Menu

DOCUMENTATION

FAQ

HELP

QUICKSTART

DEVELOPERS

WIKI

Contact

Stay up to date with the ClusterLabs community by subscribing to our mailing lists or by following the project development on Github.

Blog

Github organization

Mailing lists

Wiki Recent Changes

All examples assume two nodes that are reachable by their short name and IP address:

- `node1 - 192.168.1.1`
- `node2 - 192.168.1.2`

The convention followed is that **[ALL] #** denotes a command that needs to be run on all cluster machines, and **[ONE] #** indicates a command that only needs to be run on one cluster host.

SLES 12

Install

Pacemaker ships as part of the SUSE High Availability Extension. To install, follow the provided documentation. It is also available in openSUSE Leap and openSUSE Tumbleweed.

Create the Cluster

Send site feedback to the ClusterLabs users mailing list.

Theme: HTML5 UP

The supported stack on SLES12 is based on Corosync 2.x.

To get started, install the cluster stack on all nodes.

```
[ALL] # zypper install ha-
cluster-bootstrap
```

First we initialize the cluster on the first machine (node1):

```
[ONE] # ha-cluster-init
```

Now we can join the cluster from the second machine (node2):

```
[TWO] # ha-cluster-join -c
node1
```

These two steps create and start a basic cluster together with the HAWK web interface. If given additional arguments, ha-cluster-init can also configure STONITH, OCFS2 and an administration IP address as part of initial configuration. It is also possible to choose whether to use multicast or unicast for corosync communication.

For more details on ha-cluster-init, see the output of ha-cluster-init --help.

Set Cluster Options

For demonstration purposes, we will force the cluster to move services after a single failure:

```
[ONE] # crm configure
property migration-
threshold=1
```

Add a Resource

Lets add a cluster service, we'll choose one doesn't require any configuration and works everywhere to make things easy. Here's the command:

```
[ONE] # crm configure
primitive my_first_svc
Dummy op monitor
interval=120s
```

"**my_first_svc**" is the name the service will be known as.

"**Dummy**" tells Pacemaker which script to use (Dummy - an agent that's useful as a template and for guides like this one), which namespace it is in (pacemaker) and what standard it conforms to (OCF).

"**op monitor interval=120s**" tells Pacemaker to check the health of this service every 2 minutes by calling the agent's **monitor** action.

You should now be able to see the service running using:

```
[ONE] # crm status
```

Simulate a Service Failure

We can simulate an error by telling the service stop directly (without telling the cluster):

```
[ONE] # crm_resource --
resource my_first_svc --
force-stop
```

If you now run **crm_mon** in interactive mode (the default), you should see (within the monitor interval - 2 minutes) the cluster notice that **my_first_svc** failed and move it to another node.

You can also watch the transition from the HAWK dashboard, by going to https://node1:7630.

Next Steps

- Configure Fencing
- Add more services - see Clusters from Scratch for examples of how to add IP address, Apache and DRBD to a cluster
- Learn how to make services prefer a specific host

- Learn how to make services run on the same host
- Learn how to make services start and stop in a specific order
- Find out what else Pacemaker can do - see Pacemaker Explained for an comprehensive list of concepts and options