Oracle® Linux 8

Setting Up High Availability Clustering
Abstract

Oracle® Linux 8: Setting Up High Availability Clustering describes tasks for configuring a high availability (HA) cluster using the Pacemaker and Corosync software packages.
# Table of Contents

Preface ................................................................................................................................. v
1 About High Availability Clustering .................................................................................. 1
2 Installing and Configuring Pacemaker and Corosync ...................................................... 3
   2.1 Enabling Access to the Pacemaker and Corosync Packages .................................... 3
   2.2 Installing and Enabling the Pacemaker and Corosync Service ................................ 4
3 Configuring an Initial Cluster and Service ...................................................................... 5
   3.1 Creating the Cluster ................................................................................................. 5
   3.2 Setting Cluster Parameters ..................................................................................... 6
   3.3 Creating a Service and Testing Failover ................................................................... 7
4 Configuring Fencing (stonith) .......................................................................................... 9
   4.1 About Fencing Configuration (stonith) ..................................................................... 9
   4.2 Fencing Configuration Examples ............................................................................. 9
   4.3 Configuring Fencing Levels ..................................................................................... 12
5 Using the Pacemaker/Corosync Web User Interface ....................................................... 13
   5.1 About the Pacemaker Web User Interface ................................................................ 13
   5.2 Initial Cluster Configuration Tasks .......................................................................... 14
      5.2.1 Creating a New Cluster .................................................................................... 14
      5.2.2 Adding an Existing Cluster .............................................................................. 15
      5.2.3 Removing or Destroying Clusters .................................................................... 16
   5.3 Managing Clusters With the Web UI ....................................................................... 16
      5.3.1 Configuring Nodes ........................................................................................... 16
      5.3.2 Configuring Additional Cluster Properties ....................................................... 17
      5.3.3 Setting Local Permissions Through ACLs ....................................................... 18
      5.3.4 Configuring Fencing ....................................................................................... 19
      5.3.5 Adding Resources to the Cluster .................................................................... 20
6 More Information .............................................................................................................. 23
Preface

*Oracle® Linux 8: Setting Up High Availability Clustering* describes how to install and configure high availability clustering in Oracle Linux using Corosync and Pacemaker, which are tools that enable you to achieve high availability for applications and services that are running on Oracle Linux.

**Audience**

This document is intended for administrators who need to configure and administer Oracle Linux networking. It is assumed that readers are familiar with web technologies and have a general understanding of using the Linux operating system, including knowledge of how to use a text editor such as *emacs* or *vim*, essential commands such as `cd`, `chmod`, `chown`, `ls`, `mkdir`, `mv`, `ps`, `pwd`, and `rm`, and using the *man* command to view manual pages.

**Document Organization**

The document is organized into the following chapters:

- **Chapter 1, About High Availability Clustering** provides an overview of high availability (HA) configuration.
- **Chapter 2, Installing and Configuring Pacemaker and Corosync** describes how to install and configure Pacemaker and Corosync.
- **Chapter 3, Configuring an Initial Cluster and Service** describes how to configure an initial cluster and enable related services.
- **Chapter 4, Configuring Fencing (stonith)** describes how to configure fencing and provides examples of various fencing configurations.
- **Chapter 5, Using the Pacemaker/Corosync Web User Interface** describes how to use the web user interface to configure and manage clusters.
- **Chapter 6, More Information** provides a link to additional information about Pacemaker and Corosync.

**Related Documents**

The documentation for this product is available at:

*Oracle® Linux 8 Documentation*

**Conventions**

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><strong>italic</strong></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td><strong>monospace</strong></td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
https://www.oracle.com/corporate/accessibility/.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit
https://www.oracle.com/corporate/accessibility/learning-support.html#support-tab.
Chapter 1 About High Availability Clustering

This chapter describes how to set up and configure the Pacemaker and Corosync technologies to create a high availability (HA) cluster that delivers continuous access to services that are running across multiple nodes.

High availability services in Oracle Linux are comprised of several open-source packages, including the Corosync and Pacemaker features. These tools enable you to achieve high availability for applications and services that are running on Oracle Linux. You can download Corosync, Pacemaker, and the functional sub-packages from the Unbreakable Linux Network (ULN) at https://linux.oracle.com or the Oracle Linux yum server Oracle Linux at https://yum.oracle.com.

Corosync is an open source cluster engine that includes an API to implement a number of high availability features, including an availability manager that can restart a process when it fails, a configuration and statistics database, and a quorum system that can notify applications when quorum is achieved or lost.

Corosync is installed in conjunction with Pacemaker, which is an open source high availability cluster resource manager that is responsible for managing the life cycle of software that is deployed on a cluster. Pacemaker also provides high availability services, which are achieved by detecting and recovering from node and resource-level failures by using the API that is provided by the cluster engine.

Pacemaker also ships with the Pacemaker Command Shell (pcs). You can use the pcs command to access and configure the cluster and its resources. The pcs daemon runs as a service on each node in the cluster, making it possible to synchronize configuration changes across all of the nodes in the cluster.

Oracle provides support for Corosync and Pacemaker that is used for an active-passive 2-node (1:1) cluster configuration on Oracle Linux 8. Note that support for clustering services does not imply support for Oracle products that are clustered by using these services.
Chapter 2 Installing and Configuring Pacemaker and Corosync

Table of Contents

2.1 Enabling Access to the Pacemaker and Corosync Packages .......................................................... 3
2.2 Installing and Enabling the Pacemaker and Corosync Service ...................................................... 4

This chapter describes how to set up and configure the Pacemaker and Corosync features to create a high availability (HA) cluster that delivers continuous access to services running across multiple nodes.

2.1 Enabling Access to the Pacemaker and Corosync Packages

The Pacemaker and Corosync packages are available on the Oracle Linux yum server in the `ol8_addons` repository, or on the Unbreakable Linux Network (ULN) in the `ol8_arch_addons` channel.

Some dependency packages may be required from the `ol8_appstream` and `ol8_baseos_latest` yum repositories, or from the `ol8_arch_appstream` and `ol8_arch_baseos_latest` channels on ULN.

Enabling Repositories with ULN

If you are registered to use ULN, use the ULN web interface to subscribe the system to the appropriate channels.

To subscribe to the ULN channels:

1. Log in to `https://linux.oracle.com` with your ULN user name and password.
2. On the Systems tab, click the link named for the system in the list of registered machines.
3. On the System Details page, click Manage Subscriptions.
4. On the System Summary page, select each required channel from the list of available channels and click the right arrow to move the channel to the list of subscribed channels. Subscribe the system to the following channels:
   - `ol8_arch_appstream`
   - `ol8_arch_baseos_latest`
   - `ol8_arch_addons`
5. Click Save Subscriptions.

Enabling Repositories with the Oracle Linux Yum Server

If you are using the Oracle Linux yum server for system updates, enable the Gluster Storage for Oracle Linux yum repository.

To enable the yum repositories:

1. Enable the following yum repositories:
   - `ol8_appstream`
   - `ol8_baseos_latest`
• ol8_addons

Use the `dnf config-manager` tool to enable the yum repositories:

```
# dnf config-manager --enable ol8_appstream ol8_baseos_latest ol8_addons
```

## 2.2 Installing and Enabling the Pacemaker and Corosync Service

On each node in the cluster, install the `pcs` and `pacemaker` software packages, along with all of the available resource and fence agents from the Oracle Linux yum server or from ULN, for example:

```
# dnf install pcs pacemaker resource-agents fence-agents-all
```

If you are running `firewalld`, add the `high-availability` service on each of the nodes so that the service components are able to communicate across the network. Per the command that is run in the following example, this step typically enables the following ports: TCP port 2224 (used by the `pcs` daemon), port 3121 (for Pacemaker Remote nodes), port 21064 (for DLM resources), and UDP ports 5405 (for Corosync clustering), and 5404 (for Corosync multicast, if configured):

```
# firewall-cmd --permanent --add-service=high-availability
# firewall-cmd --add-service=high-availability
```

To use the `pcs` command to configure and manage your cluster, you must set a password on each node for the `hacluster` user.

---

**Tip**

It is helpful if you set the same password for this user as the password you set for the user on each node.

Use the `passwd` command on each node to set the password:

```
# passwd hacluster
```

Note that to use the `pcs` command, the `pcsd` service must be running on each of the nodes in the cluster. You can set this service to run and to start at boot by running the following command:

```
# systemctl enable --now pcsd.service
```
Chapter 3 Configuring an Initial Cluster and Service

Table of Contents

3.1 Creating the Cluster ................................................................. 5
3.2 Setting Cluster Parameters .......................................................... 6
3.3 Creating a Service and Testing Failover .............................................. 7

This chapter provides an example, along with step-by-step instructions on configuring an initial cluster across two nodes that are hosted on systems with the resolvable host names node1 and node2. Each system is installed and configured by using the instructions that are provided in Chapter 2, Installing and Configuring Pacemaker and Corosync.

The cluster is configured to run a service, Dummy, that is included in the resource-agents package. You should have installed this package along with the pacemaker packages. This tool simply keeps track of whether the service is or is not running. Pacemaker is configured with an interval parameter that determines how long it should wait between checks to determine whether the Dummy process has failed.

The Dummy process is manually stopped outside of the Pacemaker tool to simulate a failure, which is used to demonstrate how the process is restarted automatically on an alternate node.

3.1 Creating the Cluster

To create the cluster:

1. Authenticate the pcs cluster configuration tool for the hacluster user on each node in your configuration by running the following command on one of the nodes that will form part of the cluster:

   ```
   # pcs host auth node1 node2 -u hacluster
   ```

   Replace node1 and node2 with the resolvable hostnames of the nodes that will form part of the cluster.

   Alternately, if the node names are not resolvable, specify the IP addresses where the nodes can be accessed, as shown in the following example:

   ```
   # pcs host auth node1 addr=192.0.2.1 node2 addr=192.0.2.2 -u hacluster
   ```

   Replace 192.0.2.1 and 192.0.2.2 with the IP addresses of each of the respective hosts in the cluster.

   The tool prompts you to provide a password for the hacluster user. Provide the password that you set for this user when you installed and configured the Pacemaker software on each node.

2. Create the cluster by using the pcs cluster setup command. You must specify a name for the cluster and the node names and IP addresses for each node in the cluster. For example, run the following command:

   ```
   # pcs cluster setup pacemaker1 node1 addr=192.0.2.1 node2 addr=192.0.2.2
   ```

   Replace pacemaker1 with an appropriate name for the cluster. Replace node1 and node2 with the resolvable hostnames of the nodes in the cluster. Replace 192.0.2.1 and 192.0.2.2 with the IP addresses of each of the respective hosts in the cluster.

   Note that if you used the addr option to specify the IP addresses when authenticated the nodes, you do not need to specify them again when running the pcs cluster setup command.
The cluster setup process destroys any existing cluster configuration on the specified nodes and creates a configuration file for the Corosync service that is copied to each of the nodes within the cluster.

You can, optionally, use the `--start` option when running the `pcs cluster setup` command to automatically start the cluster once it is created.

3. If you have not already started the cluster as part of the cluster setup command, start the cluster on all of the nodes. To start the cluster manually, use the `pcs` command:

   ```
   # pcs cluster start --all
   ```

   Starting the `pacemaker` service from `systemd` is another way to start the cluster on all nodes, for example:

   ```
   # systemctl start pacemaker.service
   ```

4. Optionally, you can enable these services to start at boot time so that if a node reboots, it automatically rejoins the cluster, for example:

   ```
   # pcs cluster enable --all
   ```

   Alternately you can enable the `pacemaker` service from `systemd` on all nodes, for example:

   ```
   # systemctl enable pacemaker.service
   ```

   **Note**

   Some users prefer not to enable these services so that a node failure resulting in a full system reboot can be properly debugged before it rejoins the cluster.

### 3.2 Setting Cluster Parameters

Fencing is an important part of setting up a production-level HA cluster. For simplicity, it is disabled in this example. If you intend to take advantage of stonith, see Section 4.1, “About Fencing Configuration (stonith)” for additional information.

To set cluster parameters:

1. Disable the fencing feature by running the following command:

   ```
   # pcs property set stonith-enabled=false
   ```

   Fencing is an advanced feature that helps protect your data from being corrupted by nodes that might be failing or are unavailable. Pacemaker uses the term `stonith` (shoot the other node in the head) to describe fencing options. This configuration depends on particular hardware and a deeper understanding of the fencing process. For this reason, it is recommended that you disable the fencing feature.

2. Optionally, configure the cluster to ignore the quorum state by running the following command:

   ```
   # pcs property set no-quorum-policy=ignore
   ```

   Because this example uses a two-node cluster, disabling the no-quorum policy makes the most sense, as quorum technically requires a minimum of three nodes to be a viable configuration. Quorum is only achieved when more than half of the nodes agree on the status of the cluster.
Creating a Service and Testing Failover

In the current release of Corosync, this issue is treated specially for two-node clusters, where the quorum value is artificially set to 1 so that the master node is always considered in quorum. In the case where a network outage results in both nodes going offline for a period, the nodes race to fence each other and the first to succeed wins quorum. The fencing agent can usually be configured to give one node priority so that it is more likely to win quorum if this is preferred.

3. Configure a migration policy by running the following command:

```
# pcs resource defaults migration-threshold=1
```

Running this command configures the cluster to move the service to a new node after a single failure.

### 3.3 Creating a Service and Testing Failover

To create a service and test failover:

Services are created and usually configured to run a resource agent that is responsible for starting and stopping processes. Most resource agents are created according to the OCF (Open Cluster Framework) specification, which is defined as an extension for the Linux Standard Base (LSB). There are many handy resource agents for commonly used processes that are included in the `resource-agents` packages, including a variety of heartbeat agents that track whether commonly used daemons or services are still running.

In the following example, a service is set up that uses a Dummy resource agent created precisely for the purpose of testing Pacemaker. This agent is used because it requires a very basic configuration and does not make any assumptions about your environment or the types of services that you intend to run with Pacemaker.

1. Add the service as a resource by using the `pcs resource create` command:

```
# pcs resource create dummy_service ocf:pacemaker:Dummy op monitor interval=120s
```

In the previous example, `dummy_service` is the name that is provided for the service for this resource:

To invoke the Dummy resource agent, a notation (`ocf:pacemaker:Dummy`) is used to specify that it conforms to the OCF standard, that it runs in the pacemaker namespace, and that the Dummy script should be used. If you were configuring a heartbeat monitor service for an Oracle Database, you might use the `ocf:heartbeat:oracle` resource agent.

The resource is configured to use the monitor operation in the agent and an interval is set to check the health of the service. In this example, the interval is set to 120s to give the service sufficient time to fail while you are demonstrating failover. By default, this interval is usually set to 20 seconds, but it can be modified depending on the type of service and your particular environment.

When you create a service, the cluster attempts to start the resource on a node by using the resource agent’s start command.

2. View the resource start and run status:

```
# pcs status
Cluster name: pacemaker1
Stack: corosync
Current DC: node2 (version 2.0.2-3.e18_1.2-744a30d655) - partition with quorum
Last updated: Fri Jan 24 04:56:27 2020
Last change: Fri Jan 24 04:56:11 2020 by root via cibadmin on node1
2 nodes configured
```
Creating a Service and Testing Failover

1 resource configured
Online: [ node1 node2 ]

Full list of resources:

dummy_service (ocf::pacemaker:Dummy): Started node1

Daemon Status:
corosync: active/disabled
pacemaker: active/disabled
pcsd: active/enabled

3. Run the `crm_resource` command to simulate service failure by force stopping the service directly:

```
# crm_resource --resource dummy_service --force-stop
```

Running the `crm_resource` command ensures that the cluster is unaware that the service has been manually stopped.

4. Run the `crm_mon` command in interactive mode so that you can wait until a node fails, to view the **Failed Actions** message:

```
# crm_mon
Stack: corosync
Current DC: node1 (version 2.0.2-3.el8_1.2-744a30d655) - partition with quorum
Last updated: Fri Jan 24 05:00:27 2020
Last change: Fri Jan 24 04:56:11 2020 by root via cibadmin on node1

3 nodes configured
1 resource configured

Online: [ node1 node2 ]

Active resources:

dummy_service (ocf::pacemaker:Dummy): Started node2

Failed Resource Actions:
* dummy_service_monitor_120000 on node1 'not running' (7): call=7, status=complete, exitreason='', last-rc-change='Fri Jan 24 05:00:11 2020', queued=0ms, exec=0ms

You should see the service restart on the alternate node. Note that the default monitor period is set to 120 seconds, so you may have to wait up to the full period before you see notification that a node has gone offline.

Tip
You can use the `Ctrl-C` key combination to exit out of `crm_mon` at any point.

5. If necessary, reboot the node where the service is running to determine whether failover also occurs in the case of node failure.

Note that if you did not previously enable the `corosync` and `pacemaker` services to start on boot, you might need to manually start the services on the node that you rebooted by running the following command:

```
# pcs cluster start node1
```
Chapter 4 Configuring Fencing (stonith)

Table of Contents

4.1 About Fencing Configuration (stonith) .............................................................. 9
4.2 Fencing Configuration Examples ...................................................................... 9
4.3 Configuring Fencing Levels ................................................................................ 12

This chapter describes how to configure fencing (stonith).

4.1 About Fencing Configuration (stonith)

Fencing, or stonith (shoot the other node in the head), is used to protect data in the event that nodes become unresponsive. If a node fails to respond, it may still be accessing data. To ensure that your data is safe, you can use fencing to prevent a live node from accessing data until the original node is truly offline. To accomplish this task, you must configure a device that can ensure a node is taken offline. There are a number of available fencing agents that can be configured for this purpose. In general, stonith relies on particular hardware and service protocols that can force reboot or shutdown nodes physically to protect the cluster.

The following are different configurations that use some of the available fencing agents. Note that these examples make certain presumptions about hardware and assume that you already know how to set up, configure, and use the affected hardware. The following examples are provided for basic guidance only. It is recommended that you also refer to upstream documentation to familiarize yourself with some of the concepts that are presented in this documentation.

Before proceeding with any of the following configurations, ensure that stonith is enabled for your cluster configuration:

```bash
# pcs property set stonith-enabled=true
```

After configuring stonith, run the following commands to check your configuration and ensure that it is set up correctly:

```bash
# pcs stonith config
# pcs cluster verify --full
```

To check the status of your stonith configuration, run the following command:

```bash
# pcs stonith
```

To check the status of your cluster, run the following command:

```bash
# pcs status
```

4.2 Fencing Configuration Examples

The following examples describe that various types of fencing configurations that you can implement.

IPMI LAN Fencing

Intelligent Platform Management Interface (IPMI) is an interface to a subsystem that provides management features of the host system's hardware and firmware and includes facilities to power cycle a system over a dedicated network without any requirement to access the system's operating system. You can configure the fence_ipmilan fencing agent for the cluster so that stonith can be achieved across the IPMI LAN.
If your systems are configured for IPMI, you can run the following commands on one of the nodes in the cluster to enable the ipmilan fencing agent and configure stonith for both nodes, for example:

```bash
# pcs stonith create ipmilan_n1_fencing fence_ipmilan pcmk_host_list=node1 delay=5 \
  ipaddr=203.0.113.1 login=root passwd=password lanplus=1 op monitor interval=60s
# pcs stonith create ipmilan_n2_fencing fence_ipmilan pcmk_host_list=node2 \
  ipaddr=203.0.113.2 login=root passwd=password lanplus=1 op monitor interval=60s
```

In the example, `node1` is a host that has an IPMI LAN interface configured on the IP address `203.0.113.1`. The host named `node2` has an IPMI LAN interface that is configured on the IP `203.0.113.2`. The root user password for the IPMI login on both systems is specified in this example as `password`. In each instance. You should replace these configuration variables with the appropriate values for your particular environment.

Note that the delay option should only be set to one node. This setting ensures that in the rare case of a fence race condition that only one node is killed and the other continues to run. Without this option set, it is possible that both nodes make the assumption that they are the only surviving node and then simultaneously reset each other.

### Warning

The IPMI LAN agent exposes the login credentials of the IPMI subsystem in plain text. Your security policy should ensure that it is acceptable for users with access to the Pacemaker configuration and tools to also have access to these credentials and the underlying subsystems that are involved.

---

**SCSI Fencing**

The SCSI Fencing agent is used to provide storage-level fencing. This configuration protects storage resources from being written to by two nodes simultaneously by using SCSI-3 PR (Persistent Reservation). Used in conjunction with a watchdog service, a node can be reset automatically by using stonith when it attempts to access the SCSI resource without a reservation.

To configure an environment in this way:

1. Install the watchdog service on both nodes and then copy the provided `fence_scsi_check` script to the watchdog configuration before enabling the service, as shown in the following example:

   ```bash
   # dnf install watchdog
   # cp /usr/share/cluster/fence_scsi_check /etc/watchdog.d/
   # systemctl enable --now watchdog
   ```

2. Enable the `iscsid` service that is provided in the `iscsi-initiator-utils` package on both nodes:

   ```bash
   # dnf install -y iscsi-initiator-utils
   # systemctl enable --now iscsid
   ```

3. After both nodes are configured with the watchdog service and the `iscsid` service, you can configure the `fence_scsi` fencing agent on one of the cluster nodes to monitor a shared storage device, such as an iSCSI target, for example:

   ```bash
   # pcs stonith create scsi_fencing fence_scsi pcmk_host_list="node1 node2" \
   devices="/dev/sdb" meta provides="unfencing"
   ```

   In the example, `node1` and `node2` represent the hostnames of the nodes in the cluster and `/dev/sdb` is the shared storage device. Replace these variables with the appropriate values for your particular environment.
SBD Fencing

The Storage Based Death (SBD) daemon can run on a system and monitor shared storage. The SBD daemon can use a messaging system to track cluster health. SBD can also trigger a reset if the appropriate fencing agent determines that *stonith* should be implemented.

To set up and configure SBD fencing:

1. Stop the cluster by running the following command on one of the nodes:
   ```bash
   # pcs cluster stop --all
   ```

2. On each node, install and configure the SBD daemon:
   ```bash
   # dnf install sbd
   ```

3. Enable the *sbd* systemd service:
   ```bash
   # systemctl enable sbd
   ```
   Note that the *sbd* systemd service is automatically started and stopped as a dependency of the *pacemaker* service, you do not need to run this service independently. Attempting to start or stop the *sbd* systemd service fails and returns an error indicating that it is controlled as a dependency service.

4. Edit the `/etc/sysconfig/sbd` file and set the `SBD_DEVICE` parameter to identify the shared storage device. For example, if your shared storage device is available on `/dev/sdc`, make sure the file contains the following line:
   ```text
   SBD_DEVICE="/dev/sdc"
   ```

5. On one of the nodes, create the SDB messaging layout on the shared storage device and confirm that it is in place. For example, to set up and verify messaging on the shared storage device at `/dev/sdc`, run the following commands:
   ```bash
   # sbd -d /dev/sdc create
   # sbd -d /dev/sdc list
   ```

6. Finally, start the cluster and configure the `fence_sbd` fencing agent for the shared storage device. For example, to configure the shared storage device, `/dev/sdc`, run the following commands on one of the nodes:
   ```bash
   # pcs cluster start --all
   # pcs stonith create sbd_fencing fence_sbd devices=/dev/sdc
   ```

IF-MIB Fencing

IF-MIB fencing takes advantage of SNMP to access the IF-MIB on an Ethernet network switch and to also shut down the port on the switch, which effectively takes a host offline. This configuration leaves the host running, while disconnecting it from the network. It is worth bearing in mind that any FibreChannel or InfiniBand connections could remain intact, even after the Ethernet connection has been terminated, which means that any data made available on these connections could still be at risk. As a result, it is best to configure this fencing method as a fallback fencing mechanism. See *Section 4.3, “Configuring Fencing Levels”* for more information about how to use multiple fencing agents in combination to maximize *stonith* success.

To configure IF-MIB fencing:
1. Configure your switch for SNMP v2c, at minimum, and make sure that SNMP SET messages are enabled. For example, on an Oracle Switch, by using the ILOM CLI, you could run the following commands:

```bash
# set /SP/services/snmp/ sets=enabled
# set /SP/services/snmp/ v2c=enabled
```

2. On one of the nodes in your cluster, configure the `fence_ifmib` fencing agent for each node in your environment, as shown in the following example:

```bash
# pcs stonith create ifmib_n1_fencing fence_ifmib pcmk_host_list=node1 \  ipaddr=203.0.113.10 community=private port=1 delay=5 op monitor interval=60s
# pcs stonith create ifmib_n2_fencing fence_ifmib pcmk_host_list=node2 \  ipaddr=203.0.113.10 community=private port=2 op monitor interval=60s
```

In the example, the SNMP IF-MIB switch is accessible at the IP address `203.0.113.10`; the `node1` host is connected to port 1 on the switch, and the `node2` host is connected to port 2 on the switch. You should replace these variables with the appropriate values for your particular environment.

### 4.3 Configuring Fencing Levels

If you have configured multiple fencing agents, you may want to set different fencing levels. Fencing levels enable you to prioritize different approaches to fencing and can provide a valuable mechanism for fallback options should your default fencing mechanism fail.

Each fencing level is attempted in ascending order, starting from level 1. If the fencing agent that is configured for a particular level fails, the fencing agent from the next level is then attempted, and so on.

For example, you may wish to configure IPMI-LAN fencing at level 1, but fallback to IF-MIB fencing as a level 2 option. Using the example configurations from Section 4.2, “Fencing Configuration Examples”, you would run the following commands on one of the nodes to set the fencing levels for each configured agent:

```bash
# pcs stonith level add 1 node1 ipmilan_n1_fencing
# pcs stonith level add 1 node2 ipmilan_n2_fencing
# pcs stonith level add 2 node1 ifmib_n1_fencing
# pcs stonith level add 2 node2 ifmib_n2_fencing
```
Chapter 5 Using the Pacemaker/Corosync Web User Interface

Table of Contents

5.1 About the Pacemaker Web User Interface ................................................................. 13
5.2 Initial Cluster Configuration Tasks ................................................................. 14
  5.2.1 Creating a New Cluster .................................................................................. 14
  5.2.2 Adding an Existing Cluster .......................................................................... 15
  5.2.3 Removing or Destroying Clusters .................................................................. 16
5.3 Managing Clusters With the Web UI ................................................................. 16
  5.3.1 Configuring Nodes ......................................................................................... 16
  5.3.2 Configuring Additional Cluster Properties .................................................... 17
  5.3.3 Setting Local Permissions Through ACLs .................................................... 18
  5.3.4 Configuring Fencing ....................................................................................... 19
  5.3.5 Adding Resources to the Cluster .................................................................... 20

This chapter describes how to create and manage clusters by using the web UI tool instead of the pcs command line.

5.1 About the Pacemaker Web User Interface

The Pacemaker service provides a web user interface tool (web UI) that enables you to configure and manage clusters in graphical mode. Use this tool as an alternative to typing pcs commands to perform those tasks.

This chapter assumes that you have completed the tasks that are described in Section 2.2, “Installing and Enabling the Pacemaker and Corosync Service” and that the nodes have been authenticated for the hacluser user.

For information about authentication and configuring hacluser credentials, see Step 1 of Section 3.1, “Creating the Cluster”.

To access the web UI, log in as user hacluser at https://node:2224, where node refers to a node authenticated for hacluser. Specify the node either by its node name or IP address.

Note
The rest of this chapter assumes that you have configured resolvable names for all of your nodes.

After you log in, the home page's Manage Clusters page is displayed. This page lists clusters that are currently under the web UI's management.

Figure 5.1 Manage Clusters

![Manage Clusters](image)
5.2 Initial Cluster Configuration Tasks

The Manage Clusters page contains the following options:

- **Create New**: Create a cluster consisting of nodes. While creating the cluster, you can optionally configure additional cluster properties.
- **Add Existing**: Add more nodes to an existing cluster.
- **Remove**: Remove a cluster from being managed by the web UI.
- **Destroy**: Delete a cluster.

5.2.1 Creating a New Cluster

To create new clusters, do the following:

1. Click **Create New**.
   
   The Create cluster window opens.

2. Specify the cluster name.

3. Specify the nodes to include in the cluster.

4. Optional: Configure other properties of the cluster.
   
   a. Click **Go to advanced settings**.
   
   b. Click the tab for the setting you want to customize.
      
      You can configure the cluster's Transport, Quorum, or Totem settings.
   
   c. For each setting, specify parameter values on their corresponding fields.

   For example, for the Transport mechanism, you can define transport, compression, and crypto options, as shown in the following image:
Figure 5.2 Transport Options

5. Click **Create Cluster**.

6. Click **Finish**.

   If you do not want to start the cluster, deselect **Start the Cluster** first before clicking **Finish**.

The new cluster is listed on the page. Selecting it displays cluster information such as its nodes and any configured resources and fences.

### 5.2.2 Adding an Existing Cluster

This option enables you to add nodes to an existing cluster.

1. From the list, select the cluster to which you want to add an existing node.

   If the cluster is unlisted, then you would need to create the cluster first.

2. Click **+ Add Existing**.

3. Specify the node that you want to add.

   You can add only one node at a time.

4. Click **Add Existing**.
Removing or Destroying Clusters

Note
The web UI provides another method of adding nodes to a cluster. See Section 5.3.1, “Configuring Nodes”.

5.2.3 Removing or Destroying Clusters

The remaining options on the Manage Cluster page are removing clusters or destroying them.

- Removing a cluster means disabling it from being managed by the web UI. The cluster continues to run. However, it can only be administered through the command line.
- Destroying a cluster means deleting the cluster and all its associated configuration files. Its constituent nodes thus become independent units. Destroying a cluster is irreversible.

5.3 Managing Clusters With the Web UI

To further customize existing clusters, from the list of clusters that is displayed, click the cluster name. A new window opens where additional menu options enable you to configure the cluster's nodes, resources, fence devices, access control lists, and cluster properties. By default, the Nodes page is displayed.

Figure 5.3 Nodes

5.3.1 Configuring Nodes

The Nodes page contains options to add nodes to the cluster or remove nodes.

- To add nodes:
  1. Click + Add.
  2. Specify the nodes to add.
  3. Click Add Node.

- To remove nodes:
  1. Select one or more nodes from the list.
2. Click **x Remove**.

3. Click **Remove Node(s)** to confirm.

For every node that you select from the list, information about that node is displayed, including the status of the cluster daemons running on the node, resource information, node attributes, and so on. You can manipulate the node further by clicking the options that correspond to the following actions:

- Stop, start, or restart the node.
- Put the node on standby mode.
- Put the node on maintenance mode.

### 5.3.2 Configuring Additional Cluster Properties

To configure more properties of a selected cluster, open the Cluster Properties page. The page displays a list of basic properties for which you specify values on their corresponding fields. To configure properties other than the basic ones, click **Show advanced settings**.

**Figure 5.4 Cluster Properties**

To obtain information about a specific property, hover the mouse pointer over the information icon (i). The icon displays a short description of the property as well as its default value.

For example, the **Batch Limit** property is described as follows:

The number of jobs that the TE is allowed to execute in parallel.

The "correct" value will depend on the speed and load of your network and cluster nodes.

Default value: 0

The properties you customize depend on your circumstances and needs. Suppose that you have a two-node cluster. For this cluster, you want to disable the fencing feature. Because the cluster consists only of two nodes, you do not need any quorum policy. Finally you want to set the migration threshold such that the cluster moves services to a new node after a single failure on a current node. In this case, you would do the following:

1. From the drop-down list of **Stonith Enabled**, select **false** to disable the fencing feature.
2. From the drop-down list of **No Quorum Policy**, select **ignore** to disregard the quorum policy.
3. Click **Show advanced settings** to display migration parameters.

4. On the **migration limit** field, type 1 to set the threshold to a single failure event before services are moved.

5. Click **Apply Changes** to accept your revisions.

6. Click **Refresh** so that the page reflects the changed parameters with their new values.

Typically, you can configure multiple properties in any random order, provided that as a final step, you click **Apply Changes** to effect the new configuration.

### 5.3.3 Setting Local Permissions Through ACLs

Access control lists (ACLs) on the ACLS page are a way of regulating access to a specific cluster so that local users are granted only the permissions they need in the cluster to perform their tasks.

**Figure 5.5 ACLs**

Creating ACLs for the cluster assumes that you have already created users and optionally have added them to defined groups on all the cluster nodes, for example:

```
$ sudo adduser user
$ sudo usermod -a -G group user
```

To define local permissions, do the following:

1. Click **+ Add**.
   
   The Add ACL Role window opens.

2. Provide a role name and description.

3. Click **Add Role**.

4. On the ACLS page, select the permission to be assigned to the role.

   You can grant three types of permissions: **read-only**, **write-only**, or **deny**. **Deny** takes precedence over the other permissions.

5. Specify users who can assume the role and click **Add**.
6. Optionally, specify a group whose members can assume the role and click **Add**.
7. Click the **Cluster Properties** menu item.
8. From the pull-down list of the **Enable ACLs** property, select **true**.

   If you omit this step, the new role remains configured but deactivated.

### 5.3.4 Configuring Fencing

To configure fencing for the cluster, click the appropriate menu item to open the Fence Devices page.

*Figure 5.6 Fence Devices*

For a brief description of fencing and its purpose, see Section 4.1, “About Fencing Configuration (stonith)”.

### 5.3.4.1 Creating Fence Devices

The web UI enables you to configure different kinds of fencing configuration. The configuration options that become available depend on the fencing type you create.

Because of the multiplicity of fencing types, the following steps show you how to configure specifically an IPMI LAN fencing and is based on the example in IPMI LAN Fencing. These steps aim to be a guide for creating other types of fencing:

1. If necessary, check that the **Stonith Enabled** property on the Cluster Properties page is set to *(Default)* or **true**.
2. On the Fence Devices page, click **+ Add**.

   The Add Fence Device window opens.
3. From the **Type** pull-down list, select **fence_ipmilan**.

   You can view a brief description of your selected fencing type through the information icon (*i*).
4. Provide a name for the fence configuration, for example, **ipmilan_n1_fencing**.

   You can configure additional arguments later after creating the fence instance.
5. Click **Create Fence Instance**.
6. Repeat the procedure to create the fence device for the next node, such as **node2**.

   The fence device is listed on the page and information about it is displayed.
5.3.4.2 Configuring Fence Device Arguments

The fence device information includes expandable lists for Optional and Advanced Arguments. For the sample IPMI LAN fence devices you just created, you can define new values for the following arguments:

- **pcmk_host_list**, for example, *node1*
- **delay**, for example, *5* seconds
- **ip** for *node1*’s IP address
- **password** for the administrator password
- **lanplus**, for example, *1* as the level of priority over other fencing types for actions that take effect in case of failure.
- **pcmk_monitor_timeout**, for example, *60* seconds

As with all other properties, information about each argument can be obtained through the information icon.

5.3.5 Adding Resources to the Cluster

You add resources and services to for the cluster’s operations through the Resources page.

**Figure 5.7 Resources**

Add resources to the cluster as follows:

1. Click + **Add**.
   
   The Add Resource window opens.

2. Select a class or provider from the pull-down list.
   
   Your selection determines which types of resources are available to add.

3. Select the resource type from the pull-down list.
   
   The resource type you select automatically creates a description of the selection. For example, the **apache** service is described as "Manage an Apache web server". More information about the type is viewable through the information icon.

4. Required: Specify a valid resource ID.

5. Optionally, configure the other listed parameters.
You can configure Optional and Advanced Arguments later.

6. Click **Create Resource**.

The created resource is added to the list.

If you have multiple resources on the list, you can designate resources to belong to a group as follows:

1. Select resources by clicking their associated boxes.

2. Click **+ Create Group**.

3. Provide a name for the group.

4. Click **+ Create Group**.

On the resource’s information detail, you can manage the resource further through the following options:

- Enforce actions on the resource such as enabling, disabling, refreshing, or removing the resource; performing resource cleanups; and putting the resource in manage or unmanage mode.
- Create a clone or a promotable clone.
- Update the resource’s group information, such as assigning it to another group.
- Configure optional as well as advanced arguments.
Chapter 6 More Information

More information and documentation on Pacemaker and Corosync can be found at http://clusterlabs.org/pacemaker/doc/.