate --name testlv -L 4G <volume group name>
```

6 To ensure that the volume group is activated cluster-wide, configure a LVM resource as follows:

```
primitive vg1 ocf:heartbeat:LVM \
        params volgrpname="<volume group name>"
clone vg1-clone vg1 \
        meta interleave="true" ordered="true"
colocation colo-vg1 inf: vg1-clone clvm-clone
order order-vg1 inf: clvm-clone vg1-clone
```

7 If you want the volume group to only be activated exclusively on one node, use the following example; in this case, cLVM will protect all logical volumes within the VG from being activated on multiple nodes, as an additional measure of protection for non-clustered applications:

```
primitive vg1 ocf:heartbeat:LVM \
        params volgrpname="<volume group name>" exclusive="yes"
colocation colo-vg1 inf: vg1 clvm-clone
order order-vg1 inf: clvm-clone vg1
```

8 The logical volumes within the VG are now available as file system mounts or raw usage. Ensure that services using them must have proper dependencies to collocate them with and order them after the VG has been activated.

After finishing these configuration steps, the LVM2 configuration can be done just like on any standalone workstation.

10.2 Configuring Eligible LVM2 Devices Explicitly

When several devices seemingly share the same physical volume signature (as can be the case for multipath devices or drbd), it is recommended to explicitly configure the devices which LVM2 scans for PVs.

For example, if the command `vgcreate` uses the physical device instead of using the mirrored block device, DRBD will be confused which may result in a split brain condition for DRBD.

To deactivate a single device for LVM2, do the following:

1 Edit the file `/etc/lvm/lvm.conf` and search for the line starting with `filter`.

2 The patterns there are handled as regular expressions. A leading “a” means to accept a device pattern to the scan, a leading “r” rejects the devices that follow the device pattern.

3 To remove a device named `/dev/sdb1`, add the following expression to the filter rule:

```
"r|^/dev/sdb1$|"
```

The complete filter line will look like the following:

```
filter = [ "r|^/dev/sdb1$|", "r|/dev/.*/by-path/.*|",
"r|/dev/.*/by-id/.*|", "a/.*/" ]
```

A filter line, that accepts DRBD and MPIO devices but rejects all other devices would look like this:

```
filter = [ "a|/dev/drbd.*|", "a|/dev/.*/by-id/dm-uuid-mpath-.*|",
"r/.*/" ]
```

4 Write the configuration file and copy it to all cluster nodes.

10.3 For More Information

Much information is available from the pacemaker mailing list, available at http://www.clusterlabs.org/wiki/Help:Contents.

The official cLVM FAQ can be found at http://sources.redhat.com/cluster/wiki/FAQ/CLVM.

Distributed Replicated Block Device (DRBD)

11

DRBD allows you to create a mirror of two block devices that are located at two different sites across an IP network. When used with OpenAIS, DRBD supports distributed high-availability Linux clusters.

IMPORTANT

The data traffic between mirrors is not encrypted. For secure data exchange, you should deploy a virtual private network (VPN) solution for the connection.

Data on the primary device is replicated to the secondary device in a way that ensures that both copies of the data are always identical. When using a cluster aware file system such as ocfs2, it is also possible to run both nodes as primary devices.

By default, DRBD uses the TCP port 7788 for communications between DRBD nodes. Make sure that your firewall does not prevent communication on this port.

You must set up the DRBD devices before creating file systems on them. Everything to do with user data should be done solely via the `/dev/drbd<n>` device, not on the raw device, because DRBD uses the last 128 MB of the raw device for metadata. Make sure to create file systems only on the `/dev/drbd<n>` device, not on the raw device.

For example, if the raw device is 1024 MB in size, the DRBD device has only 896 MB available for data, with 128 MB hidden and reserved for the metadata. Any attempt to access the space between 896 MB and 1024 MB fails because it is not available for user data.

11.1 Installing DRBD Services

To install the needed packages for `drbd`, Install the High Availability Extension Add-On product on both SUSE Linux Enterprise Server machines in your networked cluster as described in Part I, “Installation and Setup” (page 1). Installing High Availability Extension also installs the `drbd` program files.

If you do not need the complete cluster stack but just want to use drbd, you may also add the High Availability Extension Add-On product, and proceed with installing `drbd`. This will also install the needed kernel modules.

11.2 Configuring the DRBD Service

NOTE

The following procedure uses the server names node 1 and node 2, and the cluster resource name r0. It sets up node 1 as the primary node. Make sure to modify the instructions to use your own node and file names.

1. Start YaST and select the configuration module *Miscellaneous* > *drbd*.
2. In *Start-up Configuration* > *Booting* select *On* to start drbd always at boot time.
3. If you need to configure more than one replicated resource, select *Global Configuration*. The input field *Minor Count* selects how many different drbd resources may be configured without restarting the computer.
4. The actual configuration of the resource is done in *Resource Configuration*. Press *Add* to create a new resource. The following parameters have to be set:

Resource Name	The name of the resource, often called `r0`.
Name	The hostname of the respective node.
Address:Port	The IP address and port number of the respective node.

Device	The device that holds the replicated data on the respective node. Use this device to create file systems and mount operations.
Disk	The device that is replicated between both nodes.
Meta-disk	The *Meta-disk* is either set to the value `internal` or specifies an explicit device extended by an index to hold the meta data needed by drbd. When using `internal`, the last 128 MB of the replicated device are used to store the meta data. A real device may also be used for multiple drbd resources. For example, if your *Meta-Disk* is `/dev/sda6[0]` for the first resource, you may use `/dev/sda6[1]` for the second resource. However, there must be at least 128 MB space for each resource available on this disk.

All of these options are explained in the examples in the `/usr/share/doc/packages/drbd/drbd.conf` file and in the man page of `drbd.conf(5)`.

5 Copy the `/etc/drbd.conf` file to the `/etc/drbd.conf` location on the secondary server (node 2).

```
scp /etc/drbd.conf <node 2>:/etc
```

6 Initialize and start the DRBD service on both systems by entering the following on each node:

```
drbdadm create-md r0
rcdrbd start
```

7 Configure `node1` as the primary node by entering the following on `node1`:

```
drbdsetup /dev/drbd0 primary --overwrite-data-of-peer
```

8 Check the DRBD service status by entering the following on each node:

```
rcdrbd status
```

Before proceeding, wait until the block devices on both nodes are fully synchronized. Repeat the `rcdrbd status` command to follow the synchronization progress.

9 After the block devices on both nodes are fully synchronized, format the DRBD device on the primary with a file system such as reiserfs. Any Linux file system can be used. For example, enter

```
mkfs.reiserfs -f /dev/drbd0
```

IMPORTANT

Always use the `/dev/drbd<n>` name in the command, not the actual `/dev/disk` device name.

11.3 Testing the DRBD Service

If the install and configuration procedures worked as expected, you are ready to run a basic test of the DRBD functionality. This test also helps with understanding how the software works.

1 Test the DRBD service on node 1.

1a Open a terminal console, then log in as the `root` user or equivalent.

1b Create a mount point on node 1, such as `/srv/r0mount`, by entering

```
mkdir -p /srv/r0mount
```

1c Mount the `drbd` device by entering

```
mount -o rw /dev/drbd0 /srv/r0mount
```

1d Create a file from the primary node by entering

```
touch /srv/r0mount/from_node1
```

2 Test the DRBD service on node 2.

2a Open a terminal console, then log in as the `root` user or equivalent.

2b Dismount the disk on node 1 by typing the following command on node 1:

```
umount /srv/r0mount
```

2c Downgrade the DRBD service on node 1 by typing the following command on node 1:

```
drbdadm secondary r0
```

2d On node 2, promote the DRBD service to primary by entering

```
drbdadm primary r0
```

2e On node 2, check to see if node 2 is primary by entering

```
rcdrbd status
```

2f On node 2, create a mount point such as `/srv/r0mount`, by entering

```
mkdir /srv/r0mount
```

2g On node 2, mount the DRBD device by entering

```
mount -o rw /dev/drbd0 /srv/r0mount
```

2h Verify that the file you created on node 1 is viewable by entering

```
ls /srv/r0mount
```

The `/srv/r0mount/from_node1` file should be listed.

3 If the service is working on both nodes, the DRBD setup is complete.

4 Set up node 1 as the primary again.

4a Dismount the disk on node 2 by typing the following command on node 2:

```
umount /srv/r0mount
```

4b Downgrade the DRBD service on node 2 by typing the following command on node 2:

```
drbdadm secondary r0
```

4c On node 1, promote the DRBD service to primary by entering

```
drbdadm primary r0
```

4d On node 1, check to see if node 1 is primary by entering

```
rcdrbd status
```

5 To get the service to automatically start and fail over if the server has a problem, you can set up DRBD as a high availability service with OpenAIS. For information about installing and configuring OpenAIS for SUSE Linux Enterprise 11 see Part II, “Configuration and Administration” (page 29).

11.4 Troubleshooting DRBD

The drbd setup involves many different components and problems may arise from different sources. The following sections cover several common problems and give hints to solve the issues.

11.4.1 Configuration

If the initial `drbd` setup does not work as expected, there is probably something wrong with your configuration.

To get information about the configuration:

1. Open a terminal console, then log in as the `root` user.
2. Test the configuration file by running `drbdadm` with the `-d` option. Enter

   ```
   drbdadm -d adjust r0
   ```

   In a dry run of the `adjust` option, `drbdadm` compares the actual configuration of the DRBD resource with your DRBD configuration file, but it does not execute the calls. Review the output to make sure you know the source and cause of any errors.
3. If there are errors in the `drbd.conf` file, correct them before continuing.
4. If the partitions and settings are correct, run `drbdadm` again without the `-d` option. Enter

   ```
   drbdadm adjust r0
   ```

   This applies the configuration file to the DRBD resource.

11.4.2 Hostnames

For DRBD, hostnames are case sensitive and therefore `Node0` would be a different host than `node0`.

If you have several network devices, and want to use a dedicated network device, the hostname will likely not resolve to the used ip address. In this case, use the parameter `disable-ip-verification` to make DRBD ignore this fact.

11.4.3 TCP Port 7788

If your system is unable to connect to the peer, this might be a problem of a local firewall. By default, DRBD uses the TCP port 7788 to access the other node. Make sure that this port is accessible on both nodes.

11.4.4 DRBD Devices Broken after Reboot

In cases when DRBD does not know which of the real devices holds the latest data, it changes to a split brain condition. In this case, the respective DRBD subsystems come up as secondary and do not connect to each other. In this case, the following message is written to `/var/log/messages`:

```
Split-Brain detected, dropping connection!
```

To resolve this situation, you must select one node whose modifications should be discarded. Log in to that node and run the following commands:

```
drbdadm secondary r0
drbdadm -- --discard-my-data connect r0
```

On the other node, run the command:

```
drbdadm connect r0
```

11.5 Additional Information

The following open source resources are available for DRBD:

- The following man pages for DRBD are available in the distribution:

  ```
  drbd(8)
  drbddisk(8)
  drbdsetup(8)
  drbdadm(8)
  drbd.conf(5)
  ```

- Find a commented example configuration for DRBD at `/usr/share/doc/packages/drbd/drbd.conf`

- The project home page http://www.drbd.org.
- http://clusterlabs.org/wiki/DRBD_HowTo_1.0 by the Linux Pacemaker Cluster Stack Project.

Part IV. Troubleshooting and Reference

12 Troubleshooting

Especially when starting to experiment with Heartbeat, strange problems may occur that are not easy to understand. However, there are several utilities that may be used to take a closer look at the Heartbeat internal processes. Use this chapter to get some ideas how to solve your issues.

12.1 Installation Problems

If you have difficulties installing the packages or in bringing the cluster online, proceed according to the following list.

Are the HA packages installed?
: The packages needed for configuring and managing a cluster are included in the `High Availability` installation pattern, available with the High Availability Extension.

  Check if High Availability Extension is installed as an add-on to SUSE Linux Enterprise Server 11 on each of the cluster nodes and if the *High Availability* pattern is installed on each of the machines as described in Section 3.1, “Installing the High Availability Extension” (page 23).

Is the initial configuration the same for all cluster nodes?
: In order to communicate with each other, all nodes belonging to the same cluster need to use the same `bindnetaddr`, `mcastaddr` and `mcastport` as described in Section 3.2, “Initial Cluster Setup” (page 24).

Check if the communication channels and options configured in `/etc/ais/openais.conf` are the same for all cluster nodes.

In case you use encrypted communication, check if the `/etc/ais/authkey` file is available on all cluster nodes.

Does the firewall allow communication via the `mcastport`?
If the mcastport used for communication between the cluster nodes is blocked by the firewall, the nodes cannot `see` each other. When configuring the initial setup with YaST as described in Section 3.1, “Installing the High Availability Extension” (page 23), the firewall settings are usually automatically adjusted.

To make sure the mcastport is not blocked by the firewall, check the settings in `/etc/sysconfig/SuSEfirewall2` on each node. Alternatively, start the YaST firewall module on each cluster node. After clicking *Allowed Service* > *Advanced*, add the mcastport to the list of allowed *UDP Ports* and confirm your changes.

Is OpenAIS started on each cluster node?
Check the OpenAIS status on each cluster node with `/etc/init.d/openais status`. In case OpenAIS is not running, start it by executing `/etc/init.d/openais start`.

12.2 “Debugging” a HA Cluster

If something is not working for your cluster, try the following first. It displays you the resource operation history (option `-o`) and inactive resources (`-r`):

```
crm_mon -o -r
```

The display is refreshed every 10 seconds (you can cancel it with Ctrl + C.) An example could look like:

Example 12.1 *Stopped Resources*

```
Refresh in 10s...

============
Last updated: Mon Jan 19 08:56:14 2009
Current DC: d42 (d42)
3 Nodes configured.
3 Resources configured.
============

Online: [ d230 d42 ]
OFFLINE: [ clusternode-1 ]

Full list of resources:

Clone Set: o2cb-clone
         Stopped: [  o2cb:0 o2cb:1o2cb:2 ]
Clone Set: dlm-clone
         Stopped [ dlm:0 dlm:1 dlm:2 ]
mySecondIP      (ocf::heartbeat:IPaddr):        Stopped

Operations:
* Node d230:
   aa: migration-threshold=1000000
    + (5) probe: rc=0 (ok)
    + (37) stop: rc=0 (ok)
    + (38) start: rc=0 (ok)
    + (39) monitor: interval=15000ms rc=0 (ok)
* Node d42:
   aa: migration-threshold=1000000
    + (3) probe: rc=0 (ok)
    + (12) stop: rc=0 (ok)
```

First get your node online (see Section 12.3 (page 122)). After that, check your resources and operations.

The *Configuration Explained* PDF under `http://clusterlabs.org/wiki/Documentation` explains in section *How Does the Cluster Interpret the OCF Return Codes?* three different recovery types:

12.3 FAQs

What is the state of my cluster?

To check the current state of your cluster, use the program `crm_mon`. This displays the current DC as well as all of the nodes and resources that are known to the current node.

Several nodes of my cluster do not see each other.

There could be several reasons:

- Look first in the configuration file `/etc/ais/openais.conf` and check if the multicast address is the same for every node in the cluster (look in the `interface` section with the key `mcastaddr`.)
- Check your firewall settings.
- Check if your switch supports multicast addresses
- Another reason could be, the connection between your nodes is broken. Most often, this is the result of a badly configured firewall. This also may be the reason for a *split brain* condition, where the cluster is partitioned.

I want to list my currently known resources.

Use the command `crm_resource -L` to learn about your current resources.

I configured a resource, but it always fails.

Try to run the resource agent manually. With LSB, just run *`scriptname`* `start` and *`scriptname`* `stop`. To check an OCF script, set the needed environment variables first. For example, when testing the `IPaddr` OCF script, you have to set the value for the variable `ip` by setting an environment variable that prefixes the name of the variable with `OCF_RESKEY_`. For this example, run the command:

```
export OCF_RESKEY_ip=<your_ip_address>
/usr/lib/ocf/resource.d/heartbeat/IPaddr validate-all
/usr/lib/ocf/resource.d/heartbeat/IPaddr start
/usr/lib/ocf/resource.d/heartbeat/IPaddr stop
```

If this fails, it is very likely that you missed some mandatory variable or just mistyped a parameter.

I just get a failed message. Is it possible to get more information?

You may always add the `-V` parameter to your commands. If you do that multiple times, the debug output becomes very verbose.

How can I clean up my resources?

If you know the IDs of your resources, which you can get with `crm_resource -L`, remove a specific resource with `crm_resource -C -r` *`resource id`* `-H` *`HOST`*.

I can not mount an ocfs2 device.

Check `/var/log/message` if there is the following line:

```
Jan 12 09:58:55 clusternode2 lrmd: [3487]: info: RA output:
(o2cb:1:start:stderr) 2009/01/12_09:58:55
  ERROR: Could not load ocfs2_stackglue
Jan 12 16:04:22 clusternode2 modprobe: FATAL: Module ocfs2_stackglue not
 found.
```

In this case the kernel module `ocfs2_stackglue.ko` is missing. Install the package `ocfs2-kmp-default`, `ocfs2-kmp-pae` or `ocfs2-kmp-xen` depending on the installed kernel.

12.4 Fore More Information

For additional information about high availability on Linux and Heartbeat including configuring cluster resources and managing and customizing a Heartbeat cluster, see http://clusterlabs.org/wiki/Documentation.

13 Cluster Management Tools

High Availability Extension ships with a comprehensive set of tools that assist you in managing your cluster from the command line. This chapter introduces the tools needed for managing the cluster configuration in the CIB and the cluster resources. Other command line tools for managing resource agents or tools used for debugging and troubleshooting your setup are covered in Chapter 12, *Troubleshooting* (page 119).

The following list presents several tasks related to cluster management and briefly introduces the tools to use to accomplish these tasks:

Monitoring the Cluster's Status

The `crm_mon` command allows you to monitor your cluster's status and configuration. Its output includes the number of nodes, uname, uuid, status, the resources configured in your cluster, and the current status of each. The output of `crm_mon` can be displayed at the console or printed into an HTML file. When provided with a cluster configuration file without the status section, `crm_mon` creates an overview of nodes and resources as specified in the file. See crm_mon(8) (page 149) for a detailed introduction to this tool's usage and command syntax.

Managing the CIB

The `cibadmin` command is the low-level administrative command for manipulating the Heartbeat CIB. It can be used to dump all or part of the CIB, update all or part of it, modify all or part of it, delete the entire CIB, or perform miscellaneous CIB administrative operations. See cibadmin(8) (page 128) for a detailed introduction to this tool's usage and command syntax.

Managing Configuration Changes

The `crm_diff` command assists you in creating and applying XML patches. This can be useful for visualizing the changes between two versions of the cluster configuration or saving changes so they can be applied at a later time using cibadmin(8) (page 128). See crm_diff(8) (page 140) for a detailed introduction to this tool's usage and command syntax.

Manipulating CIB Attributes

The `crm_attribute` command lets you query and manipulate node attributes and cluster configuration options that are used in the CIB. See crm_attribute(8) (page 137) for a detailed introduction to this tool's usage and command syntax.

Validating the Cluster Configuration

The `crm_verify` command checks the configuration database (CIB) for consistency and other problems. It can check a file containing the configuration or connect to a running cluster. It reports two classes of problems. Errors must be fixed before Heartbeat can work properly while warning resolution is up to the administrator. `crm_verify` assists in creating new or modified configurations. You can take a local copy of a CIB in the running cluster, edit it, validate it using `crm_verify`, then put the new configuration into effect using `cibadmin`. See crm_verify(8) (page 177) for a detailed introduction to this tool's usage and command syntax.

Managing Resource Configurations

The `crm_resource` command performs various resource-related actions on the cluster. It lets you modify the definition of configured resources, start and stop resources, or delete and migrate resources between nodes. See crm_resource(8) (page 153) for a detailed introduction to this tool's usage and command syntax.

Managing Resource Fail Counts

The `crm_failcount` command queries the number of failures per resource on a given node. This tool can also be used to reset the failcount, allowing the resource to again run on nodes where it had failed too often. See crm_failcount(8) (page 143) for a detailed introduction to this tool's usage and command syntax.

Managing a Node's Standby Status

The `crm_standby` command can manipulate a node's standby attribute. Any node in standby mode is no longer eligible to host resources and any resources that are there must be moved. Standby mode can be useful for performing maintenance tasks, such as kernel updates. Remove the standby attribute from the node as it

should become a fully active member of the cluster again. See crm_standby(8) (page 174) for a detailed introduction to this tool's usage and command syntax.

cibadmin (8)

cibadmin — Provides direct access to the cluster configuration

Synopsis

Allows the configuration, or sections of it, to be queried, modified, replaced and deleted.

```
cibadmin (--query|-Q) -[Vrwlsmfbp] [-i xml-object-id|-o
  xml-object-type] [-t t-flag-whatever] [-h hostname]

cibadmin (--create|-C) -[Vrwlsmfbp] [-X xml-string]
  [-x xml- filename] [-t t-flag-whatever] [-h hostname]

cibadmin (--replace-R)  -[Vrwlsmfbp]  [-i xml-object-id|
  -o xml-object-type]  [-X  xml-string] [-x xml-filename] [-t
  t-flag- whatever] [-h hostname]

cibadmin (--update|-U) -[Vrwlsmfbp]  [-i xml-object-id|
  -o xml-object-type]  [-X  xml-string] [-x xml-filename] [-t
  t-flag- whatever] [-h hostname]

cibadmin (--modify|-M) -[Vrwlsmfbp]  [-i xml-object-id|
           -o xml-object-type]  [-X  xml-string] [-x xml-filename] [-t
           t-flag- whatever] [-h hostname]

cibadmin (--delete|-D) -[Vrwlsmfbp]  [-i xml-object-id|
           -o xml-object-type] [-t t-flag-whatever] [-h hostname]

cibadmin (--delete_alt|-d) -[Vrwlsmfbp] -o
           xml-object-type [-X xml-string|-x xml-filename]
           [-t t-flag-whatever]  [-h hostname]

cibadmin --erase (-E)

cibadmin --bump (-B)

cibadmin --ismaster (-m)

cibadmin --master (-w)

cibadmin --slave (-r)

cibadmin --sync (-S)

cibadmin --help (-?)
```

Description

The `cibadmin` command is the low-level administrative command for manipulating the Heartbeat CIB. Use it to dump all or part of the CIB, update all or part of it, modify all or part of it, delete the entire CIB, or perform miscellaneous CIB administrative operations.

`cibadmin` operates on the XML trees of the CIB, largely without knowledge of the meaning of the updates or queries performed. This means that shortcuts that seem natural to humans who understand the meaning of the elements in the XML tree are impossible to use with `cibadmin`. It requires a complete lack of ambiguity and can only deal with valid XML subtrees (tags and elements) for both input and output.

NOTE

`cibadmin` should always be used in preference to editing the `cib.xml` file by hand—especially if the cluster is active. The cluster goes to great lengths to detect and discourage this practice so that your data is not lost or corrupted.

Options

`--obj_type` *`object-type`*, `-o` *`object-type`*
: Specify the type of object on which to operate. Valid values are `nodes`, `resources`, `constraints`, `crm_status`, and `status`.

`--verbose`, `-V`
: Turn on debug mode. Additional `-V` options increase the verbosity of the output.

`--help`, `-?`
: Obtain a help message from `cibadmin`.

`--xpath` *`PATHSPEC`*, `-A` *`PATHSPEC`*
: Supply a valid XPath to use instead of an obj_type.

Commands

`--bump`, `-B`
: Increase the `epoch` version counter in the CIB. Normally this value is increased automatically by the cluster when a new leader is elected. Manually increasing it can be useful if you want to make an older configuration obsolete (such as one stored on inactive cluster nodes).

`--create`, `-C`
: Create a new CIB from the XML content of the argument.

`--delete`, `-D`
: Delete the first object matching the supplied criteria, for example, `<op id="rsc1_op1" name="monitor"/>`. The tag name and all attributes must match in order for the element to be deleted

`--erase`, `-E`
: Erase the contents of the entire CIB.

`--ismaster`, `-m`
: Print a message indicating whether or not the local instance of the CIB software is the master instance or not. Exits with return code 0 if it is the master instance or 35 if not.

`--modify`, `-M`
: Find the object somewhere in the CIB's XML tree and update it.

`--query`, `-Q`
: Query a portion of the CIB.

`--replace`, `-R`
: Recursively replace an XML object in the CIB.

`--sync`, `-S`
: Force a resync of all nodes with the CIB on the specified host (if `-h` is used) or with the DC (if no `-h` option is used).

XML Data

`--xml-text` *`string`*, `-X` *`string`*
: Specify an XML tag or fragment on which crmadmin should operate. It must be a complete tag or XML fragment.

`--xml-file` *`filename`*, `-x` *`filename`*
: Specify XML from a file on which cibadmin should operate. It must be a complete tag or XML fragment.

`--xml_pipe`, `-p`
: Specify that the XML on which `cibadmin` should operate comes from standard input. It must be a complete tag or XML fragment.

Advanced Options

`--host` *`hostname`*, `-h` *`hostname`*
: Send command to specified host. Applies to `query` and `sync` commands only.

`--local`, `-l`
: Let a command take effect locally (rarely used, advanced option).

`--no-bcast`, `-b`
: Command will not be broadcast even if it altered the CIB.

IMPORTANT

Use this option with care to avoid ending up with a divergent cluster.

`--sync-call`, `-s`
: Wait for call to complete before returning.

Examples

To get a copy of the entire active CIB (including status section, etc.) delivered to stdout, issue this command:

```
cibadmin -Q
```

To add an IPaddr2 resource to the *resources* section, first create a file `foo` with the following contents:

```
<primitive id="R_10.10.10.101" class="ocf" type="IPaddr2"
 provider="heartbeat">
 <instance_attributes id="RA_R_10.10.10.101">
  <attributes>
   <nvpair id="R_ip_P_ip" name="ip" value="10.10.10.101"/>
   <nvpair id="R_ip_P_nic" name="nic" value="eth0"/>
  </attributes>
 </instance_attributes>
</primitive>
```

Then issue the following command:

```
cibadmin --obj_type resources -U -x foo
```

To change the IP address of the IPaddr2 resource previously added, issue the command below:

```
cibadmin -M -X '<nvpair id="R_ip_P_ip" name="ip" value="10.10.10.102"/>'
```

NOTE

This does not change the resource name to match the new IP address. To do that, delete then re-add the resource with a new ID tag.

To stop (disable) the IP address resource added previously without removing it, create a file called `bar` with the following content in it:

```
<primitive id="R_10.10.10.101">
 <instance_attributes id="RA_R_10.10.10.101">
  <attributes>
   <nvpair id="stop_R_10.10.10.101" name="target-role" value="Stopped"/>
  </attributes>
 </instance_attributes>
</primitive>
```

Then issue the following command:

```
cibadmin --obj_type resources -U -x bar
```

To restart the IP address resource stopped by the previous step, issue:

```
cibadmin -D -X '<nvpair id="stop_R_10.10.10.101">'
```

To completely remove the IP address resource from the CIB, issue this command:

```
cibadmin -D -X '<primitive id="R_10.10.10.101"/>'
```

To replace the CIB with a new manually-edited version of the CIB, use the following command:

```
cibadmin -R -x $HOME/cib.xml
```

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk.

See Also

crm_resource(8) (page 153), crmadmin(8) (page 134), lrmadmin(8), heartbeat(8)

Author

`cibadmin` was written by Andrew Beekhof.

This manual page was originally written by Alan Robertson.

Caveats

Avoid working on the automatically maintained copy of the CIB on the local disk. Whenever anything in the cluster changes, the CIB is updated. Therefore using an outdated backup copy of the CIB to propagate your configuration changes might result in an inconsistent cluster.

crmadmin (8)

crmadmin — controls the Cluster Resource Manager

Synopsis

```
crmadmin [-V|-q] [-i|-d|-K|-S|-E] node
crmadmin [-V|-q] -N -B
crmadmin [-V|-q] -D
crmadmin -v
crmadmin -?
```

Description

`crmadmin` was originally designed to control most of the actions of the CRM daemon. However, the largest part of its functionality has been made obsolete by other tools, such as `crm_attribute` and `crm_resource`. Its remaining functionality is mostly related to testing and the status of the crmd process.

WARNING

Some `crmadmin` options are geared towards testing and cause trouble if used incorrectly. In particular, do not use the `--kill` or `--election` options unless you know exactly what you are doing.

Options

`--help`, `-?`
: Print the help text.

`--version`, `-v`
: Print version details for HA, CRM, and CIB feature set.

`--verbose`, `-V`
: Turn on command debug information.

NOTE

Increase the level of verbosity by providing additional instances.

`--quiet`, `-q`
: Do not provide any debug information at all and reduce the output to a minimum.

`--bash-export`, `-B`
: Create bash export entries of the form `export uname=uuid`. This applies only to the `crmadmin -N node` command.

NOTE

The `-B` functionality is rarely useful and may be removed in future versions.

Commands

`--debug_inc node`, `-i node`
: Increment the CRM daemon's debug level on the specified node. This can also be achieved by sending the USR1 signal to the crmd process.

`--debug_dec node`, `-d node`
: Decrement the CRM daemon's debug level on the specified node. This can also be achieved by sending the USR2 signal to the crmd process.

`--kill node`, `-K node`
: Shut down the CRM daemon on the specified node.

WARNING

Use this with extreme caution. This action should normally only be issued by Heartbeat and may have unintended side effects.

`--status node`, `-S node`
: Query the status of the CRM daemon on the specified node.

  The output includes a general heath indicator and the internal FSM state of the crmd process. This can be helpful when determining what the cluster is doing.

`--election` *node*, `-E` *node*
Initiate an election from the specified node.

WARNING

Use this with extreme caution. This action is normally initiated internally and may have unintended side effects.

`--dc_lookup`, `-D`
Query the uname of the current DC.

The location of the DC is only of significance to the crmd internally and is rarely useful to administrators except when deciding on which node to examine the logs.

`--nodes`, `-N`
Query the uname of all member nodes. The results of this query may include nodes in `offline` mode.

NOTE

The `-i`, `-d`, `-K`, and `-E` options are rarely used and may be removed in future versions.

See Also

crm_attribute(8) (page 137), crm_resource(8) (page 153)

Author

`crmadmin` was written by Andrew Beekhof.

crm_attribute (8)

crm_attribute — Allows node attributes and cluster options to be queried, modified and deleted

Synopsis

```
crm_attribute [options]
```

Description

The `crm_attribute` command queries and manipulates node attributes and cluster configuration options that are used in the CIB.

Options

`--help`, `-?`
Print a help message.

`--verbose`, `-V`
Turn on debug information.

> **NOTE**
>
> Increase the level of verbosity by providing additional instances.

`--quiet`, `-Q`
When doing an attribute query using `-G`, print just the value to stdout. Use this option with `-G`.

`--get-value`, `-G`
Retrieve rather than set the preference.

`--delete-attr`, `-D`
Delete rather than set the attribute.

`--attr-id` *string*, `-i` *string*
For advanced users only. Identifies the id attribute.

`--attr-value` *string*, `-v` *string*
Value to set. This is ignored when used with `-G`.

`--node` *node_name*, `-N` *node_name*
The uname of the node to change

`--set-name` *string*, `-s` *string*
Specify the set of attributes in which to read or write the attribute.

`--attr-name` *string*, `-n` *string*
Specify the attribute to set or query.

`--type` *string*, `-t` *type*
Determine to which section of the CIB the attribute should be set or to which section of the CIB the attribute that is queried belongs Possible values are `nodes`, `status`, or `crm_config`.

Examples

Query the value of the `location` attribute in the `nodes` section for the host *myhost* in the CIB:

```
crm_attribute -G -t nodes -U myhost -n location
```

Query the value of the `cluster-delay` attribute in the `crm_config` section in the CIB:

```
crm_attribute -G -t crm_config -n cluster-delay
```

Query the value of the `cluster-delay` attribute in the `crm_config` section in the CIB. Print just the value:

```
crm_attribute -G -Q -t crm_config -n cluster-delay
```

Delete the `location` attribute for the host *myhost* from the `nodes` section of the CIB:

```
crm_attribute -D -t nodes -U myhost -n location
```

Add a new attribute called `location` with the value of `office` to the *`set`* subsection of the `nodes` section in the CIB (settings applied to the host *`myhost`*):

```
crm_attribute -t nodes -U myhost -s set -n location -v office
```

Change the value of the `location` attribute in the `nodes` section for the *`myhost`* host:

```
crm_attribute -t nodes -U myhost -n location -v backoffice
```

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk. Editing this file directly is strongly discouraged.

See Also

cibadmin(8) (page 128)

Author

`crm_attribute` was written by Andrew Beekhof.

crm_diff (8)

crm_diff — identify changes to the cluster configuration and apply patches to the configuration files

Synopsis

```
crm_diff [-?|-V] [-o filename] [-O string] [-p filename] [-n filename] [-N string]
```

Description

The `crm_diff` command assists in creating and applying XML patches. This can be useful for visualizing the changes between two versions of the cluster configuration or saving changes so they can be applied at a later time using `cibadmin`.

Options

`--help`, `-?`
Print a help message.

`--original` *filename*, `-o` *filename*
Specify the original file against which to diff or apply patches.

`--new` *filename*, `-n` *filename*
Specify the name of the new file.

`--original-string` *string*, `-O` *string*
Specify the original string against which to diff or apply patches.

`--new-string` *string*, `-N` *string*
Specify the new string.

`--patch` *filename*, `-p` *filename*
Apply a patch to the original XML. Always use with `-o`.

`--cib, -c`
: Compare or patch the inputs as a CIB. Always specify the base version with `-o` and provide either the patch file or the second version with `-p` or `-n`, respectively.

`--stdin, -s`
: Read the inputs from stdin.

Examples

Use crm_diff to determine the differences between various CIB configuration files and to create patches. By means of patches, easily reuse configuration parts without having to use the cibadmin command on every single one of them.

1. Obtain the two different configuration files by running cibadmin on the two cluster setups to compare:

   ```
   cibadmin -Q > cib1.xml
   cibadmin -Q > cib2.xml
   ```

2. Determine whether to diff the entire files against each other or compare just a subset of the configurations.

3. To print the difference between the files to stdout, use the following command:

   ```
   crm_diff -o cib1.xml -n cib2.xml
   ```

4. To print the difference between the files to a file and create a patch, use the following command:

   ```
   crm_diff -o cib1.xml -n cib2.xml > patch.xml
   ```

5. Apply the patch to the original file:

   ```
   crm_diff -o cib1.xml -p patch.xml
   ```

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk. Editing this file directly is strongly discouraged.

See Also

cibadmin(8) (page 128)

Author

`crm_diff` was written by Andrew Beekhof.

crm_failcount (8)

crm_failcount — Manage the counter recording each resource's failures

Synopsis

```
crm_failcount [-?|-V] -D -u|-U node -r resource
crm_failcount [-?|-V] -G -u|-U node -r resource
crm_failcount [-?|-V] -v string -u|-U node -r resource
```

Description

Heartbeat implements a sophisticated method to compute and force failover of a resource to another node in case that resource tends to fail on the current node. A resource carries a `resource-stickiness` attribute to determine how much it prefers to run on a certain node. It also carries a `migration-threshold` that determines the threshold at which the resource should failover to another node.

The `failcount` attribute is added to the resource and increased on resource monitoring failure. The value of `failcount` multiplied by the value of `migration-threshold` determines the *failover score* of this resource. If this number exceeds the preference set for this resource, the resource is moved to another node and not run again on the original node until the failure count is reset.

The `crm_failcount` command queries the number of failures per resource on a given node. This tool can also be used to reset the failcount, allowing the resource to run again on nodes where it had failed too often.

Options

`--help`, `-?`
: Print a help message.

`--verbose`, `-V`
: Turn on debug information.

NOTE

Increase the level of verbosity by providing additional instances.

`--quiet`, `-Q`
When doing an attribute query using `-G`, print just the value to stdout. Use this option with `-G`.

`--get-value`, `-G`
Retrieve rather than set the preference.

`--delete-attr`, `-D`
Specify the attribute to delete.

`--attr-id` *`string`*, `-i` *`string`*
For advanced users only. Identifies the id attribute.

`--attr-value` *`string`*, `-v` *`string`*
Specify the value to use. This option is ignored when used with `-G`.

`--node` *`node_uname`*, `-U` *`node_uname`*
Specify the uname of the node to change.

`--resource-id` *`resource name`*, `-r` *`resource name`*
Specify the name of the resource on which to operate.

Examples

Reset the failcount for the resource `myrsc` on the node `node1`:

```
crm_failcount -D -U node1 -r my_rsc
```

Query the current failcount for the resource `myrsc` on the node `node1`:

```
crm_failcount -G -U node1 -r my_rsc
```

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk. Editing this file directly is strongly discouraged.

See Also

crm_attribute(8) (page 137), cibadmin(8) (page 128), and the Linux High Availability FAQ Web site [`http://www.linux-ha.org/v2/faq/forced_failover`]

Author

`crm_failcount` was written by Andrew Beekhof.

crm_master (8)

crm_master — Manage a master/slave resource's preference for being promoted on a given node

Synopsis

```
crm_master [-V|-Q] -D [-l lifetime]
crm_master [-V|-Q] -G [-l lifetime]
crm_master [-V|-Q] -v string [-l string]
```

Description

`crm_master` is called from inside the resource agent scripts to determine which resource instance should be promoted to master mode. It should never be used from the command line and is just a helper utility for the resource agents. RAs use `crm_master` to promote a particular instance to master mode or to remove this preference from it. By assigning a lifetime, determine whether this setting should survive a reboot of the node (set lifetime to `forever`) or whether it should not survive a reboot (set lifetime to `reboot`).

A resource agent needs to determine on which resource `crm_master` should operate. These queries must be handled inside the resource agent script. The actual calls of `crm_master` follow a syntax similar to those of the `crm_attribute` command.

Options

`--help`, `-?`
: Print a help message.

`--verbose`, `-V`
: Turn on debug information.

NOTE

Increase the level of verbosity by providing additional instances.

`--quiet`, `-Q`
When doing an attribute query using `-G`, print just the value to stdout. Use this option with `-G`.

`--get-value`, `-G`
Retrieve rather than set the preference to be promoted.

`--delete-attr`, `-D`
Delete rather than set the attribute.

`--attr-id` *string*, `-i` *string*
For advanced users only. Identifies the id attribute.

`--attr-value` *string*, `-v` *string*
Value to set. This is ignored when used with `-G`.

`--lifetime` *string*, `-l` *string*
Specify how long the preference lasts. Possible values are `reboot` or `forever`.

Environment Variables

`OCF_RESOURCE_INSTANCE`—the name of the resource instance

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk.

See Also

cibadmin(8) (page 128), crm_attribute(8) (page 137)

Author

crm_master was written by Andrew Beekhof.

crm_mon (8)

crm_mon — monitor the cluster's status

Synopsis

```
crm_mon [-V] -d -pfilename -h filename
crm_mon [-V] [-1|-n|-r] -h filename
crm_mon [-V] [-n|-r] -X filename
crm_mon [-V] [-n|-r] -c|-1
crm_mon [-V] -i interval
crm_mon -?
```

Description

The `crm_mon` command allows you to monitor your cluster's status and configuration. Its output includes the number of nodes, uname, uuid, status, the resources configured in your cluster, and the current status of each. The output of `crm_mon` can be displayed at the console or printed into an HTML file. When provided with a cluster configuration file without the status section, `crm_mon` creates an overview of nodes and resources as specified in the file.

Options

`--help`, `-?`
: Provide help.

`--verbose`, `-V`
: Increase the debug output.

`--interval` *seconds*, `-i` *seconds*
: Determine the update frequency. If `-i` is not specified, the default of 15 seconds is assumed.

`--group-by-node`, `-n`
Group resources by node.

`--inactive`, `-r`
Display inactive resources.

`--simple-status`, `-s`
Display the cluster status once as a simple one line output (suitable for nagios).

`--one-shot`, `-1`
Display the cluster status once on the console then exit (does not use ncurses).

`--as-html` *filename*, `-h` *filename*
Write the cluster's status to the specified file.

`--web-cgi`, `-w`
Web mode with output suitable for CGI.

`--daemonize`, `-d`
Run in the background as a daemon.

`--pid-file` *filename*, `-p` *filename*
Specify the daemon's pid file.

Examples

Display your cluster's status and get an updated listing every 15 seconds:

```
crm_mon
```

Display your cluster's status and get an updated listing after an interval specified by `-i`. If `-i` is not given, the default refresh interval of 15 seconds is assumed:

```
crm_mon -i interval[s]
```

Display your cluster's status on the console:

```
crm_mon -c
```

Display your cluster's status on the console just once then exit:

```
crm_mon -1
```

Display your cluster's status and group resources by node:

```
crm_mon -n
```

Display your cluster's status, group resources by node, and include inactive resources in the list:

```
crm_mon -n -r
```

Write your cluster's status to an HTML file:

```
crm_mon -h filename
```

Run `crm_mon` as a daemon in the background, specify the daemon's pid file for easier control of the daemon process, and create HTML output. This option allows you to constantly create HTML output that can be easily processed by other monitoring applications:

```
crm_mon -d -p filename -h filename
```

Display the cluster configuration laid out in an existing cluster configuration file (*`filename`*), group the resources by node, and include inactive resources. This command can be used for dry-runs of a cluster configuration before rolling it out to a live cluster.

```
crm_mon -r -n -X filename
```

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk. Editing this file directly is strongly discouraged.

Author

`crm_mon` was written by Andrew Beekhof.

crm_node (8)

crm_node — Lists the members of a cluster

Synopsis

```
crm_node [-V] [-p|-e|-q]
```

Description

Lists the members of a cluster.

Options

`-V`
: be verbose

`--partition`, `-p`
: print the members of this partition

`--epoch`, `-e`
: print the epoch this node joined the partition

`--quorum`, `-q`
: print a 1 if our partiton has quorum

crm_resource (8)

crm_resource — Perform tasks related to cluster resources

Synopsis

```
crm_resource  [-?|-V|-S] -L|-Q|-W|-D|-C|-P|-p [options]
```

Description

The `crm_resource` command performs various resource-related actions on the cluster. It can modify the definition of configured resources, start and stop resources, and delete and migrate resources between nodes.

`--help`, `-?`
Print the help message.

`--verbose`, `-V`
Turn on debug information.

NOTE

Increase the level of verbosity by providing additional instances.

`--quiet`, `-Q`
Print only the value on stdout (for use with `-W`).

Commands

`--list`, `-L`
List all resources.

`--query-xml`, `-x`
Query a resource.

Requires: `-r`

`--locate`, `-W`
Locate a resource.

Requires: `-r`

`--migrate`, `-M`
Migrate a resource from its current location. Use `-N` to specify a destination.

If `-N` is not specified, the resource is forced to move by creating a rule for the current location and a score of `-INFINITY`.

NOTE

This prevents the resource from running on this node until the constraint is removed with `-U`.

Requires: `-r`, Optional: `-N`, `-f`

`--un-migrate`, `-U`
Remove all constraints created by `-M`

Requires: `-r`

`--delete`, `-D`
Delete a resource from the CIB.

Requires: `-r`, `-t`

`--cleanup`, `-C`
Delete a resource from the LRM.

Requires: `-r`. Optional: `-H`

`--reprobe`, `-P`
Recheck for resources started outside the CRM.

Optional: `-H`

`--refresh`, `-R`
Refresh the CIB from the LRM.

Optional: `-H`

`--set-parameter` *`string`*, `-p` *`string`*
Set the named parameter for a resource.

Requires: `-r`, `-v`. Optional: `-i`, `-s`, and `--meta`

`--get-parameter` *`string`*, `-g` *`string`*
Get the named parameter for a resource.

Requires: `-r`. Optional: `-i`, `-s`, and `--meta`

`--delete-parameter` *`string`*, `-d` *`string`*
Delete the named parameter for a resource.

Requires: `-r`. Optional: `-i`, and `--meta`

`--list-operations` *`string`* , `-O` *`string`*
List the active resource operations. Optionally filtered by resource, node, or both.
Optional: `-N`, `-r`

`--list-all-operations` *`string`* , `-o` *`string`*
List all resource operations. Optionally filtered by resource, node, or both. Optional: `-N`, `-r`

Options

`--resource` *`string`*, `-r` *`string`*
Specify the resource ID.

`--resource-type` *`string`*, `-t` *`string`*
Specify the resource type (`primitive`, `clone`, `group`, etc.).

`--property-value` *`string`*, `-v` *`string`*
Specify the property value.

`--node` *`string`*, `-N` *`string`*
Specify the hostname.

`--meta`
Modify a resource's configuration option rahter than one which is passed to the resouce agent script. For use with `-p`, `-g` and `-d`.

`--lifetime` *`string`*, `-u` *`string`*
Lifespan of migration constraints.

`--force`, `-f`
Force the resource to move by creating a rule for the current location and a score of `-INFINITY`

This should be used if the resource's stickiness and constraint scores total more than `INFINITY` (currently 100,000).

NOTE

This prevents the resource from running on this node until the constraint is removed with `-U`.

`-s` *`string`*
(Advanced Use Only) Specify the ID of the `instance_attributes` object to change.

`-i` *`string`*
(Advanced Use Only) Specify the ID of the `nvpair` object to change or delete.

Examples

Listing all resources:

```
crm_resource -L
```

Checking where a resource is running (and if it is):

```
crm_resource -W -r my_first_ip
```

If the `my_first_ip` resource is running, the output of this command reveals the node on which it is running. If it is not running, the output shows this.

Start or stop a resource:

```
crm_resource -r my_first_ip -p target_role -v started
crm_resource -r my_first_ip -p target_role -v stopped
```

Query the definition of a resource:

```
crm_resource -Q -r my_first_ip
```

Migrate a resource away from its current location:

```
crm_resource -M -r my_first_ip
```

Migrate a resource to a specific location:

```
crm_resource -M -r my_first_ip -H c001n02
```

Allow a resource to return to its normal location:

```
crm_resource -U -r my_first_ip
```

NOTE

The values of `resource_stickiness` and `default_resource_stickiness` may mean that it does not move back. In such cases, you should use `-M` to move it back before running this command.

Delete a resource from the CRM:

```
crm_resource -D -r my_first_ip -t primitive
```

Delete a resource group from the CRM:

```
crm_resource -D -r my_first_group -t group
```

Disable resource management for a resource in the CRM:

```
crm_resource -p is-managed -r my_first_ip -t primitive -v off
```

Enable resource management for a resource in the CRM:

```
crm_resource -p is-managed -r my_first_ip -t primitive -v on
```

Reset a failed resource after having been manually cleaned up:

```
crm_resource -C -H c001n02 -r my_first_ip
```

Recheck all nodes for resources started outside the CRM:

```
crm_resource -P
```

Recheck one node for resources started outside the CRM:

```
crm_resource -P -H c001n02
```

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk. Editing this file directly is strongly discouraged.

See Also

cibadmin(8) (page 128), crmadmin(8) (page 134), lrmadmin(8), heartbeat(8)

Author

`crm_resource` was written by Andrew Beekhof.

crm_shadow (8)

crm_shadow — Perform Configuration Changes in a Sandbox Before Updating The Live Cluster

Synopsis

```
crm_shadow [-V] [-p|-e|-q]
```

Description

Sets up an environment in which configuration tools (`cibadmin`, `crm_resource`, etc) work offline instead of against a live cluster, allowing changes to be previewed and tested for side-effects.

Options

`--verbose`, `-V`
: turn on debug info. additional instance increase verbosity

`--which`, `-w`
: indicate the active shadow copy

`--display`, `-p`
: display the contents of the shadow copy

`--diff`, `-d`
: display the changes in the shadow copy

`--create-empty`, `-e`*NAME*
: create the named shadow copy with an empty cluster configuration

`--create`, `-c`*NAME*
: create the named shadow copy of the active cluster configuration

`--reset`, `-r`*NAME*
: recreate the named shadow copy from the active cluster configuration

`--commit`, `-c`*NAME*
: upload the contents of the named shadow copy to the cluster

`--delete`, `-d`*NAME*
: delete the contents of the named shadow copy

`--edit`, `-e`*NAME*
: Edit the contents of the named shadow copy with your favorite editor

`--batch`, `-b`
: do not spawn a new shell

`--force`, `-f`
: do not spawn a new shell

`--switch`, `-s`
: switch to the named shadow copy

Internal Commands

To work with a shadow configuration, you need to create one first:

```
crm_shadow --create-empty YOUR_NAME
```

It gives you an internal shell like the one from the `crm` tool. Use `help` to get an overview of all internal commands, or `help` *`subcommand`* for a specific command.

Table 13.1 *Overview of Internal Commands*

Command	Syntax/Description
`alias`	`alias [-p] [name[=value] ... ]` `alias` with no arguments or with the -p option prints the list of aliases in the form alias `NAME=VALUE` on standard output. Otherwise, an alias is defined for each NAME whose VALUE is given. A trailing space in VALUE causes the next word to be checked for alias substi-

Command	Syntax/Description
	tution when the alias is expanded. Alias returns true unless a NAME is given for which no alias has been defined.
`bg`	`bg [JOB_SPEC ...]` Place each JOB_SPEC in the background, as if it had been started with &. If JOB_SPEC is not present, the shell's notion of the current job is used.

`bind`	`bind [-lpvsPVS] [-m keymap] [-f filename]`
	`[-q name] [-u name] [-r keyseq]`
	`[-x keyseq:shell-command]`
	`[keyseq:readline-function or readline-command]`
	
	Bind a key sequence to a Readline function or a macro, or set a Readline variable. The non-option argument syntax is equivalent to that found in `~/.inputrc`, but must be passed as a single argument: bind `"\C-x\C-r": re-read-init-file`.

`break`	`break [N]` Exit from within a `for`, `while` or `until` loop. If N is specified, break N levels.
`builtin`	`builtin [shell-builtin [arg ...]]` Run a shell builtin. This is useful when you wish to rename a shell builtin to be a function, but need the functionality of the builtin within the function itself.
`caller`	`caller [EXPR]` Returns the context of the current subroutine call. Without *EXPR*, returns `$line $filename`. With *EXPR*, returns `$line $subroutine $filename`; this extra information can be used to provide a stack trace.
`case`	`case WORD in [PATTERN [| PATTERN] [COMMANDS;;] ... esac`

Command	Syntax/Description
	Selectively execute *COMMANDS* based upon *WORD* matching *PATTERN*. The `|' is used to separate multiple patterns.
cd	`cd [-L|-P] [dir]` Change the current directory to DIR.

command	`command [-pVv]`
	`command [arg ...]`
	Runs *COMMAND* with *ARGS* ignoring shell functions. If you have a shell function called `ls', and you wish to call the command `ls', you can say "command ls". If the -p option is given, a default value is used for PATH that is guaranteed to find all of the standard utilities. If the -V or -v option is given, a string is printed describing COMMAND. The -V option produces a more verbose description.

compgen	`compgen [-abcdefgjksuv] [-o option] [-A action]`
	`[-G globpat] [-W wordlist] [-P prefix]`
	`[-S suffix] [-X filterpat] [-F function]`
	`[-C command] [WORD]`
	Display the possible completions depending on the options. Intended to be used from within a shell function generating possible completions. If the optional *WORD* argument is supplied, matches against *WORD* are generated.

complete	`complete [-abcdefgjksuv] [-pr] [-o option]`
	`[-A action] [-G globpat] [-W wordlist] [-P prefix]`
	`[-S suffix] [-X filterpat] [-F function] [-C command]`
	`[name ...]`
	For each *NAME*, specify how arguments are to be completed. If the `-p` option is supplied, or if no options are supplied, existing completion specifications are printed in a way that allows them to be reused as input. The `-r` option removes a completion specification for each *NAME*, or, if no *NAME*s are supplied, all completion specifications.

continue	`continue [N]`

Command	Syntax/Description
	Resume the next iteration of the enclosing FOR, WHILE or UNTIL loop. If *N* is specified, resume at the *N*-th enclosing loop.
`declare`	`declare [-afFirtx] [-p] [name[=value] ...]` Declare variables and/or give them attributes. If no *NAME*s are given, then display the values of variables instead. The `-p` option will display the attributes and values of each *NAME*.
`dirs`	`dirs [-clpv] [+N] [-N]` Display the list of currently remembered directories. Directories find their way onto the list with the `pushd` command; you can get back up through the list with the `popd` command.
`disown`	`disown [-h] [-ar] [JOBSPEC ...]` By default, removes each *JOBSPEC* argument from the table of active jobs. If the `-h` option is given, the job is not removed from the table, but is marked so that SIGHUP is not sent to the job if the shell receives a SIGHUP. The `-a` option, when *JOBSPEC* is not supplied, means to remove all jobs from the job table; the `-r` option means to remove only running jobs.
`echo`	`echo [-neE] [arg ...]` Output the ARGs. If -n is specified, the trailing newline is suppressed. If the -e option is given, interpretation of the following backslash-escaped characters is turned on: `\a` (alert, bell) `\b` (backspace) `\c` (suppress trailing newline) `\E` (escape character) `\f` (form feed) `\n` (new line)

Command	Syntax/Description
	\r (carriage return) \t (horizontal tab) \v (vertical tab) \\ (backslash) \0nnn (the character whose ASCII code is NNN (octal). NNN can be 0 to 3 octal digits) You can explicitly turn off the interpretation of the above characters with the -E option.
enable	`enable [-pnds] [-a] [-f filename] [name...]` Enable and disable builtin shell commands. This allows you to use a disk command which has the same name as a shell builtin without specifying a full pathname. If -n is used, the *NAME*s become disabled; otherwise *NAME*s are enabled. For example, to use the test found in $PATH instead of the shell builtin version, type enable -n test. On systems supporting dynamic loading, the -f option may be used to load new builtins from the shared object *FILENAME*. The -d option will delete a builtin previously loaded with -f. If no non-option names are given, or the -p option is supplied, a list of builtins is printed. The -a option means to print every builtin with an indication of whether or not it is enabled. The -s option restricts the output to the POSIX.2 `special' builtins. The -n option displays a list of all disabled builtins.
eval	`eval [ARG ...]` Read *ARG*s as input to the shell and execute the resulting command(s).
exec	`exec [-cl] [-a name] file [redirection ...]` Exec *FILE*, replacing this shell with the specified program. If *FILE* is not specified, the redirections take effect in this shell. If the first argument is -l, then place a dash in the zeroth arg passed to *FILE*, as login does. If the -c option is supplied, FILE is executed with a null environment. The -a option means to make set argv[0] of the executed

Command	Syntax/Description
	process to *NAME*. If the file cannot be executed and the shell is not interactive, then the shell exits, unless the shell option `execfail` is set.
`exit`	`exit [N]` Exit the shell with a status of *N*. If *N* is omitted, the exit status is that of the last command executed.

`export`	`export [-nf] [NAME[=value] ...]`
	`export -p`
	
	*NAME*s are marked for automatic export to the environment of subsequently executed commands. If the `-f` option is given, the *NAME*s refer to functions. If no *NAME*s are given, or if `-p` is given, a list of all names that are exported in this shell is printed. An argument of `-n` says to remove the export property from subsequent *NAME*s. An argument of `--` disables further option processing.

`false`	`false` Return an unsuccessful result.

`fc`	`fc [-e ename] [-nlr] [FIRST] [LAST]`
	`fc -s [pat=rep] [cmd]`
	
	fc is used to list or edit and re-execute commands from the history list. *FIRST* and *LAST* can be numbers specifying the range, or *FIRST* can be a string, which means the most recent command beginning with that string.

`fg`	`fg [JOB_SPEC]` Place *JOB_SPEC* in the foreground, and make it the current job. If *JOB_SPEC* is not present, the shell's notion of the current job is used.
`for`	`for NAME [in WORDS ... ;] do COMMANDS; done`

Command	Syntax/Description
	The `for` loop executes a sequence of commands for each member in a list of items. If `in WORDS ...;` is not present, then `in "$@"` is assumed. For each element in *WORDS*, *NAME* is set to that element, and the *COMMANDS* are executed.

`function`	`function NAME { COMMANDS ; }`
	`function NAME () { COMMANDS ; }`
	
	Create a simple command invoked by *NAME* which runs *COMMANDS*. Arguments on the command line along with *NAME* are passed to the function as `$0` .. `$n`.

`getopts`	`getopts OPTSTRING NAME [arg]` Getopts is used by shell procedures to parse positional parameters.
`hash`	`hash [-lr] [-p PATHNAME] [-dt] [NAME...]` For each *NAME*, the full pathname of the command is determined and remembered. If the `-p` option is supplied, *PATHNAME* is used as the full pathname of *NAME*, and no path search is performed. The `-r` option causes the shell to forget all remembered locations. The `-d` option causes the shell to forget the remembered location of each *NAME*. If the `-t` option is supplied the full pathname to which each *NAME* corresponds is printed. If multiple *NAME* arguments are supplied with `-t`, the *NAME* is printed before the hashed full pathname. The `-l` option causes output to be displayed in a format that may be reused as input. If no arguments are given, information about remembered commands is displayed.

`history`	`history [-c] [-d OFFSET] [n]`
	`history -ps arg [arg...]`
	`history -awrm [filename]`
	
	Display the history list with line numbers. Lines listed with with a `*` have been modified. Argument of *N* says to list only the last *N* lines. The `-c` option causes the history list to be cleared by deleting all of

Command	Syntax/Description
	the entries. The `-d` option deletes the history entry at offset *OFFSET*. The `-w` option writes out the current history to the history file; `-r` means to read the file and append the contents to the history list instead. `-a` means to append history lines from this session to the history file. Argument `-n` means to read all history lines not already read from the history file and append them to the history list.

`jobs`	`jobs [-lnprs] [JOBSPEC ...]`
	`job -x COMMAND [ARGS]`
	
	Lists the active jobs. The `-l` option lists process id's in addition to the normal information; the `-p` option lists process id's only. If `-n` is given, only processes that have changed status since the last notification are printed. *JOBSPEC* restricts output to that job. The `-r` and `-s` options restrict output to running and stopped jobs only, respectively. Without options, the status of all active jobs is printed. If `-x` is given, *COMMAND* is run after all job specifications that appear in *ARGS* have been replaced with the process ID of that job's process group leader.

`kill`	`kill [-s sigspec | -n signum | -sigspec] pid | JOBSPEC ...`
	`kill -l [sigspec]`
	
	Send the processes named by PID (or *JOBSPEC*) the signal SIGSPEC. If SIGSPEC is not present, then SIGTERM is assumed. An argument of `-l` lists the signal names; if arguments follow `-l` they are assumed to be signal numbers for which names should be listed. Kill is a shell builtin for two reasons: it allows job IDs to be used instead of process IDs, and, if you have reached the limit on processes that you can create, you don't have to start a process to kill another one.

`let`	`let ARG [ARG ...]` Each *ARG* is an arithmetic expression to be evaluated. Evaluation is done in fixed-width integers with no check for overflow, though division by 0 is trapped and flagged as an error. The following list of operators is grouped into levels of equal-precedence operators. The levels are listed in order of decreasing precedence.

Command	Syntax/Description
`local`	`local NAME[=VALUE] ...` Create a local variable called *NAME*, and give it *VALUE*. `local` can only be used within a function; it makes the variable *NAME* have a visible scope restricted to that function and its children.
`logout`	`logout` Logout of a login shell.
`popd`	`popd [+N | -N] [-n]` Removes entries from the directory stack. With no arguments, removes the top directory from the stack, and cd's to the new top directory.
`printf`	`printf [-v var] format [ARGUMENTS]` printf formats and prints *ARGUMENTS* under control of the *FORMAT*. *FORMAT* is a character string which contains three types of objects: plain characters, which are simply copied to standard output, character escape sequences which are converted and copied to the standard output, and format specifications, each of which causes printing of the next successive argument. In addition to the standard printf(1) formats, `%b` means to expand backslash escape sequences in the corresponding argument, and `%q` means to quote the argument in a way that can be reused as shell input. If the `-v` option is supplied, the output is placed into the value of the shell variable VAR rather than being sent to the standard output.
`pushd`	`pushd [dir | +N | -N] [-n]` Adds a directory to the top of the directory stack, or rotates the stack, making the new top of the stack the current working directory. With no arguments, exchanges the top two directories.
`pwd`	`pwd [-LP]`

Command	Syntax/Description
	Print the current working directory. With the `-P` option, `pwd` prints the physical directory, without any symbolic links; the `-L` option makes `pwd` follow symbolic links.
`read`	`read [-ers] [-u fd] [-t timeout] [-p prompt] [-a array] [-n nchars] [-d delim] [NAME ...]` The given *NAME*s are marked readonly and the values of these *NAME*s may not be changed by subsequent assignment. If the `-f` option is given, then functions corresponding to the NAMEs are so marked. If no arguments are given, or if `-p` is given, a list of all readonly names is printed. The `-a` option means to treat each NAME as an array variable. An argument of `--` disables further option processing.

`readonly`	`readonly [-af] [NAME[=VALUE] ...]`
	`readonly -p`
	
	The given *NAME*s are marked readonly and the values of these *NAME*s may not be changed by subsequent assignment. If the `-f` option is given, then functions corresponding to the *NAME*s are so marked. If no arguments are given, or if `-p` is given, a list of all readonly names is printed. The `-a` option means to treat each *NAME* as an array variable. An argument of `--` disables further option processing.

`return`	`return [N]` Causes a function to exit with the return value specified by *N*. If *N* is omitted, the return status is that of the last command.
`select`	`select NAME [in WORDS ... ;] do COMMANDS; done` The *WORDS* are expanded, generating a list of words. The set of expanded words is printed on the standard error, each preceded by a number. If `in WORDS` is not present, `in "$@"` is assumed. The `PS3` prompt is then displayed and a line read from the standard input. If the line consists of the number corresponding to one of the displayed words, then *NAME* is set to that word. If the line is empty, *WORDS* and

Command	Syntax/Description
	the prompt are redisplayed. If EOF is read, the command completes. Any other value read causes *NAME* to be set to null. The line read is saved in the variable *REPLY*. *COMMANDS* are executed after each selection until a break command is executed.
set	`set [--abefhkmnptuvxBCHP] [-o OPTION] [ARG...]` Sets internal shell options.
shift	`shift [n]` The positional parameters from `$N+1 ...` are renamed to `$1 ...` If *N* is not given, it is assumed to be 1.
shopt	`shopt [-pqsu] [-o long-option] OPTNAME [OPTNAME...]` Toggle the values of variables controlling optional behavior. The `-s` flag means to enable (set) each *OPTNAME*; the `-u` flag unsets each *OPTNAME*. The `-q` flag suppresses output; the exit status indicates whether each *OPTNAME* is set or unset. The `-o` option restricts the *OPTNAME*s to those defined for use with `set -o`. With no options, or with the `-p` option, a list of all settable options is displayed, with an indication of whether or not each is set.
source	`source FILENAME [ARGS]` Read and execute commands from *FILENAME* and return. The pathnames in `$PATH` are used to find the directory containing *FILENAME*. If any *ARGS* are supplied, they become the positional parameters when *FILENAME* is executed.
suspend	`suspend [-f]` Suspend the execution of this shell until it receives a SIGCONT signal. The `-f` if specified says not to complain about this being a login shell if it is; just suspend anyway.

Command	Syntax/Description
test	`test [expr]` Exits with a status of 0 (true) or 1 (false) depending on the evaluation of *EXPR*. Expressions may be unary or binary. Unary expressions are often used to examine the status of a file. There are string operators as well, and numeric comparison operators.
time	`time [-p] PIPELINE` Execute *PIPELINE* and print a summary of the real time, user CPU time, and system CPU time spent executing *PIPELINE* when it terminates. The return status is the return status of PIPELINE. The `-p` option prints the timing summary in a slightly different format. This uses the value of the TIMEFORMAT variable as the output format.
times	`times` Print the accumulated user and system times for processes run from the shell.
trap	`trap [-lp] [ARG SIGNAL_SPEC ...]` The command *ARG* is to be read and executed when the shell receives signal(s) *SIGNAL_SPEC*. If *ARG* is absent (and a single *SIGNAL_SPEC* is supplied) or `-`, each specified signal is reset to its original value. If *ARG* is the null string each *SIGNAL_SPEC* is ignored by the shell and by the commands it invokes. If a *SIGNAL_SPEC* is EXIT (0) the command ARG is executed on exit from the shell. If a *SIGNAL_SPEC* is DEBUG, *ARG* is executed after every simple command. If the `-p` option is supplied then the trap commands associated with each *SIGNAL_SPEC* are displayed. If no arguments are supplied or if only `-p` is given, trap prints the list of commands associated with each signal. Each *SIGNAL_SPEC* is either a signal name in `signal.h` or a signal number. Signal names are case insensitive and the SIG prefix is optional. `trap -l` prints a list of signal names and their corresponding numbers. Note that a signal can be sent to the shell with `kill -signal $$`.

Command	Syntax/Description
`true`	`true` Return a successful result.
`type`	`type [-afptP] NAME [NAME ...]` Obsolete, see `declare`.
`typeset`	`typeset [-afFirtx] [-p] name[=value]` Obsolete, see `declare`.
`ulimit`	`ulimit [-SHacdfilmnpqstuvx] [limit` Ulimit provides control over the resources available to processes started by the shell, on systems that allow such control.
`umask`	`umask [-p] [-S] [MODE]` The user file-creation mask is set to *`MODE`*. If *`MODE`* is omitted, or if `-S` is supplied, the current value of the mask is printed. The `-S` option makes the output symbolic; otherwise an octal number is output. If `-p` is supplied, and *`MODE`* is omitted, the output is in a form that may be used as input. If *`MODE`* begins with a digit, it is interpreted as an octal number, otherwise it is a symbolic mode string like that accepted by chmod(1).
`unalias`	`unalias [-a] NAME [NAME ...]` Remove *`NAME`*s from the list of defined aliases. If the `-a` option is given, then remove all alias definitions.
`unset`	`unset [-f] [-v] [NAME ...]` For each *`NAME`*, remove the corresponding variable or function. Given the `-v`, unset will only act on variables. Given the `-f` flag, unset will only act on functions. With neither flag, unset first tries to unset a variable, and if that fails, then tries to unset a function. Some variables cannot be unset; also see readonly.

Command	Syntax/Description
`until`	`until COMMANDS; do COMMANDS; done` Expand and execute *`COMMANDS`* as long as the final command in the `until` *`COMMANDS`* has an exit status which is not zero.
`wait`	`wait [N]` Wait for the specified process and report its termination status. If *`N`* is not given, all currently active child processes are waited for, and the return code is zero. *`N`* may be a process ID or a job specification; if a job spec is given, all processes in the job's pipeline are waited for.
`while`	`while COMMANDS; do COMMANDS; done` Expand and execute *`COMMANDS`* as long as the final command in the `while` *`COMMANDS`* has an exit status of zero.

crm_standby (8)

crm_standby — manipulate a node's standby attribute to determine whether resources can be run on this node

Synopsis

```
crm_standby [-?|-V] -D -u|-U node -r resource

crm_standby  [-?|-V] -G -u|-U node -r resource

crm_standby [-?|-V] -v string -u|-U node -r resource [-l string]
```

Description

The `crm_standby` command manipulates a node's standby attribute. Any node in standby mode is no longer eligible to host resources and any resources that are there must be moved. Standby mode can be useful for performing maintenance tasks, such as kernel updates. Remove the standby attribute from the node when it should become a fully active member of the cluster again.

By assigning a lifetime to the `standby` attribute, determine whether the standby setting should survive a reboot of the node (set lifetime to `forever`) or should be reset with reboot (set lifetime to `reboot`). Alternatively, remove the `standby` attribute and bring the node back from standby manually.

Options

`--help`, `-?`
: Print a help message.

`--verbose`, `-V`
: Turn on debug information.

> **NOTE**
>
> Increase the level of verbosity by providing additional instances.

`--quiet`, `-Q`
: When doing an attribute query using `-G`, print just the value to stdout. Use this option with `-G`.

`--get-value`, `-G`
: Retrieve rather than set the preference.

`--delete-attr`, `-D`
: Specify the attribute to delete.

`--attr-value` *`string`*, `-v` *`string`*
: Specify the value to use. This option is ignored when used with `-G`.

`--attr-id` *`string`*, `-i` *`string`*
: For advanced users only. Identifies the id attribute..

`--node` *`node_uname`*, `-u` *`node_uname`*
: Specify the uname of the node to change.

`--lifetime` *`string`*, `-l` *`string`*
: Determine how long this preference lasts. Possible values are `reboot` or `forever`.

NOTE

If a `forever` value exists, it is always used by the CRM instead of any `reboot` value.

Examples

Have a local node go to standby:

```
crm_standby -v true
```

Have a node (`node1`) go to standby:

```
crm_standby -v true -U node1
```

Query the standby status of a node:

```
crm_standby -G -U node1
```

Remove the standby property from a node:

```
crm_standby -D -U node1
```

Have a node go to standby for an indefinite period of time:

```
crm_standby -v true -l forever -U node1
```

Have a node go to standby until the next reboot of this node:

```
crm_standby -v true -l reboot -U node1
```

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk. Editing this file directly is strongly discouraged.

See Also

cibadmin(8) (page 128), crm_attribute(8) (page 137)

Author

`crm_standby` was written by Andrew Beekhof.

crm_verify (8)

crm_verify — check the CIB for consistency

Synopsis

```
crm_verify [-V] -x file
crm_verify [-V] -X string
crm_verify [-V] -L|-p
crm_verify [-?]
```

Description

`crm_verify` checks the configuration database (CIB) for consistency and other problems. It can be used to check a file containing the configuration or can it can connect to a running cluster. It reports two classes of problems, errors and warnings. Errors must be fixed before Heartbeat can work properly. However, it is left up to the administrator to decide if the warnings should also be fixed.

`crm_verify` assists in creating new or modified configurations. You can take a local copy of a CIB in the running cluster, edit it, validate it using `crm_verify`, then put the new configuration into effect using `cibadmin`.

Options

`--help`, `-h`
: Print a help message.

`--verbose`, `-V`
: Turn on debug information.

> **NOTE**
>
> Increase the level of verbosity by providing additional instances.

`--live-check`, `-L`
: Connect to the running cluster and check the CIB.

`--crm_xml` *`string`*, `-X` *`string`*
: Check the configuration in the supplied string. Pass complete CIBs only.

`--xml-file` *`file`*, `-x` *`file`*
: Check the configuration in the named file.

`--xml-pipe`, `-p`
: Use the configuration piped in via stdin. Pass complete CIBs only.

Examples

Check the consistency of the configuration in the running cluster and produce verbose output:

```
crm_verify -VL
```

Check the consistency of the configuration in a given file and produce verbose output:

```
crm_verify -Vx file1
```

Pipe a configuration into `crm_verify` and produce verbose output:

```
cat file1.xml | crm_verify -Vp
```

Files

`/var/lib/heartbeat/crm/cib.xml`—the CIB (minus status section) on disk. Editing this file directly is strongly discouraged.

See Also

cibadmin(8) (page 128)

Author

`crm_verify` was written by Andrew Beekhof.

14 Cluster Resources

This chapter summarizes the most important facts and figures related to cluster resources: the resource agent classes the High Availability Extension supports, the error codes for OCF resource agents and how the cluster reacts to the error codes, the available resource options, resource operations and instance attributes. Use this overview as a reference when configuring resources (either manually with the `crm` line tool or with the Linux HA Management Client).

14.1 Supported Resource Agent Classes

For each cluster resource you add, you need to define the standard that the resource agent confirms to. Resource agents abstract the services they provides and present an accurate status to the cluster, which allows the cluster to be agnostic about the resources it manages. The cluster relies on the resource agent to do the right thing when given a start, stop or monitor command.

Typically resource agents come in the form of shell scripts. The High Availability Extension supports the following classes of resource agents:

Legacy Heartbeat 1 Resource Agents
: Heartbeat version 1 came with its own style of resource agents. As many people have written their own agents based on its conventions, these resource agents are still supported. However, it is recommended to migrate your configurations to High

Availability OCF RAs if possible. For more information, see http://wiki.linux-ha.org/HeartbeatResourceAgent.

Linux Standards Base (LSB) Scripts
: LSB resource agents are generally provided by the operating system/distribution and are found in /etc/init.d. To be used with the cluster, they must conform to the LSB specification. For example, they must have several actions implemented, which are at least start, stop, restart, reload, force-reload, and status as explained in http://www.linuxbase.org/spec/refspecs/LSB_3.0.0/LSB-Core- generic/LSB-Core-generic/iniscrptact.html.

Open Cluster Framework (OCF) Resource Agents
: OCF RA agents are suited best for use with High Availability, especially when you need master resources or special monitoring abilities. The agents are generally located in /usr/lib/ocf/resource.d/heartbeat/. Their functionality is similar to that of LSB scripts. However, the configuration is always done with environment variables which allows them to accept and process parameters easily. The OCF specification (as it relates to resource agents) can be found at http://www.opencf.org/cgi-bin/viewcvs.cgi/specs/ra/resource-agent-api.txt?rev=HEAD_4. OCF specifications have strict definitions of which exit codes must be return by actions. The cluster follows these specifications exactly. For more information, see http://wiki.linux-ha.org/OCFResourceAgent. For a detailed list of all available OCF RAs, refer to Chapter 15, *HA OCF Agents* (page 189).

STONITH Resource Agents
: This class is used exclusively for fencing related resources. For more information, see Chapter 8, *Fencing and STONITH* (page 81).

The agents supplied with the High Availability Extension are written to OCF specifications.

14.2 OCF Return Codes

According to the OCF specification, there are strict definitions of the exit codes an action must return. The cluster always checks the return code against the expected result. If

the result does not match the expected value, then the operation is considered to have failed and a recovery action is initiated. There are three types of failure recovery:

Table 14.1 *Failure Recovery Types*

Recovery Type	Description	Action Taken by the Cluster
soft	A transient error occurred.	Restart the resource or move it to a new location.
hard	A non-transient error occurred. The error may be specific to the current node.	Move the resource elsewhere and prevent it from being retried on the current node.
fatal	A non-transient error occurred that will be common to all cluster nodes. This means a bad configuration was specified.	Stop the resource and prevent it from being started on any cluster node.

Assuming an action is considered to have failed, the following table outlines the different OCF return codes and the type of recovery the cluster will initiate when the respective error code is received.

Table 14.2 *OCF Return Codes*

OCF Return Code	OCF Alias	Description	Recovery Type
0	OCF_SUCCESS	Success. The command completed successfully. This is the expected result for all start, stop, promote and demote commands.	soft
1	OCF_ERR_GENERIC	Generic "there was a problem" error code.	soft

OCF Return Code	OCF Alias	Description	Recovery Type
2	OCF_ERR_ARGS	The resource's configuration is not valid on this machine (for example, it refers to a location/tool not found on the node).	hard
3	OCF_ERR_UNIMPLEMENTED	The requested action is not implemented.	hard
4	OCF_ERR_PERM	The resource agent does not have sufficient privileges to complete the task.	hard
5	OCF_ERR_INSTALLED	The tools required by the resource are not installed on this machine.	hard
6	OCF_ERR_CONFIGURED	The resource's configuration is invalid (for example, required parameters are missing).	fatal
7	OCF_NOT_RUNNING	The resource is not running. The cluster will not attempt to stop a resource that returns this for any action. This OCF return code may or may not require resource recovery—it depends on what is the expected resource status. If unexpected, then `soft` recovery.	N/A
8	OCF_RUNNING_MASTER	The resource is running in Master mode.	soft
9	OCF_FAILED_MASTER	The resource is in Master mode but has failed. The resource will be demoted, stopped and then started (and possibly promoted) again.	soft
other	N/A	Custom error code.	soft

14.3 Resource Options

For each resource you add, you can define options. Options are used by the cluster to decide how your resource should behave—they tell the CRM how to treat a specific resource. Resource options can be set with the `crm_resource --meta` command or with the GUI as described in .

Table 14.3 *Options for a Primitive Resource*

Option	Description
`priority`	If not all resources can be active, the cluster will stop lower priority resources in order to keep higher priority ones active.
`target-role`	What state should the cluster attempt to keep this resource in? Allowed values: `Stopped`, `Started`.
`is-managed`	Is the cluster allowed to start and stop the resource? Allowed values: `true`, `false`.
`resource-stickiness`	How much does the resource prefer to stay where it is? Defaults to the value of `default-resource-stickiness`.
`migration-threshold`	How many failures should occur for this resource on a node before making the node ineligible to host this resource? Default: `none`.
`multiple-active`	What should the cluster do if it ever finds the resource active on more than one node? Allowed values: `block` (mark the resource as unmanaged), `stop_only`, `stop_start`.
`failure-timeout`	How many seconds to wait before acting as if the failure had not occurred (and potentially allowing the resource back to the node on which it failed)? Default: `never`.

14.4 Resource Operations

By default, the cluster will not ensure that your resources are still healthy. To instruct the cluster to do this, you need to add a monitor operation to the resource's definition. Monitor operations can be added for all classes or resource agents.

Table 14.4 *Resource Operations*

Operation	Description
`id`	Your name for the action. Must be unique.
`name`	The action to perform. Common values: `monitor`, `start`, `stop`.
`interval`	How frequently to perform the operation. Unit: seconds .
`timeout`	How long to wait before declaring the action has failed.
`requires`	What conditions need to be satisfied before this action occurs. Allowed values: `nothing`, `quorum`, `fencing`. The default depends on whether fencing is enabled and if the resource's class is `stonith`. For STONITH resources, the default is `nothing`.
`on-fail`	The action to take if this action ever fails. Allowed values: • `ignore`: Pretend the resource did not fail. • `block`: Do not perform any further operations on the resource. • `stop`: Stop the resource and do not start it elsewhere. • `restart`: Stop the resource and start it again (possibly on a different node). • `fence`: Bring down the node on which the resource failed (STONITH).

Operation	Description
	• `standby`: Move *all* resources away from the node on which the resource failed.
`enabled`	If `false`, the operation is treated as if it does not exist. Allowed values: `true`, `false`.

14.5 Instance Attributes

The scripts of all resource classes can be given parameters which determine how they behave and which instance of a service they control. If your resource agent supports parameters, you can add them with the `crm_resource` command as described in . In the `crm` command line utility, instance attributes are called `params`. The list of instance attributes supported by an OCF script can be found by executing the following command as `root` :

```
crm ra meta resource_agent class
```

For example

```
crm ra meta Ipaddr ocf heartbeat
```

shows you

15 HA OCF Agents

All OCF agents require several parameters to be set when they are started. The following overview shows how to manually operate these agents. The data that is available in this appendix is directly taken from the `meta-data` invocation of the respective RA. Find all these agents in `/usr/lib/ocf/resource.d/heartbeat/`.

When configuring an RA, omit the `OCF_RESKEY_` prefix to the parameter name. Parameters that are in square brackets may be omitted in the configuration.

ocf:anything_ra (7)

ocf:anything_ra — anything

Synopsis

OCF_RESKEY_binfile=string [OCF_RESKEY_cmdline_options=string] [OCF_RESKEY_pidfile=string] [OCF_RESKEY_logfile=string] [OCF_RESKEY_errlogfile=string] [OCF_RESKEY_user=string] [OCF_RESKEY_monitor_hook=string] anything_ra [start | stop | monitor | meta-data | validate-all]

Description

This is a generic OCF RA to manage almost anything.

Supported Parameters

OCF_RESKEY_binfile=Full path name of the binary to be executed
The full name of the binary to be executed. This is expected to keep running with the same pid and not just do something and exit.

OCF_RESKEY_cmdline_options=Command line options
Command line options to pass to the binary

OCF_RESKEY_pidfile=File to write STDOUT to
File to read/write the PID from/to.

OCF_RESKEY_logfile=File to write STDOUT to
File to write STDOUT to

OCF_RESKEY_errlogfile=File to write STDERR to
File to write STDERR to

`OCF_RESKEY_user`=User to run the command as
User to run the command as

`OCF_RESKEY_monitor_hook`=Command to run in monitor operation
Command to run in monitor operation

ocf:apache (7)

ocf:apache — Apache web server

Synopsis

OCF_RESKEY_configfile=string [OCF_RESKEY_httpd=string] [OCF_RESKEY_port=integer] [OCF_RESKEY_statusurl=string] [OCF_RESKEY_testregex=string] [OCF_RESKEY_options=string] [OCF_RESKEY_envfiles=string] apache [start | stop | status | monitor | meta-data | validate-all]

Description

This is the resource agent for the Apache web server. Thie resource agent operates both version 1.x and version 2.x Apache servers. The start operation ends with a loop in which monitor is repeatedly called to make sure that the server started and that it is operational. Hence, if the monitor operation does not succeed within the start operation timeout, the apache resource will end with an error status. The monitor operation by default loads the server status page which depends on the mod_status module and the corresponding configuration file (usually /etc/apache2/mod_status.conf). Make sure that the server status page works and that the access is allowed *only* from localhost (address 127.0.0.1). See the statusurl and testregex attributes for more details. See also http://httpd.apache.org/

Supported Parameters

OCF_RESKEY_configfile=configuration file path
: The full pathname of the Apache configuration file. This file is parsed to provide defaults for various other resource agent parameters.

OCF_RESKEY_httpd=httpd binary path
: The full pathname of the httpd binary (optional).

`OCF_RESKEY_port`=httpd port

A port number that we can probe for status information using the statusurl. This will default to the port number found in the configuration file, or 80, if none can be found in the configuration file.

`OCF_RESKEY_statusurl`=url name

The URL to monitor (the apache server status page by default). If left unspecified, it will be inferred from the apache configuration file. If you set this, make sure that it succeeds *only* from the localhost (127.0.0.1). Otherwise, it may happen that the cluster complains about the resource being active on multiple nodes.

`OCF_RESKEY_testregex`=monitor regular expression

Regular expression to match in the output of statusurl. It is case insensitive.

`OCF_RESKEY_options`=command line options

Extra options to apply when starting apache. See man httpd(8).

`OCF_RESKEY_envfiles`=environment settings files

Files (one or more) which contain extra environment variables, such as /etc/apache2/envvars.

ocf:AudibleAlarm (7)

ocf:AudibleAlarm — AudibleAlarm resource agent

Synopsis

[OCF_RESKEY_nodelist=string] AudibleAlarm [start | stop | restart | status | monitor | meta-data | validate-all]

Description

Resource script for AudibleAlarm. It sets an audible alarm running by beeping at a set interval.

Supported Parameters

OCF_RESKEY_nodelist=Node list

The node list that should never sound the alarm.

ocf:ClusterMon (7)

ocf:ClusterMon — ClusterMon resource agent

Synopsis

[OCF_RESKEY_user=string] [OCF_RESKEY_update=integer]
[OCF_RESKEY_extra_options=string] OCF_RESKEY_pidfile=string
OCF_RESKEY_htmlfile=string ClusterMon [start | stop | monitor | meta-data | validate-all]

Description

This is a ClusterMon Resource Agent. It outputs current cluster status to the html.

Supported Parameters

OCF_RESKEY_user=The user we want to run crm_mon as
The user we want to run crm_mon as

OCF_RESKEY_update=Update interval
How frequently should we update the cluster status

OCF_RESKEY_extra_options=Extra options
Additional options to pass to crm_mon. Eg. -n -r

OCF_RESKEY_pidfile=PID file
PID file location to ensure only one instance is running

OCF_RESKEY_htmlfile=HTML output
Location to write HTML output to.

ocf:db2 (7)

ocf:db2 — db2 resource agent

Synopsis

[OCF_RESKEY_instance=string] [OCF_RESKEY_admin=string] db2 [start | stop | status | monitor | validate-all | meta-data | methods]

Description

Resource script for db2. It manages a DB2 Universal Database instance as an HA resource.

Supported Parameters

OCF_RESKEY_instance=instance
The instance of database.

OCF_RESKEY_admin=admin
The admin user of the instance.

ocf:Delay (7)

ocf:Delay — Delay resource agent

Synopsis

[OCF_RESKEY_startdelay=integer] [OCF_RESKEY_stopdelay=integer] [OCF_RESKEY_mondelay=integer] Delay [start | stop | status | monitor | meta-data | validate-all]

Description

This script is a test resource for introducing delay.

Supported Parameters

OCF_RESKEY_startdelay=Start delay
How long in seconds to delay on start operation.

OCF_RESKEY_stopdelay=Stop delay
How long in seconds to delay on stop operation. Defaults to "startdelay" if unspecified.

OCF_RESKEY_mondelay=Monitor delay
How long in seconds to delay on monitor operation. Defaults to "startdelay" if unspecified.

ocf:drbd (7)

ocf:drbd — This resource agent manages a Distributed Replicated Block Device (DRBD) object as a master/slave resource. DRBD is a mechanism for replicating storage; please see the documentation for setup details.

Synopsis

OCF_RESKEY_drbd_resource=string [OCF_RESKEY_drbdconf=string] [OCF_RESKEY_clone_overrides_hostname=boolean] [OCF_RESKEY_clone_max=integer] [OCF_RESKEY_clone_node_max=integer] [OCF_RESKEY_master_max=integer] [OCF_RESKEY_master_node_max=integer] drbd [start | promote | demote | notify | stop | monitor | monitor | meta-data | validate-all]

Description

Master/Slave OCF Resource Agent for DRBD

Supported Parameters

OCF_RESKEY_drbd_resource=drbd resource name
The name of the drbd resource from the drbd.conf file.

OCF_RESKEY_drbdconf=Path to drbd.conf
Full path to the drbd.conf file.

OCF_RESKEY_clone_overrides_hostname=Override drbd hostname
Whether or not to override the hostname with the clone number. This can be used to create floating peer configurations; drbd will be told to use node_<cloneno> as the hostname instead of the real uname, which can then be used in drbd.conf.

OCF_RESKEY_clone_max=Number of clones
Number of clones of this drbd resource. Do not fiddle with the default.

`OCF_RESKEY_clone_node_max`=Number of nodes
: Clones per node. Do not fiddle with the default.

`OCF_RESKEY_master_max`=Number of primaries
: Maximum number of active primaries. Do not fiddle with the default.

`OCF_RESKEY_master_node_max`=Number of primaries per node
: Maximum number of primaries per node. Do not fiddle with the default.

ocf:Dummy (7)

ocf:Dummy — Dummy resource agent

Synopsis

`OCF_RESKEY_state`=string `Dummy` [start | stop | monitor | reload | migrate_to | migrate_from | meta-data | validate-all]

Description

This is a Dummy Resource Agent. It does absolutely nothing except keep track of whether its running or not. Its purpose in life is for testing and to serve as a template for RA writers.

Supported Parameters

`OCF_RESKEY_state`=State file

Location to store the resource state in.

ocf:eDir88 (7)

ocf:eDir88 — eDirectory resource agent

Synopsis

OCF_RESKEY_eDir_config_file=string
[OCF_RESKEY_eDir_monitor_ldap=boolean]
[OCF_RESKEY_eDir_monitor_idm=boolean]
[OCF_RESKEY_eDir_jvm_initial_heap=integer]
[OCF_RESKEY_eDir_jvm_max_heap=integer]
[OCF_RESKEY_eDir_jvm_options=string] eDir88 [start | stop | monitor | meta-data | validate-all]

Description

Resource script for managing an eDirectory instance. Manages a single instance of eDirectory as an HA resource. The "multiple instances" feature or eDirectory has been added in version 8.8. This script will not work for any version of eDirectory prior to 8.8. This RA can be used to load multiple eDirectory instances on the same host. It is very strongly recommended to put eDir configuration files (as per the eDir_config_file parameter) on local storage on each node. This is necessary for this RA to be able to handle situations where the shared storage has become unavailable. If the eDir configuration file is not available, this RA will fail, and heartbeat will be unable to manage the resource. Side effects include STONITH actions, unmanageable resources, etc... Setting a high action timeout value is _very_ _strongly_ recommended. eDir with IDM can take in excess of 10 minutes to start. If heartbeat times out before eDir has had a chance to start properly, mayhem _WILL ENSUE_. The LDAP module seems to be one of the very last to start. So this script will take even longer to start on installations with IDM and LDAP if the monitoring of IDM and/or LDAP is enabled, as the start command will wait for IDM and LDAP to be available.

Supported Parameters

`OCF_RESKEY_eDir_config_file`=eDir config file
Path to configuration file for eDirectory instance.

`OCF_RESKEY_eDir_monitor_ldap`=eDir monitor ldap
Should we monitor if LDAP is running for the eDirectory instance?

`OCF_RESKEY_eDir_monitor_idm`=eDir monitor IDM
Should we monitor if IDM is running for the eDirectory instance?

`OCF_RESKEY_eDir_jvm_initial_heap`=DHOST_INITIAL_HEAP value
Value for the DHOST_INITIAL_HEAP java environment variable. If unset, java defaults will be used.

`OCF_RESKEY_eDir_jvm_max_heap`=DHOST_MAX_HEAP value
Value for the DHOST_MAX_HEAP java environment variable. If unset, java defaults will be used.

`OCF_RESKEY_eDir_jvm_options`=DHOST_OPTIONS value
Value for the DHOST_OPTIONS java environment variable. If unset, original values will be used.

ocf:Filesystem (7)

ocf:Filesystem — Filesystem resource agent

Synopsis

[OCF_RESKEY_device=string] [OCF_RESKEY_directory=string]
[OCF_RESKEY_fstype=string] [OCF_RESKEY_options=string] Filesystem
[start | stop | notify | monitor | validate-all | meta-data]

Description

Resource script for Filesystem. It manages a Filesystem on a shared storage medium.

Supported Parameters

OCF_RESKEY_device=block device
: The name of block device for the filesystem, or -U, -L options for mount, or NFS mount specification.

OCF_RESKEY_directory=mount point
: The mount point for the filesystem.

OCF_RESKEY_fstype=filesystem type
: The optional type of filesystem to be mounted.

OCF_RESKEY_options=options
: Any extra options to be given as -o options to mount. For bind mounts, add "bind" here and set fstype to "none". We will do the right thing for options such as "bind,ro".

ocf:ICP (7)

ocf:ICP — ICP resource agent

Synopsis

[OCF_RESKEY_driveid=string] [OCF_RESKEY_device=string] ICP [start | stop | status | monitor | validate-all | meta-data]

Description

Resource script for ICP. It Manages an ICP Vortex clustered host drive as an HA resource.

Supported Parameters

OCF_RESKEY_driveid=ICP cluster drive ID
The ICP cluster drive ID.

OCF_RESKEY_device=device
The device name.

ocf:ids (7)

ocf:ids — OCF resource agent for the IBM's database server called Informix Dynamic Server (IDS)

Synopsis

[OCF_RESKEY_informixdir=string] [OCF_RESKEY_informixserver=string] [OCF_RESKEY_onconfig=string] [OCF_RESKEY_dbname=string] [OCF_RESKEY_sqltestquery=string] ids [start | stop | status | monitor | validate-all | meta-data | methods | usage]

Description

OCF resource agent to manage an IBM Informix Dynamic Server (IDS) instance as an High-Availability resource.

Supported Parameters

OCF_RESKEY_informixdir= INFORMIXDIR environment variable
: The value the environment variable INFORMIXDIR has after a typical installation of IDS. Or in other words: the path (without trailing '/') where IDS was installed to. If this parameter is unspecified the script will try to get the value from the shell environment.

OCF_RESKEY_informixserver= INFORMIXSERVER environment variable
: The value the environment variable INFORMIXSERVER has after a typical installation of IDS. Or in other words: the name of the IDS server instance to manage. If this parameter is unspecified the script will try to get the value from the shell environment.

OCF_RESKEY_onconfig= ONCONFIG environment variable
: The value the environment variable ONCONFIG has after a typical installation of IDS. Or in other words: the name of the configuration file for the IDS instance specified in INFORMIXSERVER. The specified configuration file will be searched

at '/etc/'. If this parameter is unspecified the script will try to get the value from the shell environment.

`OCF_RESKEY_dbname`= database to use for monitoring, defaults to 'sysmaster'
This parameter defines which database to use in order to monitor the IDS instance. If this parameter is unspecified the script will use the 'sysmaster' database as a default.

`OCF_RESKEY_sqltestquery`= SQL test query to use for monitoring, defaults to 'SELECT COUNT(*) FROM systables;'
SQL test query to run on the database specified by the parameter 'dbname' in order to monitor the IDS instance and determine if it's functional or not. If this parameter is unspecified the script will use 'SELECT COUNT(*) FROM systables;' as a default.

ocf:IPaddr2 (7)

ocf:IPaddr2 — Manages virtual IPv4 addresses

Synopsis

OCF_RESKEY_ip=string [OCF_RESKEY_nic=string]
[OCF_RESKEY_cidr_netmask=string] [OCF_RESKEY_broadcast=string]
[OCF_RESKEY_iflabel=string] [OCF_RESKEY_lvs_support=boolean]
[OCF_RESKEY_mac=string] [OCF_RESKEY_clusterip_hash=string]
[OCF_RESKEY_unique_clone_address=boolean]
[OCF_RESKEY_arp_interval=integer] [OCF_RESKEY_arp_count=integer]
[OCF_RESKEY_arp_bg=string] [OCF_RESKEY_arp_mac=string] IPaddr2 [start
| stop | status | monitor | meta-data | validate-all]

Description

This Linux-specific resource manages IP alias IP addresses. It can add an IP alias, or remove one. In addition, it can implement Cluster Alias IP functionality if invoked as a clone resource.

Supported Parameters

OCF_RESKEY_ip=IPv4 address
: The IPv4 address to be configured in dotted quad notation, for example "192.168.1.1".

OCF_RESKEY_nic=Network interface
: The base network interface on which the IP address will be brought online. If left empty, the script will try and determine this from the routing table. Do NOT specify an alias interface in the form eth0:1 or anything here; rather, specify the base interface only.

`OCF_RESKEY_cidr_netmask`=CIDR netmask
The netmask for the interface in CIDR format (e.g., 24 and not 255.255.255.0) If unspecified, the script will also try to determine this from the routing table.

`OCF_RESKEY_broadcast`=Broadcast address
Broadcast address associated with the IP. If left empty, the script will determine this from the netmask.

`OCF_RESKEY_iflabel`=Interface label
You can specify an additional label for your IP address here. This label is appended to your interface name. If a label is specified in nic name, this parameter has no effect.

`OCF_RESKEY_lvs_support`=Enable support for LVS DR
Enable support for LVS Direct Routing configurations. In case a IP address is stopped, only move it to the loopback device to allow the local node to continue to service requests, but no longer advertise it on the network.

`OCF_RESKEY_mac`=Cluster IP MAC address
Set the interface MAC address explicitly. Currently only used in case of the Cluster IP Alias. Leave empty to chose automatically.

`OCF_RESKEY_clusterip_hash`=Cluster IP hashing function
Specify the hashing algorithm used for the Cluster IP functionality.

`OCF_RESKEY_unique_clone_address`=Create a unique address for cloned instances
If true, add the clone ID to the supplied value of ip to create a unique address to manage

`OCF_RESKEY_arp_interval`=ARP packet interval in ms
Specify the interval between unsolicited ARP packets in milliseconds.

`OCF_RESKEY_arp_count`=ARP packet count
Number of unsolicited ARP packets to send.

`OCF_RESKEY_arp_bg`=ARP from background
Whether or not to send the arp packets in the background.

`OCF_RESKEY_arp_mac`=ARP MAC

MAC address to send the ARP packets too. You really shouldn't be touching this.

ocf:IPaddr (7)

ocf:IPaddr — Manages virtual IPv4 addresses

Synopsis

OCF_RESKEY_ip=string [OCF_RESKEY_nic=string]
[OCF_RESKEY_cidr_netmask=string] [OCF_RESKEY_broadcast=string]
[OCF_RESKEY_iflabel=string] [OCF_RESKEY_lvs_support=boolean]
[OCF_RESKEY_local_stop_script=string]
[OCF_RESKEY_local_start_script=string]
[OCF_RESKEY_ARP_INTERVAL_MS=integer] [OCF_RESKEY_ARP_REPEAT=integer] [OCF_RESKEY_ARP_BACKGROUND=boolean]
[OCF_RESKEY_ARP_NETMASK=string] IPaddr [start | stop | monitor | validate-all | meta-data]

Description

This script manages IP alias IP addresses It can add an IP alias, or remove one.

Supported Parameters

OCF_RESKEY_ip=IPv4 address
: The IPv4 address to be configured in dotted quad notation, for example "192.168.1.1".

OCF_RESKEY_nic=Network interface
: The base network interface on which the IP address will be brought online. If left empty, the script will try and determine this from the routing table. Do NOT specify an alias interface in the form eth0:1 or anything here; rather, specify the base interface only.

`OCF_RESKEY_cidr_netmask`=Netmask
The netmask for the interface in CIDR format. (ie, 24), or in dotted quad notation 255.255.255.0). If unspecified, the script will also try to determine this from the routing table.

`OCF_RESKEY_broadcast`=Broadcast address
Broadcast address associated with the IP. If left empty, the script will determine this from the netmask.

`OCF_RESKEY_iflabel`=Interface label
You can specify an additional label for your IP address here.

`OCF_RESKEY_lvs_support`=Enable support for LVS DR
Enable support for LVS Direct Routing configurations. In case a IP address is stopped, only move it to the loopback device to allow the local node to continue to service requests, but no longer advertise it on the network.

`OCF_RESKEY_local_stop_script`=Script called when the IP is released
Script called when the IP is released

`OCF_RESKEY_local_start_script`=Script called when the IP is added
Script called when the IP is added

`OCF_RESKEY_ARP_INTERVAL_MS`=milliseconds between gratuitous ARPs
milliseconds between ARPs

`OCF_RESKEY_ARP_REPEAT`=repeat count
How many gratuitous ARPs to send out when bringing up a new address

`OCF_RESKEY_ARP_BACKGROUND`=run in background
run in background (no longer any reason to do this)

`OCF_RESKEY_ARP_NETMASK`=netmask for ARP
netmask for ARP - in nonstandard hexadecimal format.

ocf:IPsrcaddr (7)

ocf:IPsrcaddr — IPsrcaddr resource agent

Synopsis

[OCF_RESKEY_ipaddress=string] IPsrcaddr [start | stop | stop | monitor | validate-all | meta-data]

Description

Resource script for IPsrcaddr. It manages the preferred source address modification.

Supported Parameters

OCF_RESKEY_ipaddress=IP address
The IP address.

ocf:IPv6addr (7)

ocf:IPv6addr — manages IPv6 alias

Synopsis

[OCF_RESKEY_ipv6addr=string] IPv6addr [start | stop | status | monitor | validate-all | meta-data]

Description

This script manages IPv6 alias IPv6 addresses,It can add an IP6 alias, or remove one.

Supported Parameters

OCF_RESKEY_ipv6addr=IPv6 address
The IPv6 address this RA will manage

ocf:iscsi (7)

ocf:iscsi — iscsi resource agent

Synopsis

[OCF_RESKEY_portal=string] OCF_RESKEY_target=string [OCF_RESKEY_discovery_type=string] [OCF_RESKEY_iscsiadm=string] [OCF_RESKEY_udev=string] iscsi [start | stop | status | monitor | validate-all | methods | meta-data]

Description

OCF Resource Agent for iSCSI. Add (start) or remove (stop) iSCSI targets.

Supported Parameters

OCF_RESKEY_portal=portal
: The iSCSI portal address in the form: {ip_address|hostname}[":"port]

OCF_RESKEY_target=target
: The iSCSI target.

OCF_RESKEY_discovery_type=discovery_type
: Discovery type. Currently, with open-iscsi, only the sendtargets type is supported.

OCF_RESKEY_iscsiadm=iscsiadm
: iscsiadm program path.

OCF_RESKEY_udev=udev
: If the next resource depends on the udev creating a device then we wait until it is finished. On a normally loaded host this should be done quickly, but you may be unlucky. If you are not using udev set this to "no", otherwise we will spin in a loop until a timeout occurs.

ocf:Ldirectord (7)

ocf:Ldirectord — Wrapper OCF Resource Agent for ldirectord

Synopsis

`OCF_RESKEY_configfile`=string [`OCF_RESKEY_ldirectord`=string]
`Ldirectord` [start | stop | monitor | meta-data | validate-all]

Description

It's a simple OCF RA wrapper for ldirectord and uses the ldirectord interface to create the OCF compliant interface. You win monitoring of ldirectord. Be warned: Asking ldirectord status is an expensive action.

Supported Parameters

`OCF_RESKEY_configfile`=configuration file path
The full pathname of the ldirectord configuration file.

`OCF_RESKEY_ldirectord`=ldirectord binary path
The full pathname of the ldirectord.

ocf:LinuxSCSI (7)

ocf:LinuxSCSI — LinuxSCSI resource agent

Synopsis

[OCF_RESKEY_scsi=string] LinuxSCSI [start | stop | methods | status | monitor | meta-data | validate-all]

Description

This is a resource agent for LinuxSCSI. It manages the availability of a SCSI device from the point of view of the linux kernel. It make Linux believe the device has gone away, and it can make it come back again.

Supported Parameters

OCF_RESKEY_scsi=SCSI instance
The SCSI instance to be managed.

ocf:LVM (7)

ocf:LVM — LVM resource agent

Synopsis

[OCF_RESKEY_volgrpname=string] LVM [start | stop | status | monitor | methods | meta-data | validate-all]

Description

Resource script for LVM. It manages an Linux Volume Manager volume (LVM) as an HA resource.

Supported Parameters

OCF_RESKEY_volgrpname=Volume group name
The name of volume group.

ocf:MailTo (7)

ocf:MailTo — MailTo resource agent

Synopsis

[OCF_RESKEY_email=string] [OCF_RESKEY_subject=string] MailTo [start | stop | status | monitor | meta-data | validate-all]

Description

This is a resource agent for MailTo. It sends email to a sysadmin whenever a takeover occurs.

Supported Parameters

OCF_RESKEY_email=Email address
The email address of sysadmin.

OCF_RESKEY_subject=Subject
The subject of the email.

ocf:ManageRAID (7)

ocf:ManageRAID — Manages RAID devices

Synopsis

[OCF_RESKEY_raidname=string] ManageRAID [start | stop | status | monitor | validate-all | meta-data]

Description

Manages starting, stopping and monitoring of RAID devices which are preconfigured in /etc/conf.d/HB-ManageRAID.

Supported Parameters

OCF_RESKEY_raidname=RAID name

Name (case sensitive) of RAID to manage. (preconfigured in /etc/conf.d/HB-ManageRAID)

ocf:ManageVE (7)

ocf:ManageVE — OpenVZ VE resource agent

Synopsis

[OCF_RESKEY_veid=integer] ManageVE [start | stop | status | monitor | validate-all | meta-data]

Description

This OCF complaint resource agent manages OpenVZ VEs and thus requires a proper OpenVZ installation including a recent vzctl util.

Supported Parameters

OCF_RESKEY_veid=OpenVZ ID of VE

OpenVZ ID of virtual environment (see output of vzlist -a for all assigned IDs)

ocf:mysql (7)

ocf:mysql — MySQL resource agent

Synopsis

[OCF_RESKEY_binary=string] [OCF_RESKEY_config=string]
[OCF_RESKEY_datadir=string] [OCF_RESKEY_user=string]
[OCF_RESKEY_group=string] [OCF_RESKEY_log=string]
[OCF_RESKEY_pid=string] [OCF_RESKEY_socket=string]
[OCF_RESKEY_test_table=string] [OCF_RESKEY_test_user=string]
[OCF_RESKEY_test_passwd=string] [OCF_RESKEY_enable_creation=integer] [OCF_RESKEY_additional_parameters=integer] mysql [start | stop | status | monitor | validate-all | meta-data]

Description

Resource script for MySQL. It manages a MySQL Database instance as an HA resource.

Supported Parameters

OCF_RESKEY_binary=MySQL binary
Location of the MySQL binary

OCF_RESKEY_config=MySQL config
Configuration file

OCF_RESKEY_datadir=MySQL datadir
Directory containing databases

OCF_RESKEY_user=MySQL user
User running MySQL daemon

OCF_RESKEY_group=MySQL group
Group running MySQL daemon (for logfile and directory permissions)

`OCF_RESKEY_log`=MySQL log file
The logfile to be used for mysqld.

`OCF_RESKEY_pid`=MySQL pid file
The pidfile to be used for mysqld.

`OCF_RESKEY_socket`=MySQL socket
The socket to be used for mysqld.

`OCF_RESKEY_test_table`=MySQL test table
Table to be tested in monitor statement (in database.table notation)

`OCF_RESKEY_test_user`=MySQL test user
MySQL test user

`OCF_RESKEY_test_passwd`=MySQL test user password
MySQL test user password

`OCF_RESKEY_enable_creation`=Create the database if it does not exist
If the MySQL database does not exist, it will be created

`OCF_RESKEY_additional_parameters`=Additional paramters to pass to mysqld
Additional parameters which are passed to the mysqld on startup. (e.g. --skip-external-locking or --skip-grant-tables)

ocf:nfsserver (7)

ocf:nfsserver — nfsserver

Synopsis

[OCF_RESKEY_nfs_init_script=string]
[OCF_RESKEY_nfs_notify_cmd=string]
[OCF_RESKEY_nfs_shared_infodir=string] [OCF_RESKEY_nfs_ip=string]
nfsserver [start | stop | monitor | meta-data | validate-all]

Description

Nfsserver helps to manage the Linux nfs server as a failover-able resource in Linux-HA. It depends on Linux specific NFS implementation details, so is considered not portable to other platforms yet.

Supported Parameters

OCF_RESKEY_nfs_init_script= Init script for nfsserver

The default init script shipped with the Linux distro. The nfsserver resource agent offloads the start/stop/monitor work to the init script because the procedure to start/stop/monitor nfsserver varies on different Linux distro.

OCF_RESKEY_nfs_notify_cmd= The tool to send out notification.

The tool to send out NSM reboot notification. Failover of nfsserver can be considered as rebooting to different machines. The nfsserver resource agent use this command to notify all clients about the happening of failover.

OCF_RESKEY_nfs_shared_infodir= Directory to store nfs server related information.

The nfsserver resource agent will save nfs related information in this specific directory. And this directory must be able to fail-over before nfsserver itself.

`OCF_RESKEY_nfs_ip`= IP address.

The floating IP address used to access the the nfs service

ocf:oracle (7)

ocf:oracle — oracle resource agent

Synopsis

OCF_RESKEY_sid=string [OCF_RESKEY_home=string]
[OCF_RESKEY_user=string] [OCF_RESKEY_ipcrm=string]
[OCF_RESKEY_clear_backupmode=boolean]
[OCF_RESKEY_shutdown_method=string] oracle [start | stop | status | monitor | validate-all | methods | meta-data]

Description

Resource script for oracle. Manages an Oracle Database instance as an HA resource.

Supported Parameters

OCF_RESKEY_sid=sid
The Oracle SID (aka ORACLE_SID).

OCF_RESKEY_home=home
The Oracle home directory (aka ORACLE_HOME). If not specified, then the SID along with its home should be listed in /etc/oratab.

OCF_RESKEY_user=user
The Oracle owner (aka ORACLE_OWNER). If not specified, then it is set to the owner of file $ORACLE_HOME/dbs/*${ORACLE_SID}.ora. If this does not work for you, just set it explicitely.

OCF_RESKEY_ipcrm=ipcrm
Sometimes IPC objects (shared memory segments and semaphores) belonging to an Oracle instance might be left behind which prevents the instance from starting. It is not easy to figure out which shared segments belong to which instance, in particular when more instances are running as same user. What we use here is the

"oradebug" feature and its "ipc" trace utility. It is not optimal to parse the debugging information, but I am not aware of any other way to find out about the IPC information. In case the format or wording of the trace report changes, parsing might fail. There are some precautions, however, to prevent stepping on other peoples toes. There is also a dumpinstipc option which will make us print the IPC objects which belong to the instance. Use it to see if we parse the trace file correctly. Three settings are possible: - none: don't mess with IPC and hope for the best (beware: you'll probably be out of luck, sooner or later) - instance: try to figure out the IPC stuff which belongs to the instance and remove only those (default; should be safe) - orauser: remove all IPC belonging to the user which runs the instance (don't use this if you run more than one instance as same user or if other apps running as this user use IPC) The default setting "instance" should be safe to use, but in that case we cannot guarantee that the instance will start. In case IPC objects were already left around, because, for instance, someone mercilessly killing Oracle processes, there is no way any more to find out which IPC objects should be removed. In that case, human intervention is necessary, and probably _all_ instances running as same user will have to be stopped. The third setting, "orauser", guarantees IPC objects removal, but it does that based only on IPC objects ownership, so you should use that only if every instance runs as separate user. Please report any problems. Suggestions/fixes welcome.

`OCF_RESKEY_clear_backupmode`=clear_backupmode
The clear of the backup mode of ORACLE.

`OCF_RESKEY_shutdown_method`=shutdown_method
How to stop Oracle is a matter of taste it seems. The default method ("checkpoint/abort") is: alter system checkpoint; shutdown abort; This should be the fastest safe way bring the instance down. If you find "shutdown abort" distasteful, set this attribute to "immediate" in which case we will shutdown immediate; If you still think that there's even better way to shutdown an Oracle instance we are willing to listen.

ocf:oralsnr (7)

ocf:oralsnr — oralsnr resource agent

Synopsis

OCF_RESKEY_sid=string [OCF_RESKEY_home=string] [OCF_RESKEY_user=string] OCF_RESKEY_listener=string oralsnr [start | stop | status | monitor | validate-all | meta-data | methods]

Description

Resource script for Oracle Listener. It manages an Oracle Listener instance as an HA resource.

Supported Parameters

OCF_RESKEY_sid=sid
: The Oracle SID (aka ORACLE_SID). Necessary for the monitor op, i.e. to do tnsping SID.

OCF_RESKEY_home=home
: The Oracle home directory (aka ORACLE_HOME). If not specified, then the SID should be listed in /etc/oratab.

OCF_RESKEY_user=user
: Run the listener as this user.

OCF_RESKEY_listener=listener
: Listener instance to be started (as defined in listener.ora). Defaults to LISTENER.

ocf:pgsql (7)

ocf:pgsql — pgsql resource agent

Synopsis

[OCF_RESKEY_pgctl=string] [OCF_RESKEY_start_opt=string]
[OCF_RESKEY_ctl_opt=string] [OCF_RESKEY_psql=string]
[OCF_RESKEY_pgdata=string] [OCF_RESKEY_pgdba=string]
[OCF_RESKEY_pghost=string] [OCF_RESKEY_pgport=string]
[OCF_RESKEY_pgdb=string] [OCF_RESKEY_logfile=string]
[OCF_RESKEY_stop_escalate=string] pgsql [start | stop | status | monitor | meta-data | validate-all | methods]

Description

Resource script for PostgreSQL. It manages a PostgreSQL as an HA resource.

Supported Parameters

OCF_RESKEY_pgctl=pgctl
Path to pg_ctl command.

OCF_RESKEY_start_opt=start_opt
Start options (-o start_opt in pgi_ctl). "-i -p 5432" for example.

OCF_RESKEY_ctl_opt=ctl_opt
Additional pg_ctl options (-w, -W etc..). Default is ""

OCF_RESKEY_psql=psql
Path to psql command.

OCF_RESKEY_pgdata=pgdata
Path PostgreSQL data directory.

`OCF_RESKEY_pgdba`=pgdba
User that owns PostgreSQL.

`OCF_RESKEY_pghost`=pghost
Hostname/IP Addreess where PosrgeSQL is listening

`OCF_RESKEY_pgport`=pgport
Port where PosrgeSQL is listening

`OCF_RESKEY_pgdb`=pgdb
Database that will be used for monitoring.

`OCF_RESKEY_logfile`=logfile
Path to PostgreSQL server log output file.

`OCF_RESKEY_stop_escalate`=stop escalation
Number of retries (using -m fast) before resorting to -m immediate

ocf:pingd (7)

ocf:pingd — pingd resource agent

Synopsis

[OCF_RESKEY_pidfile=string] [OCF_RESKEY_user=string]
[OCF_RESKEY_dampen=integer] [OCF_RESKEY_set=integer]
[OCF_RESKEY_name=integer] [OCF_RESKEY_section=integer]
[OCF_RESKEY_multiplier=integer] [OCF_RESKEY_host_list=integer]
pingd [start | stop | monitor | meta-data | validate-all]

Description

This is a pingd Resource Agent. It records (in the CIB) the current number of ping nodes a node can connect to.

Supported Parameters

OCF_RESKEY_pidfile=PID file
PID file

OCF_RESKEY_user=The user we want to run pingd as
The user we want to run pingd as

OCF_RESKEY_dampen=Dampening interval
The time to wait (dampening) further changes occur

OCF_RESKEY_set=Set name
The name of the instance_attributes set to place the value in. Rarely needs to be specified.

OCF_RESKEY_name=Attribute name
The name of the attributes to set. This is the name to be used in the constraints.

`OCF_RESKEY_section`=Section name

The section place the value in. Rarely needs to be specified.

`OCF_RESKEY_multiplier`=Value multiplier

The number by which to multiply the number of connected ping nodes by

`OCF_RESKEY_host_list`=Host list

The list of ping nodes to count. Defaults to all configured ping nodes. Rarely needs to be specified.

ocf:portblock (7)

ocf:portblock — portblock resource agent

Synopsis

[OCF_RESKEY_protocol=string] [OCF_RESKEY_portno=integer] [OCF_RESKEY_action=string] portblock [start | stop | status | monitor | meta-data | validate-all]

Description

Resource script for portblock. It is used to temporarily block ports using iptables.

Supported Parameters

OCF_RESKEY_protocol=protocol
The protocol used to be blocked/unblocked.

OCF_RESKEY_portno=portno
The port number used to be blocked/unblocked.

OCF_RESKEY_action=action
The action (block/unblock) to be done on the protocol::portno.

ocf:Pure-FTPd (7)

ocf:Pure-FTPd — OCF Resource Agent compliant FTP script.

Synopsis

OCF_RESKEY_script=string OCF_RESKEY_conffile=string
OCF_RESKEY_daemon_type=string [OCF_RESKEY_pidfile=string]
Pure-FTPd [start | stop | monitor | validate-all | meta-data]

Description

This script manages Pure-FTPd in an Active-Passive setup

Supported Parameters

OCF_RESKEY_script=Script name with full path
The full path to the Pure-FTPd startup script. For example, "/sbin/pure-config.pl"

OCF_RESKEY_conffile=Configuration file name with full path
The Pure-FTPd configuration file name with full path. For example, "/etc/pure-ftpd/pure-ftpd.conf"

OCF_RESKEY_daemon_type=Configuration file name with full path
The Pure-FTPd daemon to be called by pure-ftpd-wrapper. Valid options are "" for pure-ftpd, "mysql" for pure-ftpd-mysql, "postgresql" for pure-ftpd-postgresql and "ldap" for pure-ftpd-ldap

OCF_RESKEY_pidfile=PID file
PID file

ocf:Raid1 (7)

ocf:Raid1 — RAID1 resource agent

Synopsis

[OCF_RESKEY_raidconf=string] [OCF_RESKEY_raiddev=string] [OCF_RESKEY_homehost=string] Raid1 [start | stop | status | monitor | validate-all | meta-data]

Description

Resource script for RAID1. It manages a software Raid1 device on a shared storage medium.

Supported Parameters

OCF_RESKEY_raidconf=RAID config file
The RAID configuration file. e.g. /etc/raidtab or /etc/mdadm.conf.

OCF_RESKEY_raiddev=block device
The block device to use.

OCF_RESKEY_homehost=Homehost for mdadm
The value for the homehost directive; this is an mdadm feature to protect RAIDs against being activated by accident. It is recommended to create RAIDs managed by the cluster with "homehost" set to a special value, so they are not accidentially auto-assembled by nodes not supposed to own them.

ocf:Route (7)

ocf:Route — Manages network routes

Synopsis

OCF_RESKEY_destination=string OCF_RESKEY_device=string OCF_RESKEY_gateway=string OCF_RESKEY_source=string Route [start | stop | monitor | reload | meta-data | validate-all]

Description

Enables and disables network routes. Supports host and net routes, routes via a gateway address, and routes using specific source addresses. This resource agent is useful if a node's routing table needs to be manipulated based on node role assignment. Consider the following example use case: - One cluster node serves as an IPsec tunnel endpoint. - All other nodes use the IPsec tunnel to reach hosts in a specific remote network. Then, here is how you would implement this scheme making use of the Route resource agent: - Configure an ipsec LSB resource. - Configure a cloned Route OCF resource. - Create an order constraint to ensure that ipsec is started before Route. - Create a collocation constraint between the ipsec and Route resources, to make sure no instance of your cloned Route resource is started on the tunnel endpoint itself.

Supported Parameters

OCF_RESKEY_destination=Destination network
: The destination network (or host) to be configured for the route. Specify the netmask suffix in CIDR notation (e.g. "/24"). If no suffix is given, a host route will be created. Specify "0.0.0.0/0" or "default" if you want this resource to set the system default route.

OCF_RESKEY_device=Outgoing network device
: The outgoing network device to use for this route.

`OCF_RESKEY_gateway`=Gateway IP address
The gateway IP address to use for this route.

`OCF_RESKEY_source`=Source IP address
The source IP address to be configured for the route.

ocf:rsyncd (7)

ocf:rsyncd — OCF Resource Agent compliant rsync daemon script.

Synopsis

[OCF_RESKEY_binpath=string] [OCF_RESKEY_conffile=string] [OCF_RESKEY_bwlimit=string] rsyncd [start | stop | monitor | validate-all | meta-data]

Description

This script manages rsync daemon

Supported Parameters

OCF_RESKEY_binpath=Full path to the rsync binary
The rsync binary path. For example, "/usr/bin/rsync"

OCF_RESKEY_conffile=Configuration file name with full path
The rsync daemon configuration file name with full path. For example, "/etc/rsyncd.conf"

OCF_RESKEY_bwlimit=limit I/O bandwidth, KBytes per second
This option allows you to specify a maximum transfer rate in kilobytes per second. This option is most effective when using rsync with large files (several megabytes and up). Due to the nature of rsync transfers, blocks of data are sent, then if rsync determines the transfer was too fast, it will wait before sending the next data block. The result is an average transfer rate equaling the specified limit. A value of zero specifies no limit.

ocf:SAPDatabase (7)

ocf:SAPDatabase — SAP database resource agent

Synopsis

OCF_RESKEY_SID=string OCF_RESKEY_DIR_EXECUTABLE=string
OCF_RESKEY_DBTYPE=string OCF_RESKEY_NETSERVICENAME=string
OCF_RESKEY_DBJ2EE_ONLY=boolean OCF_RESKEY_JAVA_HOME=string
OCF_RESKEY_STRICT_MONITORING=boolean
OCF_RESKEY_AUTOMATIC_RECOVER=boolean
OCF_RESKEY_DIR_BOOTSTRAP=string OCF_RESKEY_DIR_SECSTORE=string
OCF_RESKEY_DB_JARS=string OCF_RESKEY_PRE_START_USEREXIT=string
OCF_RESKEY_POST_START_USEREXIT=string
OCF_RESKEY_PRE_STOP_USEREXIT=string
OCF_RESKEY_POST_STOP_USEREXIT=string SAPDatabase [start | stop | status | monitor | validate-all | meta-data | methods]

Description

Resource script for SAP databases. It manages a SAP database of any type as an HA resource.

Supported Parameters

OCF_RESKEY_SID=SAP system ID
The unique SAP system identifier. e.g. P01

OCF_RESKEY_DIR_EXECUTABLE=path of sapstartsrv and sapcontrol
The full qualified path where to find sapstartsrv and sapcontrol.

OCF_RESKEY_DBTYPE=database vendor
The name of the database vendor you use. Set either: ORA,DB6,ADA

OCF_RESKEY_NETSERVICENAME=listener name
The Oracle TNS listener name.

OCF_RESKEY_DBJ2EE_ONLY=only JAVA stack installed
If you do not have a ABAP stack installed in the SAP database, set this to TRUE

OCF_RESKEY_JAVA_HOME=Path to Java SDK
This is only needed if the DBJ2EE_ONLY parameter is set to true. Enter the path to the Java SDK which is used by the SAP WebAS Java

OCF_RESKEY_STRICT_MONITORING=Activates application level monitoring
This controls how the resource agent monitors the database. If set to true, it will use SAP tools to test the connect to the database. Do not use with Oracle, because it will result in unwanted failovers in case of an archiver stuck

OCF_RESKEY_AUTOMATIC_RECOVER=Enable or disable automatic startup recovery
The SAPDatabase resource agent tries to recover a failed start attempt automaticaly one time. This is done by running a forced abort of the RDBMS and/or executing recovery commands.

OCF_RESKEY_DIR_BOOTSTRAP=path to j2ee bootstrap directory
The full qualified path where to find the J2EE instance bootstrap directory. e.g. /usr/sap/P01/J00/j2ee/cluster/bootstrap

OCF_RESKEY_DIR_SECSTORE=path to j2ee secure store directory
The full qualified path where to find the J2EE security store directory. e.g. /usr/sap/P01/SYS/global/security/lib/tools

OCF_RESKEY_DB_JARS=file name of the jdbc driver
The full qualified filename of the jdbc driver for the database connection test. It will be automaticaly read from the bootstrap.properties file in Java engine 6.40 and 7.00. For Java engine 7.10 the parameter is mandatory.

OCF_RESKEY_PRE_START_USEREXIT=path to a pre-start script
The full qualified path where to find a script or program which should be executed before this resource gets started.

OCF_RESKEY_POST_START_USEREXIT=path to a post-start script
The full qualified path where to find a script or program which should be executed after this resource got started.

`OCF_RESKEY_PRE_STOP_USEREXIT`=path to a pre-start script

The full qualified path where to find a script or program which should be executed before this resource gets stopped.

`OCF_RESKEY_POST_STOP_USEREXIT`=path to a post-start script

The full qualified path where to find a script or program which should be executed after this resource got stopped.

ocf:SAPInstance (7)

ocf:SAPInstance — SAP instance resource agent

Synopsis

OCF_RESKEY_InstanceName=string OCF_RESKEY_DIR_EXECUTABLE=string
OCF_RESKEY_DIR_PROFILE=string OCF_RESKEY_START_PROFILE=string
OCF_RESKEY_START_WAITTIME=string
OCF_RESKEY_AUTOMATIC_RECOVER=boolean
OCF_RESKEY_PRE_START_USEREXIT=string
OCF_RESKEY_POST_START_USEREXIT=string
OCF_RESKEY_PRE_STOP_USEREXIT=string
OCF_RESKEY_POST_STOP_USEREXIT=string SAPInstance [start | recover | stop | status | monitor | validate-all | meta-data | methods]

Description

Resource script for SAP. It manages a SAP Instance as an HA resource.

Supported Parameters

OCF_RESKEY_InstanceName=instance name: SID_INSTANCE_VIR-HOSTNAME
The full qualified SAP instance name. e.g. P01_DVEBMGS00_sapp01ci

OCF_RESKEY_DIR_EXECUTABLE=path of sapstartsrv and sapcontrol
The full qualified path where to find sapstartsrv and sapcontrol.

OCF_RESKEY_DIR_PROFILE=path of start profile
The full qualified path where to find the SAP START profile.

OCF_RESKEY_START_PROFILE=start profile name
The name of the SAP START profile.

OCF_RESKEY_START_WAITTIME=Check the successful start after that time (do not wait for J2EE-Addin)

After that time in seconds a monitor operation is executed by the resource agent. Does the monitor return SUCCESS, the start is handled as SUCCESS. This is useful to resolve timing problems with e.g. the J2EE-Addin instance.

OCF_RESKEY_AUTOMATIC_RECOVER=Enable or disable automatic startup recovery

The SAPInstance resource agent tries to recover a failed start attempt automaticaly one time. This is done by killing runing instance processes and executing cleanipc.

OCF_RESKEY_PRE_START_USEREXIT=path to a pre-start script

The full qualified path where to find a script or program which should be executed before this resource gets started.

OCF_RESKEY_POST_START_USEREXIT=path to a post-start script

The full qualified path where to find a script or program which should be executed after this resource got started.

OCF_RESKEY_PRE_STOP_USEREXIT=path to a pre-start script

The full qualified path where to find a script or program which should be executed before this resource gets stopped.

OCF_RESKEY_POST_STOP_USEREXIT=path to a post-start script

The full qualified path where to find a script or program which should be executed after this resource got stopped.

ocf:scsi2reserve (7)

ocf:scsi2reserve — scsi-2 reservation

Synopsis

[OCF_RESKEY_scsi_reserve=string] [OCF_RESKEY_sharedisk=string] [OCF_RESKEY_start_loop=string] scsi2reserve [start | stop | monitor | meta-data | validate-all]

Description

The scsi-2-reserve resource agent is a place holder for SCSI-2 reservation. A healthy instance of scsi-2-reserve resource, indicates the own of the specified SCSI device. This resource agent depends on the scsi_reserve from scsires package, which is Linux specific.

Supported Parameters

OCF_RESKEY_scsi_reserve= scsi_reserve command

The scsi_reserve is a command from scsires package. It helps to issue SCSI-2 reservation on SCSI devices.

OCF_RESKEY_sharedisk= Shared disk.

The shared disk that can be reserved.

OCF_RESKEY_start_loop= Times to re-try before giving up.

We are going to try several times before giving up. Start_loop indicates how many times we are going to re-try.

ocf:SendArp (7)

ocf:SendArp — SendArp resource agent

Synopsis

[OCF_RESKEY_ip=string] [OCF_RESKEY_nic=string] SendArp [start | stop | monitor | meta-data | validate-all]

Description

This script send out gratuitous Arp for an IP address

Supported Parameters

OCF_RESKEY_ip=IP address
The IP address for sending arp package.

OCF_RESKEY_nic=NIC
The nic for sending arp package.

ocf:ServeRAID (7)

ocf:ServeRAID — ServeRAID resource agent

Synopsis

[OCF_RESKEY_serveraid=integer] [OCF_RESKEY_mergegroup=integer] ServeRAID [start | stop | status | monitor | validate-all | meta-data | methods]

Description

Resource script for ServeRAID. It enables/disables shared ServeRAID merge groups.

Supported Parameters

OCF_RESKEY_serveraid=serveraid
The adapter number of the ServeRAID adapter.

OCF_RESKEY_mergegroup=mergegroup
The logical drive under consideration.

ocf:sfex (7)

ocf:sfex — SF-EX resource agent

Synopsis

[OCF_RESKEY_device=string] [OCF_RESKEY_index=integer]
[OCF_RESKEY_collision_timeout=integer]
[OCF_RESKEY_monitor_interval=integer]
[OCF_RESKEY_lock_timeout=integer] sfex [start | stop | monitor | meta-data]

Description

Resource script for SF-EX. It manages a shared storage medium exclusively .

Supported Parameters

OCF_RESKEY_device=block device
Block device path that stores exclusive control data.

OCF_RESKEY_index=index
Location in block device where exclusive control data is stored. 1 or more is specified. Default is 1.

OCF_RESKEY_collision_timeout=waiting time for lock acquisition
Waiting time when a collision of lock acquisition is detected. Default is 1 second.

OCF_RESKEY_monitor_interval=monitor interval
Monitor interval(sec). Default is 10 seconds

OCF_RESKEY_lock_timeout=Valid term of lock
Valid term of lock(sec). Default is 20 seconds.

ocf:SphinxSearchDaemon (7)

ocf:SphinxSearchDaemon — searchd resource agent

Synopsis

OCF_RESKEY_config=string [OCF_RESKEY_searchd=string]
[OCF_RESKEY_search=string] [OCF_RESKEY_testQuery=string]
SphinxSearchDaemon [start | stop | monitor | meta-data | validate-all]

Description

This is a searchd Resource Agent. It manages the Sphinx Search Daemon.

Supported Parameters

OCF_RESKEY_config=Configuration file
searchd configuration file

OCF_RESKEY_searchd=searchd binary
searchd binary

OCF_RESKEY_search=search binary
Search binary for functional testing in the monitor action.

OCF_RESKEY_testQuery=test query
Test query for functional testing in the monitor action. The query does not need to match any documents in the index. The purpose is merely to test whether the search daemon is is able to query its indices and respond properly.

ocf:Squid (7)

ocf:Squid — The RA of Squid

Synopsis

[OCF_RESKEY_squid_exe=string] OCF_RESKEY_squid_conf=string
OCF_RESKEY_squid_pidfile=string OCF_RESKEY_squid_port=integer
[OCF_RESKEY_squid_stop_timeout=integer]
[OCF_RESKEY_debug_mode=string] [OCF_RESKEY_debug_log=string] Squid
[start | stop | status | monitor | meta-data | validate-all]

Description

The resource agent of Squid. This manages a Squid instance as an HA resource.

Supported Parameters

OCF_RESKEY_squid_exe=Executable file
This is a required parameter. This parameter specifies squid's executable file.

OCF_RESKEY_squid_conf=Configuration file
This is a required parameter. This parameter specifies a configuration file for a squid instance managed by this RA.

OCF_RESKEY_squid_pidfile=Pidfile
This is a required parameter. This parameter specifies a process id file for a squid instance managed by this RA.

OCF_RESKEY_squid_port=Port number
This is a required parameter. This parameter specifies a port number for a squid instance managed by this RA. If plural ports are used, you must specifiy the only one of them.

OCF_RESKEY_squid_stop_timeout=Number of seconds to await to confirm a normal stop method

This is an omittable parameter. On a stop action, a normal stop method is firstly used. and then the confirmation of its completion is awaited for the specified seconds by this parameter. The default value is 10.

OCF_RESKEY_debug_mode=Debug mode

This is an optional parameter. This RA runs in debug mode when this parameter includes 'x' or 'v'. If 'x' is included, both of STDOUT and STDERR redirect to the logfile specified by "debug_log", and then the builtin shell option 'x' is turned on. It is similar about 'v'.

OCF_RESKEY_debug_log=A destination of the debug log

This is an optional and omittable parameter. This parameter specifies a destination file for debug logs and works only if this RA run in debug mode. Refer to "debug_mode" about debug mode. If no value is given but it's requied, it's made by the following rules: "/var/log/" as a directory part, the basename of the configuration file given by "syslog_ng_conf" as a basename part, ".log" as a suffix.

ocf:Stateful (7)

ocf:Stateful — Example stateful resource agent

Synopsis

`OCF_RESKEY_state`=string `Stateful` [start | stop | monitor | meta-data | validate-all]

Description

This is an example resource agent that impliments two states

Supported Parameters

`OCF_RESKEY_state`=State file
Location to store the resource state in

ocf:SysInfo (7)

ocf:SysInfo — SysInfo resource agent

Synopsis

[OCF_RESKEY_pidfile=string] [OCF_RESKEY_delay=string] SysInfo [start | stop | monitor | meta-data | validate-all]

Description

This is a SysInfo Resource Agent. It records (in the CIB) various attributes of a node Sample Linux output: arch: i686 os: Linux-2.4.26-gentoo-r14 free_swap: 1999 cpu_info: Intel(R) Celeron(R) CPU 2.40GHz cpu_speed: 4771.02 cpu_cores: 1 cpu_load: 0.00 ram_total: 513 ram_free: 117 root_free: 2.4 Sample Darwin output: arch: i386 os: Darwin-8.6.2 cpu_info: Intel Core Duo cpu_speed: 2.16 cpu_cores: 2 cpu_load: 0.18 ram_total: 2016 ram_free: 787 root_free: 13 Units: free_swap: Mb ram_*: Mb root_free: Gb cpu_speed (Linux): bogomips cpu_speed (Darwin): Ghz

Supported Parameters

OCF_RESKEY_pidfile=PID file
: PID file

OCF_RESKEY_delay=Dampening Delay
: Interval to allow values to stabilize

ocf:tomcat (7)

ocf:tomcat — tomcat resource agent

Synopsis

OCF_RESKEY_tomcat_name=string OCF_RESKEY_script_log=string
[OCF_RESKEY_tomcat_stop_timeout=integer]
[OCF_RESKEY_tomcat_suspend_trialcount=integer]
[OCF_RESKEY_tomcat_user=string] [OCF_RESKEY_statusurl=string]
[OCF_RESKEY_java_home=string] OCF_RESKEY_catalina_home=string
OCF_RESKEY_catalina_pid=string
[OCF_RESKEY_tomcat_start_opts=string]
[OCF_RESKEY_catalina_opts=string]
[OCF_RESKEY_catalina_rotate_log=string]
[OCF_RESKEY_catalina_rotatetime=integer] tomcat [start | stop | status | monitor | meta-data | validate-all]

Description

Resource script for tomcat. It manages a Tomcat instance as an HA resource.

Supported Parameters

OCF_RESKEY_tomcat_name=The name of the resource
The name of the resource

OCF_RESKEY_script_log=A destination of the log of this script
A destination of the log of this script

OCF_RESKEY_tomcat_stop_timeout=Time-out at the time of the stop
Time-out at the time of the stop

`OCF_RESKEY_tomcat_suspend_trialcount`=The re-try number of times awaiting a stop

The re-try number of times awaiting a stop

`OCF_RESKEY_tomcat_user`=A user name to start a resource

A user name to start a resource

`OCF_RESKEY_statusurl`=URL for state confirmation

URL for state confirmation

`OCF_RESKEY_java_home`=Home directory of the Java

Home directory of the Java

`OCF_RESKEY_catalina_home`=Home directory of Tomcat

Home directory of Tomcat

`OCF_RESKEY_catalina_pid`=A PID file name of Tomcat

A PID file name of Tomcat

`OCF_RESKEY_tomcat_start_opts`=Tomcat start options

Tomcat start options

`OCF_RESKEY_catalina_opts`=Catalina options

Catalina options

`OCF_RESKEY_catalina_rotate_log`=Rotate catalina.out flag

Rotate catalina.out flag

`OCF_RESKEY_catalina_rotatetime`=Time span of the rotate catalina.out

Time span of the rotate catalina.out

ocf:VIPArip (7)

ocf:VIPArip — Virtual IP Address by RIP2 protocol

Synopsis

OCF_RESKEY_ip=string [OCF_RESKEY_nic=string] VIPArip [start | stop | monitor | validate-all | meta-data]

Description

Virtual IP Address by RIP2 protocol. This script manages IP alias in different subnet with quagga/ripd. It can add an IP alias, or remove one.

Supported Parameters

OCF_RESKEY_ip=The IP address in different subnet
The IPv4 address in different subnet, for example "192.168.1.1".

OCF_RESKEY_nic=The nic for broadcast the route information
The nic for broadcast the route information. The ripd uses this nic to broadcast the route informaton to others

ocf:VirtualDomain (7)

ocf:VirtualDomain — Manages virtual domains

Synopsis

OCF_RESKEY_config=string [OCF_RESKEY_hypervisor=string]
[OCF_RESKEY_force_stop=boolean]
[OCF_RESKEY_migration_transport=string]
[OCF_RESKEY_monitor_scripts=string] VirtualDomain [start | stop | status | monitor | migrate_from | migrate_to | meta-data | validate-all]

Description

Resource agent for a virtual domain (a.k.a. domU, virtual machine, virtual environment etc., depending on context) managed by libvirtd.

Supported Parameters

OCF_RESKEY_config=Virtual domain configuration file
Absolute path to the libvirt configuration file, for this virtual domain.

OCF_RESKEY_hypervisor=Hypervisor URI
Hypervisor URI to connect to. See the libvirt documentation for details on supported URI formats. The default is system dependent.

OCF_RESKEY_force_stop=Force shutdown on stop
Forcefully shut down ("destroy") the domain on stop. Enable this only if your virtual domain (or your virtualization backend) does not support graceful shutdown.

OCF_RESKEY_migration_transport=Remote hypervisor transport
Transport used to connect to the remote hypervisor while migrating. Please refer to the libvirt documentation for details on transports available. If this parameter is omitted, the resource will use libvirt's default transport to connect to the remote hypervisor.

`OCF_RESKEY_monitor_scripts`=space-separated list of monitor scripts

To additionally monitor services within the virtual domain, add this parameter with a list of scripts to monitor. Note: when monitor scripts are used, the start and migrate_from operations will complete only when all monitor scripts have completed successfully. Be sure to set the timeout of these operations to accommodate this delay.

ocf:WAS6 (7)

ocf:WAS6 — WAS6 resource agent

Synopsis

[OCF_RESKEY_profile=string] WAS6 [start | stop | status | monitor | validate-all | meta-data | methods]

Description

Resource script for WAS6. It manages a Websphere Application Server (WAS6) as an HA resource.

Supported Parameters

OCF_RESKEY_profile=profile name
The WAS profile name.

ocf:WAS (7)

ocf:WAS — WAS resource agent

Synopsis

[OCF_RESKEY_config=string] [OCF_RESKEY_port=integer] WAS [start | stop | status | monitor | validate-all | meta-data | methods]

Description

Resource script for WAS. It manages a Websphere Application Server (WAS) as an HA resource.

Supported Parameters

OCF_RESKEY_config=configration file
The WAS-configuration file.

OCF_RESKEY_port=port
The WAS-(snoop)-port-number.

ocf:WinPopup (7)

ocf:WinPopup — WinPopup resource agent

Synopsis

[OCF_RESKEY_hostfile=string] WinPopup [start | stop | status | monitor | validate-all | meta-data]

Description

Resource script for WinPopup. It sends WinPopups message to a sysadmin's workstation whenever a takeover occurs.

Supported Parameters

OCF_RESKEY_hostfile=Host file

The file containing the hosts to send WinPopup messages to.

ocf:Xen (7)

ocf:Xen — Manages Xen DomUs

Synopsis

[OCF_RESKEY_xmfile=string] [OCF_RESKEY_name=string]
[OCF_RESKEY_allow_migrate=boolean]
[OCF_RESKEY_shutdown_timeout=boolean]
[OCF_RESKEY_allow_mem_management=boolean]
[OCF_RESKEY_reserved_Dom0_memory=string]
[OCF_RESKEY_monitor_scripts=string] Xen [start | stop | migrate_from | migrate_to | monitor | meta-data | validate-all]

Description

Resource Agent for the Xen Hypervisor. Manages Xen virtual machine instances by mapping cluster resource start and stop, to Xen create and shutdown, respectively. A note on names We will try to extract the name from the config file (the xmfile attribute). If you use a simple assignment statement, then you should be fine. Otherwise, if there's some python acrobacy involved such as dynamically assigning names depending on other variables, and we will try to detect this, then please set the name attribute. You should also do that if there is any chance of a pathological situation where a config file might be missing, for example if it resides on a shared storage. If all fails, we finally fall back to the instance id to preserve backward compatibility. Para-virtualized guests can also be migrated by enabling the meta_attribute allow_migrate.

Supported Parameters

OCF_RESKEY_xmfile=Xen control file
: Absolute path to the Xen control file, for this virtual machine.

OCF_RESKEY_name=Xen DomU name
: Name of the virtual machine.

`OCF_RESKEY_allow_migrate`=Use live migration

This bool parameters allows to use live migration for paravirtual machines.

`OCF_RESKEY_shutdown_timeout`=Shutdown escalation timeout

The Xen agent will first try an orderly shutdown using xm shutdown. Should this not succeed within this timeout, the agent will escalate to xm destroy, forcibly killing the node. If this is not set, it will default to two-third of the stop action timeout. Setting this value to 0 forces an immediate destroy.

`OCF_RESKEY_allow_mem_management`=Use dynamic memory management

This parameter enables dynamic adjustment of memory for start and stop actions used for Dom0 and the DomUs. The default is to not adjust memory dynamically.

`OCF_RESKEY_reserved_Dom0_memory`=Minimum Dom0 memory

In case memory management is used, this parameter defines the minimum amount of memory to be reserved for the dom0. The default minimum memory is 512MB.

`OCF_RESKEY_monitor_scripts`=list of space separated monitor scripts

To additionally monitor services within the unprivileged domain, add this parameter with a list of scripts to monitor. NB: In this case make sure to set the start-delay of the monitor operation to at least the time it takes for the DomU to start all services.

ocf:Xinetd (7)

ocf:Xinetd — Xinetd resource agent

Synopsis

[OCF_RESKEY_service=string] Xinetd [start | stop | restart | status | monitor | validate-all | meta-data]

Description

Resource script for Xinetd. It starts/stops services managed by xinetd. Note that the xinetd daemon itself must be running: we are not going to start it or stop it ourselves. Important: in case the services managed by the cluster are the only ones enabled, you should specify the -stayalive option for xinetd or it will exit on Heartbeat stop. Alternatively, you may enable some internal service such as echo.

Supported Parameters

OCF_RESKEY_service=service name
The service name managed by xinetd.

Part V. Appendix

GNU Licenses

A

This appendix contains the GNU General Public License and the GNU Free Documentation License.

GNU General Public License

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program”, below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does. Copyright (C) yyyy name of author

```
This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.


This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.


You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.


signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License [http://www.fsf.org/licenses/lgpl.html] instead of this License.

GNU Free Documentation License

Version 1.2, November 2002

Copyright (C) 2000,2001,2002 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (c) YEAR YOUR NAME.
```

```
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
only as published by the Free Software Foundation;
with the Invariant Section being this copyright notice and license.
A copy of the license is included in the section entitled “GNU
Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Terminology

active/active, active/passive
: A concept about how services are running on nodes. An active-passive scenario is one or more services are running on the active node and the passive nodes waits for the active node to fail. On the other hand, active-active means, each node is active and passive at the same time.

cluster
: A high-performance cluster is a group of computers (real or virtual) sharing application load to get things done fast. A high availability cluster is designed primarily to secure the highest possible availability of services.

cluster partition
: Whenever communication fails between one or more nodes and the rest of the cluster, a cluster partition occurs. The nodes of a cluster partition are still active and able to communicate with each other, but they are unaware of the nodes with which they cannot communicate. As the loss of the other partition cannot be confirmed, a split brain scenario develops (see also split brain (page 276)).

consensus cluster membership (CCM)
: The CCM determines which nodes make up the cluster and shares this information across the cluster. Any new addition and any loss of nodes or quorum is delivered by the CCM. A CCM module runs on each node of the cluster.

cluster information base (CIB)
: A representation of the whole cluster configuration and status (node membership, resources, constraints, etc.) written in XML and residing in memory. A master CIB is kept and maintained on the DC and replicated to the other nodes.

cluster resource manager (CRM)
: The main management entity responsible for coordinating all nonlocal interactions. Each node of the cluster has its own CRM, but the one running on the DC is the one elected to relay decisions to the other nonlocal CRMs and process their input. A CRM interacts with a number of components: local resource managers both on its own node and on the other nodes, nonlocal CRMs, administrative commands, the fencing functionality, and the membership layer.

designated coordinator (DC)
: The “master” node. This node is where the master copy of the CIB is kept. All other nodes get their configuration and resource allocation information from the current DC. The DC is elected from all nodes in the cluster after a membership change.

Distributed replicated block device (drbd)
: DRBD is a block device designed for building high availability clusters. The whole block device is mirrored via a dedicated network and is seen as a network RAID-1.

failover
: Occurs when a resource or node fails on one machine and the affected resources are started on another node.

fencing
: Describes the concept of preventing access to a shared resource by non-cluster members. It can be archived by killing (shutting down) a “misbehaving” node to prevent it from causing trouble, locking resources away from a node whose status is uncertain, or in several other ways. Furthermore fencing is distinguished between node and resource fencing.

Heartbeat resource agent
: Heartbeat resource agents were widely used with Heartbeat version 1. Their use is deprecated, but still supported in version 2. A Heartbeat resource agent can perform `start`, `stop`, and `status` operations and resides under `/etc/ha.d/resource.d` or `/etc/init.d`. For more information about Heartbeat resource agents, refer to `http://www.linux-ha.org/HeartbeatResourceAgent`.

local resource manager (LRM)
: The local resource manager (LRM) is responsible for performing operations on resources. It uses the resource agent scripts to carry out the work. The LRM is “dumb” in that it does not know of any policy by itself. It needs the DC to tell it what to do.

LSB resource agent
: LSB resource agents are standard LSB init scripts. LSB init scripts are not limited to use in a high availability context. Any LSB-compliant Linux system uses LSB init scripts to control services. Any LSB resource agent supports a `start`, `stop`, `restart`, `status` and `force-reload` option and may optionally provide

`try-restart` and `reload` as well. LSB resource agents are located in `/etc/init.d`. Find more information about LSB resource agents and the actual specification at http://www.linux-ha.org/LSBResourceAgent and http://www.linux-foundation.org/spec/refspecs/LSB_3.0.0/LSB-Core-generic/LSB-Core-generic/iniscrptact.html

node
: Any computer (real or virtual) that is a member of a cluster and invisible for the user.

pingd
: The ping daemon. It continuously contacts one or more servers outside the cluster with ICMP pings.

policy engine (PE)
: The policy engine computes the actions that need to be taken to implement policy changes in the CIB. This information is then passed on to the transaction engine, which in turn implements the policy changes in the cluster setup. The PE always runs on the DC.

OCF resource agent
: OCF resource agents are similar to LSB resource agents (init scripts). Any OCF resource agent must support `start`, `stop`, and `status` (sometimes called `monitor`) options. Additionally, they support a `metadata` option that returns the description of the resource agent type in XML. Additional options may be supported, but are not mandatory. OCF resource agents reside in `/usr/lib/ocf/resource.d/`*`provider`*`.` Find more information about OCF resource agents and a draft of the specification at http://www.linux-ha.org/OCFResourceAgent and http://www.opencf.org/cgi-bin/viewcvs.cgi/specs/ra/resource-agent-api.txt?rev=HEAD.

quorum
: In a cluster, a cluster partition is defined to have quorum (is "quorate") if it has the majority of nodes (or votes). Quorum distinguishes exactly one partition. It is part of the algorithm to prevent several disconnected partitions or nodes from proceeding and causing data and service corruption (split brain). Quorum is a prerequisite for fencing, which then ensures that quorum is indeed unique.

resource
: Any type of service or application that is known to Heartbeat. Examples include an IP address, a file system, or a database.

resource agent (RA)
: A resource agent (RA) is a script acting as a proxy to manage a resource. There are three different kinds of resource agents: OCF (Open Cluster Framework) resource agents, LSB resource agents (Standard LSB init scripts), and Heartbeat resource agents (Heartbeat v1 resources).

Single Point of Failure (SPOF)
: A single point of failure (SPOF) is any component of a cluster that, should it fail, triggers the failure of the entire cluster.

split brain
: A scenario in which the cluster nodes are divided into two or more groups that do not know of each other (either through a software or hardware failure). To prevent a split brain situation from badly affecting the entire cluster, STONITH must come to the rescue. Also known as a "partitioned cluster" scenario.

STONITH
: The acronym for "Shoot the other node in the head" which is basically bringing down a misbehaving node to prevent it from causing trouble in the cluster.

transition engine (TE)
: The transition engine (TE) takes the policy directives from the PE and carries them out. The TE always runs on the DC. From there, it instructs the local resource managers on the others nodes which actions to take.