SUSE

Installing a Basic Two-Node High Availability Cluster

Get started with SUSE Linux Enterprise High Availability by installing a basic two-node cluster with QDevice, diskless SBD and a software watchdog. This cluster can be used for testing purposes or as a minimal cluster configuration that can be extended later. Installing a basic High Availability cluster takes approximately 15 minutes, depending on the speed of your network connection (not including the time required to install SUSE Linux Enterprise Server).

Publication Date: 06 Nov 2026

Contents

1 Installation overview 3
2 System requirements 3
3 Enabling the High Availability extension 6
4 Setting up the QNetd server 6
5 Setting up the first node 7
6 Adding the second node 11
7 Logging in to Hawk 12
8 Checking the QDevice and QNetd setup 13
9 Testing the cluster 16
10 Next steps 18

11 Legal Notice 19

A GNU Free Documentation License 19

HA glossary 27

1 Installation overview

This guide describes the setup of a minimal High Availability cluster with the following properties:

- Two cluster nodes with passwordless SSH access to each other.
- A floating, virtual IP address that allows clients to connect to the graphical management tool Hawk, no matter which node the service is running on.
- Diskless SBD (STONITH Block Device) and a software watchdog used as the node fencing mechanism to avoid split-brain scenarios.
- QDevice working with QNetd to participate in cluster quorum decisions. QDevice and QNetd are required for this setup so that diskless SBD can handle split-brain scenarios for the two-node cluster.
- Failover of resources from one node to another if the active host breaks down (*active/passive* setup).

This is a simple cluster setup with minimal external requirements. You can use this cluster for testing purposes or as a basic cluster configuration that you can extend for a production environment later.

To install the High Availability cluster as described above, you must perform the following tasks:

1. Review *Section 2, "System requirements"* to make sure you have everything you need.
2. Install SUSE Linux Enterprise High Availability on the cluster nodes with *Section 3, "Enabling the High Availability extension"*.
3. Install QNetd on a non-cluster server with *Section 4, "Setting up the QNetd server"*.
4. Initialize the cluster on the first node with *Section 5, "Setting up the first node"*.
5. Add more nodes to the cluster with *Section 6, "Adding the second node"*.
6. Log in to the Hawk Web interface to monitor the cluster with *Section 7, "Logging in to Hawk"*.
7. Check the status of QDevice and QNetd with *Section 8, "Checking the QDevice and QNetd setup"*.
8. Perform basic tests to make sure the cluster works as expected with *Section 9, "Testing the cluster"*.
9. Review *Section 10, "Next steps"* for advice on extending the cluster for a production environment.

2 System requirements

This section describes the system requirements for a minimal setup of SUSE Linux Enterprise High Availability.

2.1 Hardware requirements

Servers

Three servers: two to act as cluster nodes, and one to run QNetd.

The servers can be bare metal or virtual machines. They do not require identical hardware (memory, disk space, etc.), but they must have the same architecture. Cross-platform clusters are not supported. See the *System Requirements* section at https://www.suse.com/download/sle-ha/ for more details about server hardware.

Network Interface Cards (NICs)

At least two NICs per cluster node. This allows you to configure two or more communication channels for the cluster, using one of the following methods:

- Combine the NICs into a network bond (preferred). In this case, you must set up the bonded device on each node before you initialize the cluster.
- Create a second communication channel in Corosync. This can be configured by the cluster setup script. In this case, the two NICs must be on different subnets.

Node fencing

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).

The minimal setup described in this guide uses a software watchdog and diskless SBD, so no additional hardware is required. Before using this cluster in a production environment, replace the software watchdog with a hardware watchdog.

2.2 Software requirements

Operating system

All nodes and the QNetd server must have SUSE Linux Enterprise Server installed and registered.

High Availability extension

The SUSE Linux Enterprise High Availability extension requires an additional registration code. This extension can be enabled during the SLES installation, or you can enable it later on a running system. This guide explains how to enable and register the extension on a running system.

2.3 Network requirements

Time synchronization

All systems must synchronize to an NTP server outside the cluster. SUSE Linux Enterprise Server uses `chrony` for NTP. When you initialize the cluster, you are warned if `chrony` is not running. Even if the nodes are synchronized, log files and cluster reports can still become difficult to analyze if the nodes have different time zones configured.

Host name and IP address

All cluster nodes must be able to find each other, and the QNetd server, by name. Use the following methods for reliable name resolution:

- Use static IP addresses.
- List all nodes in the `/etc/hosts` file with their IP address, FQDN and short host name.

Only the primary IP address on each NIC is supported.

SSH

All cluster nodes must be able to access each other, and the QNetd server, via SSH. Certain cluster operations also require passwordless SSH authentication. When you initialize the cluster, the setup script checks for existing SSH keys and generates them if they do not exist.

Important: `root` SSH access in SUSE Linux Enterprise 16

In SUSE Linux Enterprise 16, `root` SSH login with a password is disabled by default.

On each node, and the QNetd server, either create a user with **`sudo`** privileges or set up passwordless SSH authentication for the `root` user before you initialize the cluster.

If you initialize the cluster with a **`sudo`** user, certain `crmsh` commands also require passwordless **`sudo`** permission.

Separate network for QNetd

We recommend having the cluster nodes reach the QNetd server via a different network than the one Corosync uses. Ideally, the QNetd server should be in a separate rack from the cluster, or at least on a separate PSU and not in the same network segment as the Corosync communication channels.

3 Enabling the High Availability extension

This procedure explains how to install SUSE Linux Enterprise High Availability on an existing SUSE Linux Enterprise Server. You can skip this procedure if you already installed the High Availability extension and packages during the SLES installation with Agama.

PREREQUISITES

- SUSE Linux Enterprise Server is installed and registered with the SUSE Customer Center.
- You have an additional registration code for SUSE Linux Enterprise High Availability.

Perform this procedure on all the machines you intend to use as cluster nodes:

1. Log in either as the `root` user or as a user with **`sudo`** privileges.
2. Check whether the High Availability extension is already enabled:

```
> sudo SUSEConnect --list-extensions
```

3. Check whether the High Availability packages are already installed:

```
> zypper search ha_sles
```

4. Enable the SUSE Linux Enterprise High Availability extension:

```
> sudo SUSEConnect -p sle-ha/16.1/x86_64 -r HA_REGCODE
```

5. Install the High Availability packages:

```
> sudo zypper install -t pattern ha_sles
```

4 Setting up the QNetd server

QNetd is an arbitrator that provides a vote to the QDevice service running on the cluster nodes. The QNetd server runs outside the cluster, so you cannot move cluster resources to this server. QNetd can support multiple clusters if each cluster has a unique name.

By default, QNetd runs the `corosync-qnetd` daemon as the user `coroqnetd` in the group `coroqnetd`. This avoids running the daemon as `root`.

PREREQUISITES

- SUSE Linux Enterprise Server is installed and registered with the SUSE Customer Center.
- You have an additional registration code for SUSE Linux Enterprise High Availability.
- We recommend having the cluster nodes reach the QNetd server via a different network than the one Corosync uses.

Perform this procedure on a server that is *not* part of the cluster:

1. Log in either as the `root` user or as a user with **`sudo`** privileges.
2. Enable the SUSE Linux Enterprise High Availability extension:

   ```
   > sudo SUSEConnect -p sle-ha/16.0/x86_64 -r HA_REGCODE
   ```

3. Install the `corosync-qnetd` package:

   ```
   > sudo zypper install corosync-qnetd
   ```

   You do not need to manually start the `corosync-qnetd` service. It starts automatically when you configure QDevice in the cluster.

The QNetd server is ready to accept connections from a QDevice client (`corosync-qdevice`). Further configuration is handled by `crmsh` when you connect QDevice clients.

5 Setting up the first node

SUSE Linux Enterprise High Availability includes setup scripts to simplify the installation of a cluster. To set up the cluster on the first node, use the **`crm cluster init`** script.

5.1 Overview of the **`crm cluster init`** script

The **`crm cluster init`** command starts a script that defines the basic parameters needed for cluster communication, resulting in a running one-node cluster.

The script checks and configures the following components:

NTP

Checks if `chrony` is configured to start at boot time. If not, a message appears.

SSH

Detects or generates SSH keys for passwordless login between cluster nodes.

Firewall

Opens the ports in the firewall that are needed for cluster communication.

Corosync

Configures the cluster communication system.

SBD/watchdog

Checks if a watchdog exists and asks whether to configure SBD as the node fencing mechanism.

Hawk cluster administration

Enables the Hawk service and displays the URL for the Hawk Web interface.

QDevice/QNetd

Asks whether to configure QDevice and QNetd to participate in quorum decisions. This is recommended for clusters with an even number of nodes, and especially for two-node clusters.

Virtual floating IP

Asks whether to configure a virtual IP address for the Hawk Web interface.



Note: Pacemaker default settings

The options set by the **`crm cluster init`** script might not be the same as the Pacemaker default settings. You can check which settings the script changed in `/var/log/crmsh/crmsh.log`. Any options set during the bootstrap process can be modified later with `crmsh`.



Note: Cluster configuration for different platforms

The **`crm cluster init`** script detects the system environment (for example, Microsoft Azure) and adjusts certain cluster settings based on the profile for that environment. For more information, see the file `/etc/crm/profiles.yml`.

5.2 Initializing the cluster on the first node

Configure the cluster on the first node with the **`crm cluster init`** script. The script prompts you for basic information about the cluster and configures the required settings and services. For more information, run the **`crm cluster init --help`** command.

PREREQUISITES

- SUSE Linux Enterprise High Availability is installed and up to date.
- All nodes have at least two network interfaces or a network bond, with static IP addresses listed in the `/etc/hosts` file along with each node's FQDN and short host name.
- The QNetd server is installed. If you log in to the QNetd server as the `root` user, passwordless SSH authentication must be enabled.

Perform this procedure on only one node:

1. Log in to the first node either as the `root` user or as a user with **`sudo`** privileges.
2. Start the **`crm cluster init`** script:

   ```
   > sudo crm cluster init
   ```

   The script checks whether `chrony` is running, opens the required firewall ports and checks for SSH keys. If no SSH keys are available, the script generates them.
3. Configure Corosync for cluster communication:

   a. Enter an IP address for the first communication channel (`ring0`). By default, the script proposes the address of the first available network interface. This could be either an individual interface or a bonded device. Accept this address or enter a different one.

   b. If the script detects multiple network interfaces, it asks whether you want to configure a second communication channel (`ring1`). If you configured the first channel with a bonded device, you can decline with `n`. If you need to configure a second channel, confirm with `y` and enter the IP address of another network interface. The two interfaces must be on different subnets.

   The script configures the default firewall ports for Corosync communication.
4. Choose whether to set up SBD as the node fencing mechanism:

   a. Confirm with `y` that you want to use SBD.

   b. When prompted for a path to a block device, enter `none` to configure diskless SBD.

   The script configures SBD, including the relevant timeout settings. Unlike disk-based SBD, diskless SBD does not require a `fence_sbd` cluster resource.
   If no hardware watchdog is available, the script configures the software watchdog `softdog`.

5. Choose whether to configure QDevice and QNetd:

    a. Confirm with `y` that you want to configure QDevice and QNetd.

    b. Enter the IP address or host name of the QNetd server, with or without a user name.

        - If you include a non-`root` user name, a later step will prompt you for the user's password and the script will configure passwordless SSH authentication from the node to the QNetd server.
        - If you omit a user name, the script defaults to the `root` user, so passwordless SSH authentication must already be configured for the node to access the QNetd server.

    c. Enter the port to use for communication with the QNetd server. The default is `5403`.

    d. Choose the algorithm that determines how votes are assigned. The default is `ffsplit`.

    e. Choose the method to use when a tie-breaker is required. The default is `lowest`.

    f. Choose whether to enable TLS for client certificate checking. The default is `on` (attempt to connect with TLS, but connect without TLS if it isn't available).

    g. Enter heuristics commands to assist in quorum calculation, or leave the field blank to skip this step.

    h. If required, the script prompts you for the password of the QNetd server, then configures passwordless SSH authentication between the cluster nodes and the QNetd server.

    The script configures QDevice and QNetd, including SSH keys, the CA and server certificates, and the firewall port. It also enables the required services on the cluster nodes and on the QNetd server. The script creates a default cluster user and password, then starts the cluster services to bring the cluster online and enable Hawk.

6. Configure a virtual IP address for cluster administration with the Hawk Web interface:

    a. Confirm with `y` that you want to configure a virtual IP address.

    b. Enter an unused IP address to use as the administration IP for Hawk.

    Instead of logging in to Hawk on an individual cluster node, you can connect to the virtual IP address.

7. When the installation is complete, replace the default cluster user's password with a secure one:

```
> sudo passwd hacluster
```

8. Check the status of the cluster:

```
> sudo crm status
```

9. Check the status of SBD:

```
> sudo crm sbd status
```

6 Adding the second node

Add more nodes to the cluster with the **`crm cluster join`** script. The script only needs access to an existing cluster node and completes the basic setup on the current machine automatically. For more information, run the **`crm cluster join --help`** command.

PREREQUISITES

- SUSE Linux Enterprise High Availability is installed and up to date.
- An existing cluster is already running on at least one node.
- All nodes have at least two network interfaces or a network bond, with static IP addresses listed in the `/etc/hosts` file along with each node's FQDN and short host name.
- *If you log in as a* **`sudo`** *user:* The same user must exist on all nodes and the QNetd server. This user must have passwordless **`sudo`** permission.
- *If you log in as the* `root` *user:* Passwordless SSH authentication must be configured on all nodes and the QNetd server.

Perform this procedure on each additional node:

1. Log in to this node as the same user you set up the first node with.
2. Start the **`crm cluster join`** script:
   - If you set up the first node as `root`, you can start the script with no additional parameters:

     ```
     # crm cluster join
     ```

   - If you set up the first node as a **`sudo`** user, you must specify that user with the `-c` option:

     ```
     > sudo crm cluster join -c USER@NODE1
     ```

The script checks if `chrony` is running and opens the required firewall ports.

3. If you did not already specify the first node with `-c`, you are prompted for its IP address or host name.
4. If you did not already configure passwordless SSH authentication between the nodes, you are prompted for the password of the first node.
5. Configure Corosync for cluster communication:

   a. The script proposes an IP address for `ring0`. This IP address must be on the same subnet as the IP address used for `ring0` on the first node. If it is not, enter the correct IP address.

   b. If the cluster has two Corosync communication channels configured, the script prompts you for an IP address for `ring1`. This IP address must be on the same subnet as the IP address used for `ring1` on the first node.

   The script copies the cluster configuration from the first node, adjusts the timeout settings to consider the new node, and brings the new node online.
6. When the installation is complete, replace the default cluster user's password with a secure one:

```
> sudo passwd hacluster
```

7. Check the status of the cluster:

```
> sudo crm status
```

8. Check the status of SBD:

```
> sudo crm sbd status
```

7 Logging in to Hawk

Hawk allows you to manage a High Availability cluster using a graphical Web browser. You can also configure a floating, virtual IP address that allows clients to connect to Hawk no matter which node the Hawk service is running on.

PREREQUISITES

- The client machine must be able to connect to the cluster nodes.
- The client machine must have a graphical Web browser with JavaScript and cookies enabled.

This procedure uses the default `hacluster` user.

1. Open a Web browser and go to `https://HAWKSERVER:7630/`, replacing *HAWKSERVER* with the IP address or host name of a cluster node, or the Hawk virtual IP address if one is configured.

   > **Note: Certificate warning**
   >
   > If a certificate warning appears when you access the URL for the first time, a self-signed certificate is in use. To verify the certificate, ask your cluster operator for the certificate details. To proceed anyway, you can add an exception in the browser to bypass the warning.

2. On the Hawk login screen, enter the *Username* and *Password* of the `hacluster` user.
3. Click *Log In*. Hawk shows the *Status* screen by default.

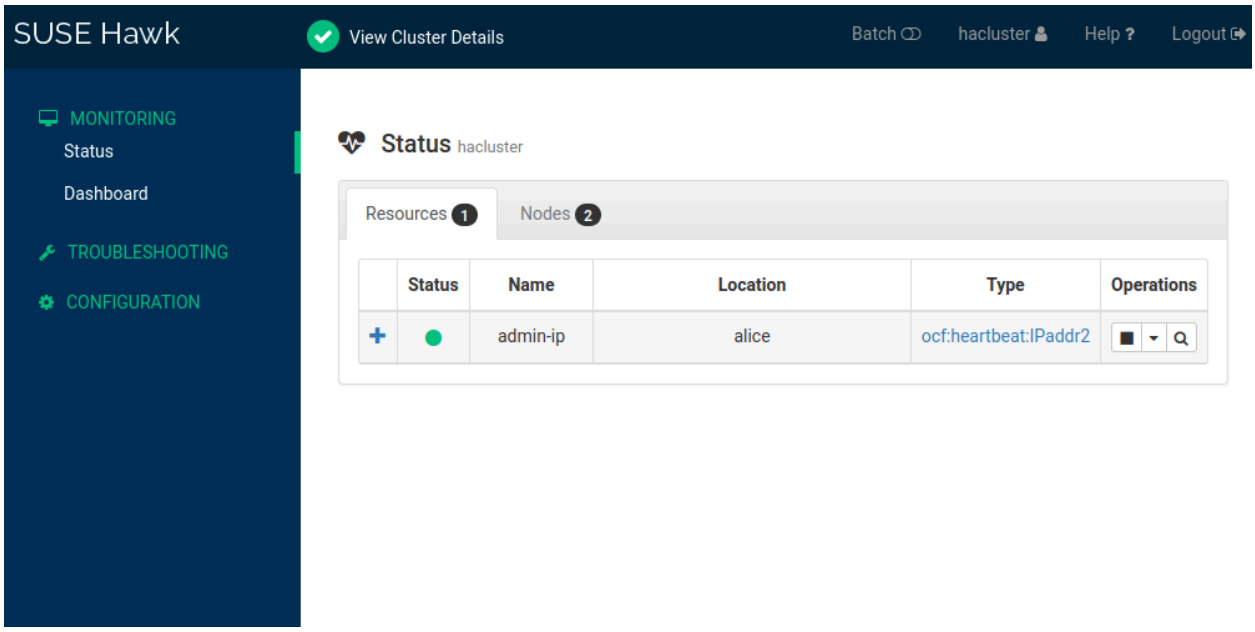


FIGURE 1: THE HAWK STATUS SCREEN

8 Checking the QDevice and QNetd setup

Use the **`crm corosync status`** command to check the cluster's quorum status and the status of QDevice and QNetd. You can run this command from any node in the cluster.

The following examples show a cluster with two nodes (`alice` and `bob`) and a QNetd server (`charlie`).

EXAMPLE 1: SHOWING THE CLUSTER'S QUORUM STATUS

```
> sudo crm corosync status quorum
1 alice member
2 bob member

Quorum information
------------------
Date:             [...]
Quorum provider:  corosync_votequorum
Nodes:            2
Node ID:          2
Ring ID:          1.e
Quorate:          Yes

Votequorum information
----------------------
Expected votes:   3
Highest expected: 3
Total votes:      3
Quorum:           2
Flags:            Quorate Qdevice

Membership information
----------------------
    Nodeid      Votes    Qdevice Name
         1          1    A,V,NMW alice
         2          1    A,V,NMW bob (local)
         0          1            Qdevice
```

The `Membership information` section shows the following status codes:

`A` (alive) or `NA` (not alive)

Shows the connectivity status between QDevice and Corosync.

`V` (vote) or `NV` (non vote)

Shows if the node has a vote. `V` means that both nodes can communicate with each other. In a split-brain scenario, one node would be set to `V` and the other node would be set to `NV`.

`MW` (master wins) or `NMW` (not master wins)

Shows if the `master_wins` flag is set. By default, the flag is not set, so the status is `NMW`.

`NR` (not registered)

Shows that the cluster is not using a quorum device.

EXAMPLE 2: SHOWING THE STATUS OF QDEVICE

```
> sudo crm corosync status qdevice
1 alice member
2 bob member

Qdevice information
-------------------
Model:                  Net
Node ID:                1
HB interval:            10000ms
Sync HB interval:       30000ms
Configured node list:
    0   Node ID = 1
    1   Node ID = 2
Heuristics:             Disabled
Ring ID:                1.e
Membership node list:   1, 2
Quorate:                Yes
Quorum node list:
    0   Node ID = 2, State = member
    1   Node ID = 1, State = member
Expected votes:         3
Last poll call:         [...]

Qdevice-net information
----------------------
Cluster name:           hacluster
QNetd host:             charlie:5403
Connect timeout:        8000ms
HB interval:            8000ms
VQ vote timer interval: 5000ms
TLS:                    Supported
Algorithm:              Fifty-Fifty split
Tie-breaker:            Node with lowest node ID
KAP Tie-breaker:        Enabled
Poll timer running:     Yes (cast vote)
State:                  Connected
TLS active:             Yes (client certificate sent)
Connected since:        [...]
Echo reply received:    [...]
```

EXAMPLE 3: SHOWING THE STATUS OF QNETD

```
> sudo crm corosync status qnetd
1 alice member
2 bob member
```

```
Cluster "hacluster":
    Algorithm:          Fifty-Fifty split (KAP Tie-breaker)
    Tie-breaker:        Node with lowest node ID
    Node ID 1:
        Client address:         ::ffff:192.168.1.185:45676
        HB interval:            8000ms
        Configured node list:   1, 2
        Ring ID:                1.e
        Membership node list:   1, 2
        Heuristics:             Undefined (membership: Undefined, regular: Undefined)
        TLS active:             Yes (client certificate verified)
        Vote:                   ACK (ACK)
    Node ID 2:
        Client address:         ::ffff:192.168.1.168:55034
        HB interval:            8000ms
        Configured node list:   1, 2
        Ring ID:                1.e
        Membership node list:   1, 2
        Heuristics:             Undefined (membership: Undefined, regular: Undefined)
        TLS active:             Yes (client certificate verified)
        Vote:                   No change (ACK)
```

9 Testing the cluster

The following tests can help you identify basic issues with the cluster setup. However, realistic tests involve specific use cases and scenarios. Before using the cluster in a production environment, test it thoroughly according to your use cases.

9.1 Testing resource failover

Check whether the cluster moves resources to another node if the current node is set to `standby`. This procedure uses example nodes called `alice` and `bob`, and a virtual IP resource called `admin-ip` with the example IP address `192.168.1.10`.

1. Open two terminals.
2. In the first terminal, ping the virtual IP address:

```
> ping 192.168.1.10
```

3. In the second terminal, log in to one of the cluster nodes.
4. Check which node the virtual IP address is running on:

```
> sudo crm status
[..]
Node List:
  * Online: [ alice bob ]

Full List of Resources:
  * admin-ip  (ocf:heartbeat:IPaddr2):    Started alice
```

5. Put `alice` into standby mode:

```
> sudo crm node standby alice
```

6. Check the cluster status again. The resource `admin-ip` should have migrated to `bob`:

```
> sudo crm status
[...]
Node List:
  * Node alice: standby
  * Online: [ bob ]

Full List of Resources:
  * admin-ip  (ocf:heartbeat:IPaddr2):    Started bob
```

7. In the first terminal, you should see an uninterrupted flow of pings to the virtual IP address during the migration. This shows that the cluster setup and the floating IP address work correctly.
8. Cancel the **`ping`** command with `Ctrl` – `C` .
9. In the second terminal, bring `alice` back online:

```
> sudo crm node online alice
```

9.2 Testing cluster failures

The **`crm cluster crash_test`** command simulates cluster failures and reports the results.

The command supports the following checks:

`--fence-node` *`NODE`*

Fences a specific node passed from the command line.

`--kill-sbd`/`--kill-corosync`/ `--kill-pacemakerd`

Kills the daemons for SBD, Corosync, or Pacemaker. After running one of these tests, you can find a report in the directory `/var/lib/crmsh/crash_test/`. The report includes a test case description, action logging, and an explanation of possible results.

`--split-brain-iptables`

Simulates a split-brain scenario by blocking the Corosync port, and checks whether one node can be fenced as expected. You must install `iptables` before you can run this test.

For more information, run the **`crm cluster crash_test --help`** command.

This example uses nodes called `alice` and `bob`, and tests fencing `bob`. To watch `bob` change status during the test, you can log in to Hawk and navigate to *Status* › *Nodes*, or run **`crm status`** from another node.

EXAMPLE 4: MANUALLY TRIGGERING NODE FENCING

```
admin@alice> sudo crm cluster crash_test --fence-node bob

==========================================
Testcase:          Fence node bob
Fence action:      reboot
Fence timeout:     95

!!! WARNING WARNING WARNING !!!
THIS CASE MAY LEAD TO NODE BE FENCED.
TYPE Yes TO CONTINUE, OTHER INPUTS WILL CANCEL THIS CASE [Yes/No](No): Yes
INFO: Trying to fence node "bob"
INFO: Waiting 95s for node "bob" reboot...
INFO: Node "bob" will be fenced by "alice"!
INFO: Node "bob" was fenced by "alice" at DATE TIME
```

10 Next steps

This guide describes a basic High Availability cluster that can be used for testing purposes. To expand this cluster for use in production environments, more steps are recommended:

Adding more nodes

If required, add more nodes to the cluster using the **`crm cluster join`** script.

Enabling a hardware watchdog

Before using the cluster in a production environment, replace `softdog` with a hardware watchdog.

To configure a new watchdog device, see Setting up the SBD watchdog (https://documentation.suse.com/sle-ha/16.1/html/HA-sbd-configuring-diskbased/index.html#ha-sbd-setting-up-watchdog).

To replace `softdog` with a hardware watchdog, see Changing the SBD watchdog device (https://documentation.suse.com/sle-ha/16.1/html/HA-sbd-changing-configuration/index.html#ha-sbd-changing-watchdog-device).

Adding more fencing devices

For critical workloads, we highly recommend having two or three fencing devices, using either physical devices or disk-based SBD.

To switch to disk-based SBD, see Changing diskless SBD to disk-based SBD (https://documentation.suse.com/sle-ha/16.1/html/HA-sbd-changing-configuration/index.html#ha-sbd-changing-diskless-to-diskbased).

To add a physical fencing device, see Configuring Node Fencing in a High Availability Cluster (https://documentation.suse.com/sle-ha/16.1/html/HA-fencing-configuring/).

11 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.