Introduction to SUSE Linux Enterprise High Availability

This article explains the capabilities, architecture, core concepts and benefits of SUSE Linux Enterprise High Availability. A high-availability cluster is a group of servers (nodes) that work together to ensure the highest possible availability of data, applications and services. Learn about the product to help you decide whether to use it, or before you set up a cluster for the first time.

Publication Date: 06 Nov 2026

Contents

1 What is SUSE Linux Enterprise High Availability? 2

2 Core concepts 6

3 Architecture overview 10

4 Installation options 12

5 For more information 13

6 Legal Notice 13

A GNU Free Documentation License 14

HA glossary 21

1 What is SUSE Linux Enterprise High Availability?

SUSE® Linux Enterprise High Availability is an integrated suite of open source clustering technologies. A *high-availability* cluster is a group of servers (*nodes*) that work together to ensure the highest possible availability of data, applications and services. If one node fails, the *resources* that were running on it move to another node with little or no downtime. You can also manually move resources between nodes for load balancing, or to perform maintenance tasks with minimal downtime. Cluster resources can include Web sites, e-mail servers, databases, file systems, virtual machines and any other server-based applications or services that must be available to users at all times.

1.1 Product availability

SUSE Linux Enterprise High Availability is available with the following products:

SUSE Linux Enterprise Server

High Availability is available as an extension on supported architectures. This requires an additional registration code.

SUSE Linux Enterprise Server for SAP applications

High Availability is included with the base product on supported architectures. No additional registration code is required.

SUSE Linux Enterprise Server with Extension Pack for IBM Z and IBM LinuxONE

High Availability is included with this subscription and can be registered with the same registration code. No additional registration code is required.

1.2 Capabilities and benefits

SUSE Linux Enterprise High Availability eliminates single points of failure to improve the availability and manageability of critical resources. This helps you maintain business continuity, protect data integrity, and reduce unplanned downtime for critical workloads.

Multiple clustering options

SUSE Linux Enterprise High Availability clusters can be configured in different ways:

- Local clusters: a single cluster in one location (for example, all nodes are located in one data center). Network latency is minimal. Storage is typically accessed synchronously by all nodes.
- Metro clusters ("stretched" clusters): a single cluster that can stretch over multiple buildings or data centers, with all sites connected by Fibre Channel. Network latency is usually low. Storage is frequently replicated using mirroring or synchronous replication.
- Hybrid clusters: virtual servers can be clustered together with physical servers. This improves service availability and resource usage.



Important: No support for mixed architectures

Clusters with mixed architectures are not supported. All nodes in the same cluster must have the same processor platform: AMD64/Intel 64, IBM Z, POWER or Arm.

Flexible and scalable resource management

SUSE Linux Enterprise High Availability supports clusters of up to 32 nodes. Resources can automatically move to another node if the current node fails, or they can be moved manually to troubleshoot hardware or balance the workload. Resources can also be configured to move back to repaired nodes at a specific time. The cluster can stop and start resources based on configurable rules.

Wide range of resource agents

The cluster manages resources via *resource agents* (RAs). SUSE Linux Enterprise High Availability supports many different resource agents that are designed to manage specific types of applications or services, including Apache, IPv4, IPv6, NFS and many more. It also ships with resource agents for third-party applications such as IBM WebSphere Application Server.

Storage and data replication

SUSE Linux Enterprise High Availability supports Fibre Channel or iSCSI storage area networks (SANs), allowing you to dynamically assign and reassign server storage as needed. It also comes with GFS2 and the cluster Logical Volume Manager (Cluster LVM). For data replication, use DRBD* to mirror a resource's data from the active node to a standby node.

Support for virtualized environments

SUSE Linux Enterprise High Availability supports the mixed clustering of both physical and virtual Linux servers. The cluster can recognize and manage resources running in virtual servers and in physical servers, and can also manage KVM virtual machines as resources.

Disaster recovery

SUSE Linux Enterprise High Availability ships with Relax-and-Recover (ReaR), a disaster recovery framework that helps to back up and restore systems.

User-friendly administration tools

SUSE Linux Enterprise High Availability includes the following tools for configuration and administration:

- The *CRM Shell* (`crmsh`) is a command-line interface for installing, configuring and managing SUSE Linux Enterprise High Availability clusters. You can run most CRM Shell commands from any node in the cluster.
- *Hawk* is a Web-based graphical interface for configuring and managing High Availability clusters. You can use Hawk on any Linux or non-Linux machine with a Web browser and network access to the cluster nodes.

1.3 How does High Availability work?

The following figure shows a three-node cluster. Each of the nodes has a Web server installed and 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 nodes.

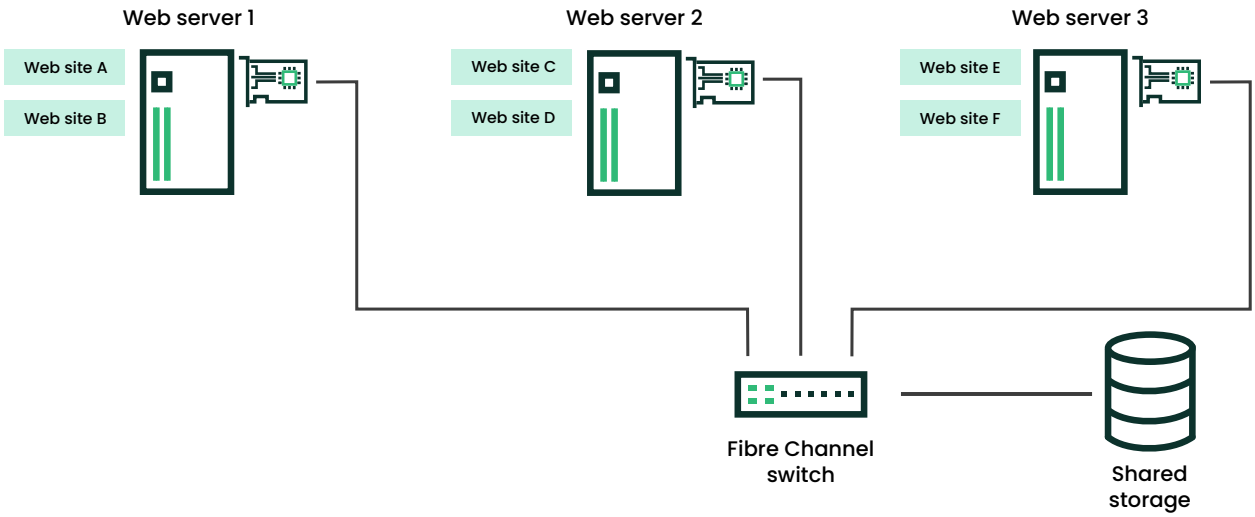


FIGURE 1: THREE-NODE CLUSTER

During normal cluster operation, each node is in constant communication with the other nodes in the cluster and periodically checks resources to detect failure.

The following figure shows how the cluster moves resources when Web server 1 fails due to hardware or software problems.

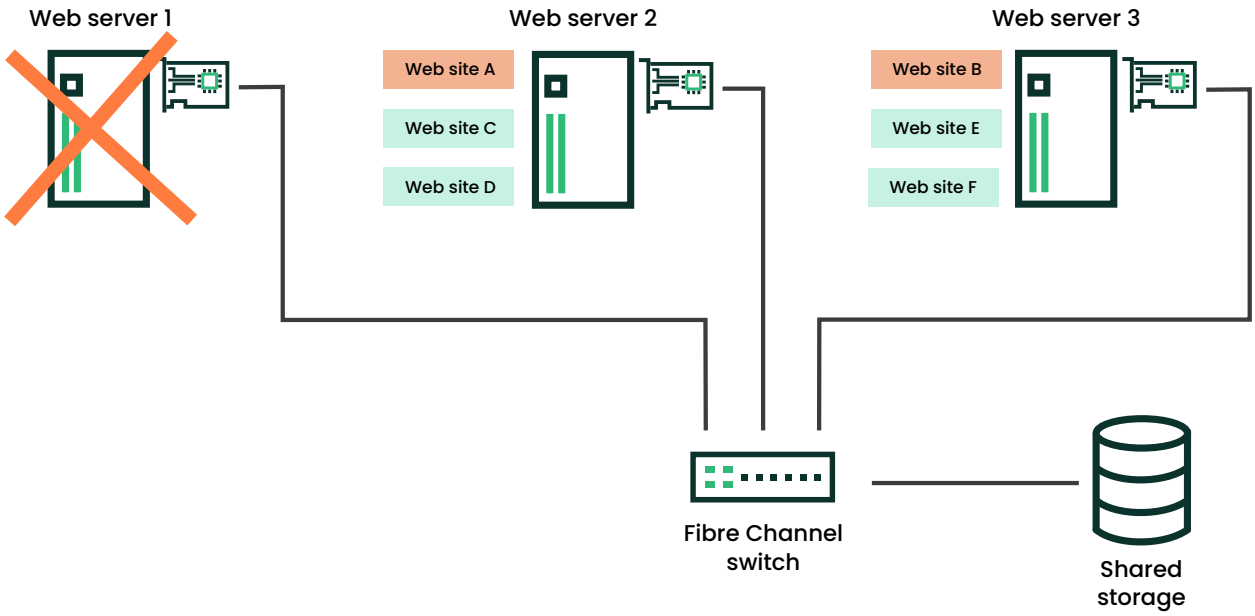


FIGURE 2: THREE-NODE CLUSTER AFTER ONE NODE FAILS

Web site A moves to Web server 2 and Web site B moves to Web server 3. The Web sites continue to be available and are evenly distributed between the remaining cluster nodes.

When Web server 1 failed, the High Availability software performed the following tasks:

1. Detected a failure and verified that Web server 1 was really down.
2. Remounted the shared data directories that were mounted on Web server 1 on Web servers 2 and 3.
3. Restarted applications that were running on Web server 1 on Web servers 2 and 3.
4. Transferred certificates and IP addresses to Web servers 2 and 3.

In this example, failover happened quickly and users regained access to the Web sites within seconds, usually without needing to log in again.

When Web server 1 returns to a normal operating state, Web site A and Web site B can move back ("fail back") to Web server 1 automatically. This incurs some downtime, so alternatively you can configure resources to only fail back at a specified time that causes minimal service interruption.

2 Core concepts

This section explains the core concepts of SUSE Linux Enterprise High Availability.

2.1 Clusters and nodes

A *high-availability* cluster is a group of servers that work together to ensure the availability of applications or services. Servers that are configured as members of the cluster are called *nodes*. If one node fails, the resources running on it can move to another node in the cluster. SUSE Linux Enterprise High Availability supports clusters with up to 32 nodes. However, clusters usually fall into one of two categories: two-node clusters, or clusters with an odd number of nodes (typically three or five).

2.2 Communication channels

Internal cluster communication is handled by *Corosync*. The Corosync Cluster Engine is a group communication system that provides messaging, membership and quorum information about the cluster. In SUSE Linux Enterprise High Availability 16, Corosync uses kronosnet (`knet`) as the default transport protocol.

We highly recommend configuring at least two communication channels for the cluster. The preferred method is to use network device bonding. If you cannot use a network bond, you can set up a redundant communication channel in Corosync (also known as a second "ring").

2.3 Resource management

In a High Availability cluster, the applications and services that need to be highly available are called *resources*. Cluster resources can include Web sites, e-mail servers, databases, file systems, virtual machines or any other applications or services that need to be available to users at all times. You can start, stop, monitor and move resources as needed. You can also specify whether resources should run together on the same node, or start and stop in sequential order. If a node fails, the resources running on it *fail over* (move) to another node instead of being lost.

In SUSE Linux Enterprise High Availability, the *cluster resource manager* (CRM) is *Pacemaker*, which manages and coordinates all cluster services. Pacemaker uses *resource agents* (RAs) to start, stop and monitor resources. A resource agent abstracts the resource and presents its status to the cluster. This means that Pacemaker only interacts with the resource agent, not with the actual application or service. SUSE Linux Enterprise High Availability supports many different resource agents that are designed to manage specific types of resources.

2.4 Node fencing

In a *split-brain scenario*, cluster nodes are divided into two or more groups (or *partitions*) that do not know about each other. This might be because of a hardware or software failure, or a failed network connection, for example. A split-brain scenario can be resolved by *fencing* (resetting or powering off) one or more of the nodes. Node fencing prevents a failed node from accessing shared resources and prevents cluster resources from running on a node with an uncertain status. This helps protect the cluster from data corruption.

To be supported, all SUSE Linux Enterprise High Availability clusters *must* have at least one node fencing device configured. For critical workloads, we recommend using two or three fencing devices. A fencing device can be either a physical device (a power switch) or a software mechanism (SBD in combination with a watchdog).

2.5 Quorum calculation

When communication fails between one or more nodes and the rest of the cluster (a *split-brain scenario*), a cluster *partition* occurs. The nodes can only communicate with other nodes in the same partition and are unaware of the separated nodes. A cluster partition has *quorum* (or is “quorate”) if it has the majority of nodes (or “votes”). This is determined by *quorum calculation*. Quorum must be calculated so the non-quorate nodes can be fenced.

Corosync calculates quorum based on the following formula:

```
N ≥ C/2 + 1

N = minimum number of operational nodes
C = number of cluster nodes
```

For example, a five-node cluster needs a minimum of three operational nodes to maintain quorum.

Clusters with an even number of nodes, especially two-node clusters, might have equal numbers of nodes in each partition and therefore no majority. To avoid this situation, you can configure the cluster to use QDevice in combination with QNetd. QNetd is an *arbitrator* running outside the cluster. It communicates with the QDevice daemon running on the cluster nodes to provide a configurable number of votes for quorum calculation. This lets a cluster sustain more node failures than the standard quorum rules allow.

2.6 Storage and data replication

High Availability clusters might include a shared disk subsystem connected via Fibre Channel or iSCSI. If a node fails, another node in the cluster automatically mounts the shared disk directories that were previously mounted on the failed node. This gives users continuous access to the directories on the shared disk subsystem. Content stored on the shared disk might include data, applications and services.

The following figure shows what a typical Fibre Channel cluster configuration might look like. The green lines depict connections to an Ethernet power switch, which can reboot a node if a ping request fails.

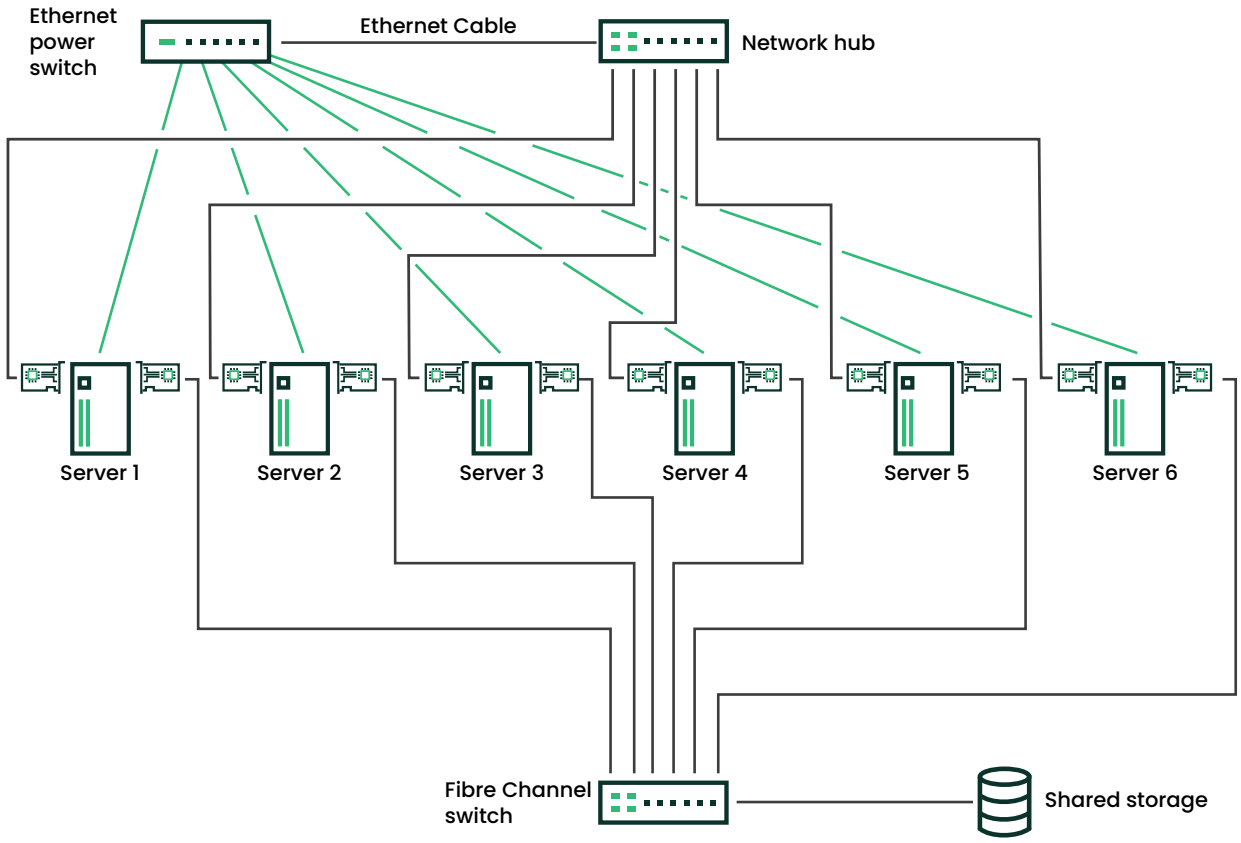


FIGURE 3: TYPICAL FIBRE CHANNEL CLUSTER CONFIGURATION

The following figure shows what a typical iSCSI cluster configuration might look like.

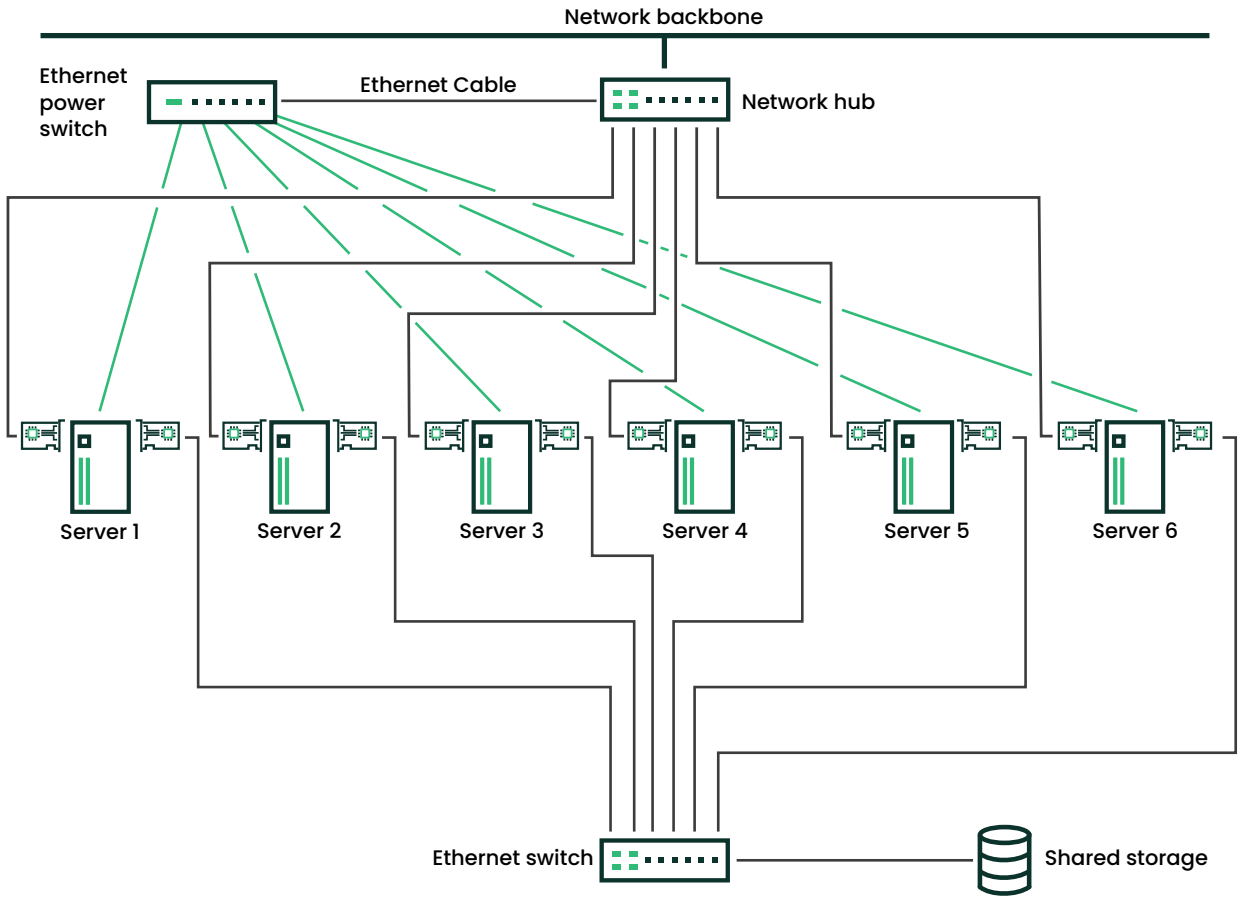


FIGURE 4: TYPICAL ISCSI CLUSTER CONFIGURATION

Although most clusters include a shared disk subsystem, you can also create a cluster without a shared disk subsystem. The following figure shows what a cluster without a shared disk subsystem might look like.

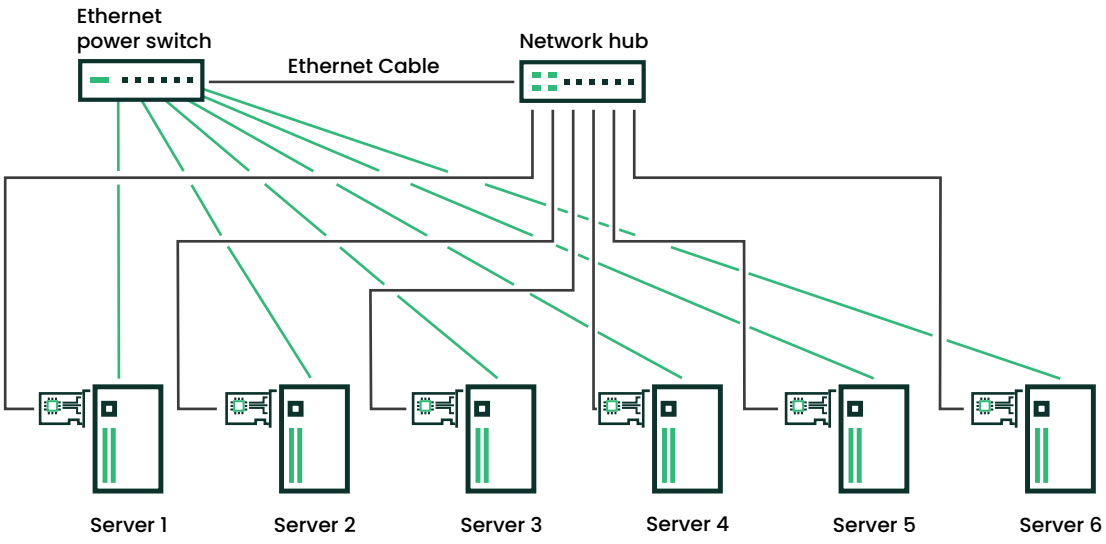


FIGURE 5: TYPICAL CLUSTER CONFIGURATION WITHOUT SHARED STORAGE

3 Architecture overview

This section explains the architecture of a SUSE Linux Enterprise High Availability cluster and how the different components interoperate.

3.1 Architecture layers

SUSE Linux Enterprise High Availability has a layered architecture. The following figure shows the different layers and their associated components.

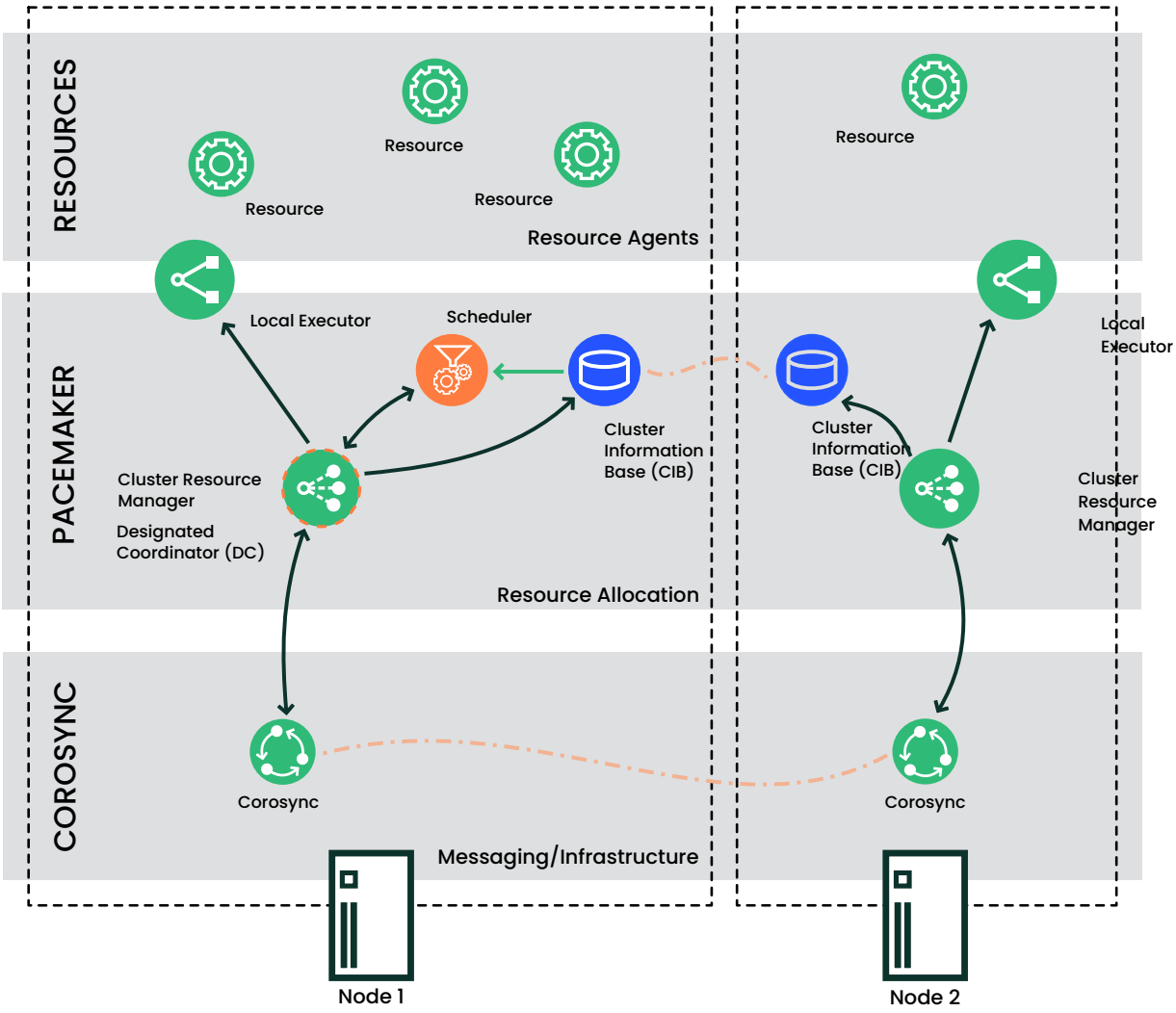


FIGURE 6: ARCHITECTURE OF A SUSE LINUX ENTERPRISE HIGH AVAILABILITY CLUSTER

Membership and messaging (Corosync)

The Corosync Cluster Engine is a group communication system that provides messaging, membership and quorum information about the cluster.

Cluster resource manager (Pacemaker)

Pacemaker is the cluster resource manager that reacts to events occurring in the cluster. Events might be nodes that join or leave the cluster, failure of resources, or scheduled activities such as maintenance, for example. The `pacemakerd` daemon launches and monitors all other related daemons. The following components are also part of the Pacemaker layer:

- **Cluster Information Database (CIB)**

  On every node, Pacemaker maintains the cluster information database (CIB). This is an XML representation of the cluster configuration (including cluster options, nodes, resources, constraints and the relationships to each other). The CIB also reflects the current cluster status. The CIB manager (`pacemaker-based`) keeps the CIB synchronized across the cluster and handles reading and writing cluster configuration and status.

- **Designated Coordinator (DC)**

  The `pacemaker-controld` daemon is the cluster controller, which coordinates all actions. This daemon has an instance on each cluster node, but only one instance is elected to act as the DC. The DC is elected when the cluster services start, or if the current DC fails or leaves the cluster. The DC decides whether a cluster-wide change must be performed, such as fencing a node or moving resources.

- **Scheduler**

  The scheduler runs on every node as the `pacemaker-schedulerd` daemon, but is only active on the DC. When a cluster transition is needed, the scheduler calculates the expected next state of the cluster and determines what actions need to be scheduled to achieve that state.

Local executor

The local executor is located between the Pacemaker layer and the resources layer on each node. The `pacemaker-execd` daemon allows Pacemaker to start, stop and monitor resources.

Resources and resource agents

In a High Availability cluster, the services that need to be highly available are called *resources*. Resource agents (RAs) are scripts that start, stop and monitor cluster resources.

3.2 Process flow

Many actions performed in the cluster cause a cluster-wide change, for example, adding or removing a cluster resource or changing resource constraints. The following example explains what happens in the cluster when you perform such an action:

EXAMPLE 1: CLUSTER PROCESS WHEN YOU ADD A NEW RESOURCE

1. You use the CRM Shell or Hawk to add a new cluster resource. You can do this from any node in the cluster. Adding the resource modifies the CIB.
2. The CIB change is replicated to all cluster nodes.
3. Based on the information in the CIB, `pacemaker-schedulerd` calculates the ideal state of the cluster and how it should be achieved. It feeds a list of instructions to the DC.
4. The DC sends commands via Corosync, which are received by the `pacemaker-controld` instances on the other nodes.
5. Each node uses its local executor (`pacemaker-execd`) to perform resource modifications. The `pacemaker-execd` daemon is not cluster-aware and interacts directly with resource agents.
6. All nodes report the results of their operations back to the DC.
7. If fencing is required, `pacemaker-fenced` calls the fencing agent to fence the node.
8. After the DC concludes that all necessary operations have been successfully performed, the cluster returns to an idle state and waits for further events.
9. If any operation was not carried out as planned, `pacemaker-schedulerd` is invoked again with the new information recorded in the CIB.

4 Installation options

This section describes the different options for installing a SUSE Linux Enterprise High Availability cluster and includes links to the available installation guides.

The following quick start guides explain how to set up a minimal cluster with default settings:

Installing a Basic Two-Node High Availability Cluster (https://documentation.suse.com/sle-ha/16.1/html/HA-installing-three-node-cluster/)

This quick start guide describes how to set up a basic two-node High Availability cluster with QDevice, diskless SBD and a software watchdog. QDevice is required for this setup so that diskless SBD can handle split-brain scenarios for the two-node cluster.

Installing a Basic Three-Node High Availability Cluster (https://documentation.suse.com/sle-ha/16.1/html/HA-installing-two-node-cluster/)

This quick start guide describes how to set up a basic three-node High Availability cluster with diskless SBD and a software watchdog. Three nodes are required for this setup so that diskless SBD can handle split-brain scenarios without the help of QDevice.

5 For more information

For more information about SUSE Linux Enterprise High Availability, see the following resources:

https://clusterlabs.org/

The upstream project for many of the components in SUSE Linux Enterprise High Availability.

https://clusterlabs.org/projects/pacemaker/doc/

Documentation for Pacemaker. For SUSE Linux Enterprise High Availability 16, see the documentation for Pacemaker 3.

6 Legal Notice

Copyright© 2006– 2026 SUSE LLC and contributors. All rights reserved.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or (at your option) version 1.3; with the Invariant Section being this copyright notice and license. A copy of the license version 1.2 is included in the section entitled “GNU Free Documentation License”.

For SUSE trademarks, see https://www.suse.com/company/legal/. All other third-party trademarks are the property of their respective owners. Trademark symbols (®, ™ etc.) denote trademarks of SUSE and its affiliates. Asterisks (*) denote third-party trademarks.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

A GNU Free Documentation License

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. 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 non-commercially. 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 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.

1. 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.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, 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.

3. 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.

4. 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.

5. 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".

6. 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.

7. 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.

8. 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.

9. 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.

10. 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 https://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

```
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
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
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.

HA glossary

ACL (access control list)

Access control lists can be used to limit or fine-tune a user's access to the CIB. An ACL consists of an ordered set of access rules. Each rule allows or denies read or write access to a part of the CIB. Rules can be combined into specific roles, and users can be assigned to a role that matches their tasks.

active/active, active/passive

How resources run on the nodes. Active/passive means that resources only run on the active node, but can move to the passive node if the active node fails. Active/active means that all nodes are active at once, and resources can run on (and move to) any node in the cluster.

arbitrator

An *arbitrator* is a machine running outside the cluster to provide an additional instance for cluster calculations. For example, *QNetd* provides a vote to help *QDevice* participate in *quorum* decisions.

CIB (cluster information base)

An XML representation of the whole cluster configuration and status (cluster options, nodes, resources, constraints, etc.). The CIB manager (`pacemaker-based`) keeps the CIB synchronized across the cluster nodes and handles requests to modify it.

clone

In the context of a cluster *resource*, a clone is a resource that can be active on multiple nodes. Any resource can be cloned if its resource agent supports it.

cluster

A *high-availability* cluster is a group of servers (physical or virtual) designed primarily to secure the highest possible availability of data, applications and services. Not to be confused with a *high-performance* cluster, which shares the application load to achieve faster results.

Cluster LVM (Cluster logical volume manager)

The term *Cluster LVM* indicates that LVM is being used in a cluster environment. This requires configuration adjustments to protect the LVM metadata on shared storage.

cluster offline upgrade

If a new product version includes major changes that are backward incompatible, the cluster requires a cluster offline upgrade. You must stop the cluster services on all nodes and upgrade the whole cluster before bringing the nodes back online.

For minor version upgrades, see *cluster rolling upgrade*.

cluster partition

A cluster partition occurs when communication fails between one or more nodes and the rest of the cluster. The nodes are split into partitions but are still active. They can only communicate with nodes in the same partition and are unaware of the separated nodes. This is known as a *split brain* scenario.

cluster rolling upgrade

In a cluster rolling upgrade, one cluster node at a time is upgraded while the rest of the cluster is still running. Stop the cluster services on a node, upgrade it, then bring it back online to join the cluster. Continue one by one until all cluster nodes are upgraded.

For major version upgrades, see *cluster offline upgrade*.

cluster services

The services that run the cluster. Typically this refers to Pacemaker and Corosync, but might also include SBD and QDevice if those services are used.

cluster stack

The ensemble of software technologies and components that make up a cluster.

colocation constraint

A type of *resource constraint* that specifies which resources can or cannot run together on a node.

concurrency violation

A resource that should be running on only one node in the cluster is running on several nodes.

Corosync

Corosync provides reliable messaging, membership and quorum information about the cluster. This is handled by the Corosync Cluster Engine, a group communication system.

CRM (cluster resource manager)

The management entity responsible for coordinating all non-local interactions in a High Availability cluster. SUSE Linux Enterprise High Availability uses *Pacemaker* as the CRM. It interacts with several components: local executors on its own node and on the other nodes, non-local CRMs, administrative commands, the fencing functionality, and the membership layer.

`crmsh` (CRM Shell)

The command-line utility `crmsh` manages the cluster, nodes and resources.

Csync2

A synchronization tool for replicating configuration files across all nodes in the cluster.

DC (designated coordinator)

The `pacemaker-controld` daemon is the cluster controller, which coordinates all actions. This daemon has an instance on each cluster node, but only one instance is elected to act as the DC. The DC is elected when the cluster services start, or if the current DC fails or leaves the cluster. The DC decides whether a cluster-wide change must be performed, such as fencing a node or moving resources.

disaster

An unexpected interruption of critical infrastructure caused by nature, humans, hardware failure, or software bugs.

disaster recovery

The process by which a function is restored to the normal, steady state after a disaster.

Disaster Recovery Plan

A strategy to recover from a disaster with the minimum impact on IT infrastructure.

DLM (Distributed Lock Manager)

DLM coordinates accesses to shared resources in a cluster, for example, managing file locking in clustered file systems to increase performance and availability.

DRBD

DRBD® is a block device designed for building High Availability clusters. It replicates data on a primary device to secondary devices in a way that ensures all copies of the data remain identical.

existing cluster

The term *existing cluster* is used to refer to any cluster that consists of at least one node. An existing cluster has a basic *Corosync* configuration that defines the communication channels, but does not necessarily have resource configuration yet.

failover

Occurs when a resource or node fails on one machine and the affected resources move to another node.

failover domain

A named subset of cluster nodes that are eligible to run a resource if a node fails.

fencing

Prevents access to a shared resource by isolated or failing cluster members. There are two classes of fencing: *resource-level* fencing and *node-level* fencing. Resource-level fencing ensures exclusive access to a resource. Node-level fencing prevents a failed node from accessing shared resources and prevents resources from running on a node with an uncertain status. This is usually done by resetting or powering off the node. In a *Pacemaker* cluster, node fencing is managed by the fencing subsystem `pacemaker-fenced`.

GFS2

Global File System 2 (GFS2) is a shared disk file system for Linux computer clusters. GFS2 allows all nodes to have direct concurrent access to the same shared block storage. GFS2 has no disconnected operating mode, and no client or server roles. All nodes in a GFS2 cluster function as peers. GFS2 supports up to 32 cluster nodes. Using GFS2 in a cluster requires hardware to allow access to the shared storage, and a lock manager to control access to the storage.

group

Resource groups contain multiple resources that need to be located together, started sequentially and stopped in the reverse order.

Hawk (High Availability Web Konsole)

A user-friendly Web-based interface for monitoring and administering a High Availability cluster from Linux or non-Linux machines. Hawk can be accessed from any machine that can connect to the cluster nodes, using a graphical Web browser.

heuristics

QDevice supports using a set of commands (*heuristics*) that run locally on start-up of cluster services, cluster membership change, successful connection to the *QNetd* server, or optionally at regular times. The result is used in calculations to determine which partition should have *quorum*.

knet (kronosnet)

A network abstraction layer supporting redundancy, security, fault tolerance, and fast fail-over of network links. In SUSE Linux Enterprise High Availability 16, *knet* is the default transport protocol for the *Corosync* communication channels.

local cluster

A single cluster in one location (for example, all nodes are located in one data center). Network latency is minimal. Storage is typically accessed synchronously by all nodes.

local executor

The local executor is located between *Pacemaker* and the resources on each node. Through the `pacemaker-execd` daemon, Pacemaker can start, stop and monitor resources.

location

In the context of a whole cluster, *location* can refer to the physical location of nodes (for example, all nodes might be located in the same data center). In the context of a *location constraint*, *location* refers to the nodes on which a resource can or cannot run.

location constraint

A type of *resource constraint* that defines the nodes on which a resource can or cannot run.

meta attribute

Parameters that tell the *CRM (cluster resource manager)* how to treat a specific *resource*. For example, you might define a resource's priority or target role.

metro cluster

A single cluster that can stretch over multiple buildings or data centers, with all sites connected by Fibre Channel. Network latency is usually low. Storage is frequently replicated using mirroring or synchronous replication.

network device bonding

Network device bonding combines two or more network interfaces into a single bonded device to increase bandwidth and/or provide redundancy. When using *Corosync*, the bonded device is not managed by the cluster software. Therefore, the bonded device must be configured on every cluster node that might need to access it.

node

Any server (physical or virtual) that is a member of a cluster.

order constraint

A type of *resource constraint* that defines the sequence of actions.

Pacemaker

Pacemaker is the *CRM (cluster resource manager)* in SUSE Linux Enterprise High Availability, or the "brain" that reacts to events occurring in the cluster. Events might be nodes that join or leave the cluster, failure of resources, or scheduled activities such as maintenance, for example. The `pacemakerd` daemon launches and monitors all other related daemons.

parameters (instance attributes)

Parameters determine which instance of a service the *resource* controls.

primitive

A primitive resource is the most basic type of cluster resource.

promotable clone

Promotable clones are a special type of *clone* resource that can be promoted. Active instances of these resources are divided into two states: promoted and unpromoted (also known as "active and passive" or "primary and secondary").

QDevice

QDevice and *QNetd* participate in *quorum* decisions. The `corosync-qdevice` daemon runs on each cluster node and communicates with QNetd to provide a configurable number of votes, allowing a cluster to sustain more node failures than the standard quorum rules allow.

QNetd

QNetd is an *arbitrator* that runs outside the cluster. The `corosync-qnetd` daemon provides a vote to the `corosync-qdevice` daemon on each node to help it participate in quorum decisions.

quorum

A *cluster partition* is defined to have quorum (be *quorate*) if it has the majority of nodes (or "votes"). Quorum distinguishes exactly one partition. This is part of the algorithm to prevent several disconnected partitions or nodes ("split brain") from proceeding and causing data and service corruption. Quorum is a prerequisite for fencing, which then ensures that quorum is unique.

RA (resource agent)

A script acting as a proxy to manage a *resource* (for example, to start, stop or monitor a resource). SUSE Linux Enterprise High Availability supports different kinds of resource agents.

ReaR (Relax and Recover)

An administrator tool set for creating *disaster recovery* images.

resource

Any type of service or application that is known to *Pacemaker*, for example, an IP address, a file system, or a database. The term *resource* is also used for *DRBD*, where it names a set of block devices that use a common connection for replication.

resource constraint

Resource constraints specify which cluster nodes a resource can run on, what order the resources should start in, and which other resources a specific resource is dependent on.

See also *colocation constraint*, *location constraint* and *order constraint*.

resource set

As an alternative format for defining location, colocation or order constraints, you can use *resource sets*, where primitives are grouped together in one set. When creating a constraint, you can specify multiple resources for the constraint to apply to.

resource template

To help create many resources with similar configurations, you can define a resource template. After being defined, it can be referenced in primitives or in certain types of constraints. If a template is referenced in a primitive, the primitive inherits all operations, instance attributes (parameters), meta attributes and utilization attributes defined in the template.

SBD (STONITH Block Device)

SBD provides a node *fencing* mechanism through the exchange of messages via shared block storage. Alternatively, it can be used in diskless mode. In either case, it needs a hardware or software *watchdog* on each node to ensure that misbehaving nodes are really stopped.

scheduler

The scheduler is implemented as `pacemaker-schedulerd`. When a cluster transition is needed, `pacemaker-schedulerd` calculates the expected next state of the cluster and determines what actions need to be scheduled to achieve the next state.

split brain

A scenario in which the cluster nodes are divided into two or more groups that do not know about each other (either through a software or hardware failure). Also known as a *partitioned cluster* scenario.

The term *split brain* is also used in *DRBD* but means that the nodes contain different data.

SPOF (single point of failure)

Any component of a cluster that, if it fails, triggers the failure of the entire cluster.

stickiness

Resource stickiness is a *meta attribute* that determines how much a resource prefers to stay on its current node.

switchover

The planned moving of resources to other nodes in a cluster. See also *failover*.

utilization

Tells the CRM what capacity a certain *resource* requires from a node.

watchdog

SBD (STONITH Block Device) needs a watchdog on each node to ensure that misbehaving nodes are really stopped. SBD "feeds" the watchdog by regularly writing a service pulse to it. If SBD stops feeding the watchdog, the hardware enforces a system restart. This protects against failures of the SBD process itself, such as becoming stuck on an I/O error.