



# Rancher and OpenEBS Enterprise Platform

Building Cloud Native On-Premise Solutions with Rancher and OpenEBS Enterprise Platform



This solution document describes the design and best practices when using OpenEBS as persistent storage for any stateful application on the Rancher platform. Topics include: how to get started with Rancher; installation and configuration of OpenEBS on Kubernetes clusters managed by Rancher; setting up OpenEBS storage pools and classes used by Rancher application catalog; how to use chaos engineering in production to harden the deployments and increase operational resilience; and how to monitor production deployments to optimize and plan the capacity and performance needs for short and long term needs.

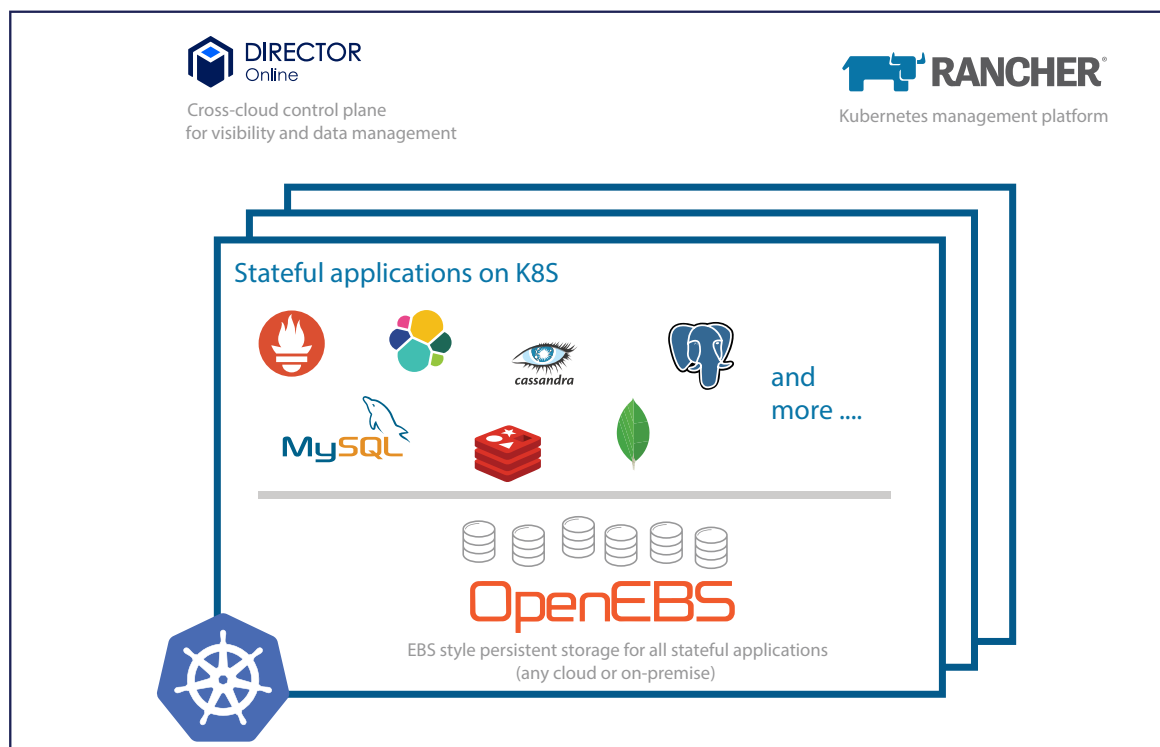
# CONTENTS

<b>INTRODUCTION</b>	<b>1</b>
.....	
<b>WHY USE OPENEBS WITH RANCHER?</b>	<b>3</b>
.....	
<b>STATEFUL APPLICATIONS WORKFLOW</b>	<b>4</b>
.....	
<b>INSTALL AND SETUP RANCHER</b>	<b>5</b>
.....	
<b>SETUP ISCSI PRE-REQUISITES</b>	<b>6</b>
.....	
<b>INSTALL AND SETUP OPENEBS</b>	<b>8</b>
.....	
<b>SETUP STORAGE</b>	<b>9</b>
.....	
<b>SETUP STORAGE CLASSES</b>	<b>11</b>
.....	
<b>DEPLOYING APPS FROM RANCHER CATALOG</b>	<b>12</b>
.....	
<b>HARDEN WITH LITMUS</b>	<b>13</b>
.....	
<b>MONITOR WITH DIRECTOR ONLINE</b>	<b>14</b>
.....	
<b>SUMMARY</b>	<b>15</b>

## INTRODUCTION

Rancher is an efficient and reliable platform, built to manage multiple Kubernetes clusters in production. Rancher not only deploys Kubernetes clusters anywhere, on any provider, but it also unites them under centralized authentication and access control. Because it's agnostic about where the resources run, you can quickly bring up clusters in a different provider and deploy applications to all of them. Instead of having several independent Kubernetes deployments, Rancher unifies them as a single, managed Kubernetes Cloud.

Kubernetes has become the dominant platform for running container workloads and is expanding to run virtual machine based workloads as well. As of early 2019, every primary cloud provider offers at least one version of Kubernetes based managed services and companies including RedHat, Cisco, Nutanix, VMware and others offer on-premise open source based distributions. Designing, provisioning and managing the data needs of stateful applications is an essential functioning in the overall ecosystem of Kubernetes. Choosing the right data management solution is essential as the agility of DevOps is either curtailed or enabled by data management.



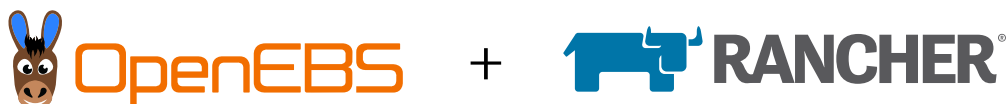
OpenEBS applies common technologies such as Kubernetes and containers as well as preferred architectures such as microservices to deliver run anywhere storage services including block and file storage and local disk management as well as optional data protection and replication features. OpenEBS utilizes Container Attached Storage or CAS architectural principles where the storage software itself runs inside containers that are themselves orchestrated by Kubernetes, making OpenEBS Kubernetes native and cloud-native.

## OpenEBS Features and Benefits



## WHY USE OPENEBS WITH RANCHER?

Rancher orchestrates the overall cloud-native applications in production. For the real power of a multi-cloud solution, OpenEBS augments Rancher with a true cloud-native multi-cloud data management capabilities. OpenEBS is well tested with Rancher in production and nicely integrates into the provisioning capabilities of Rancher console.



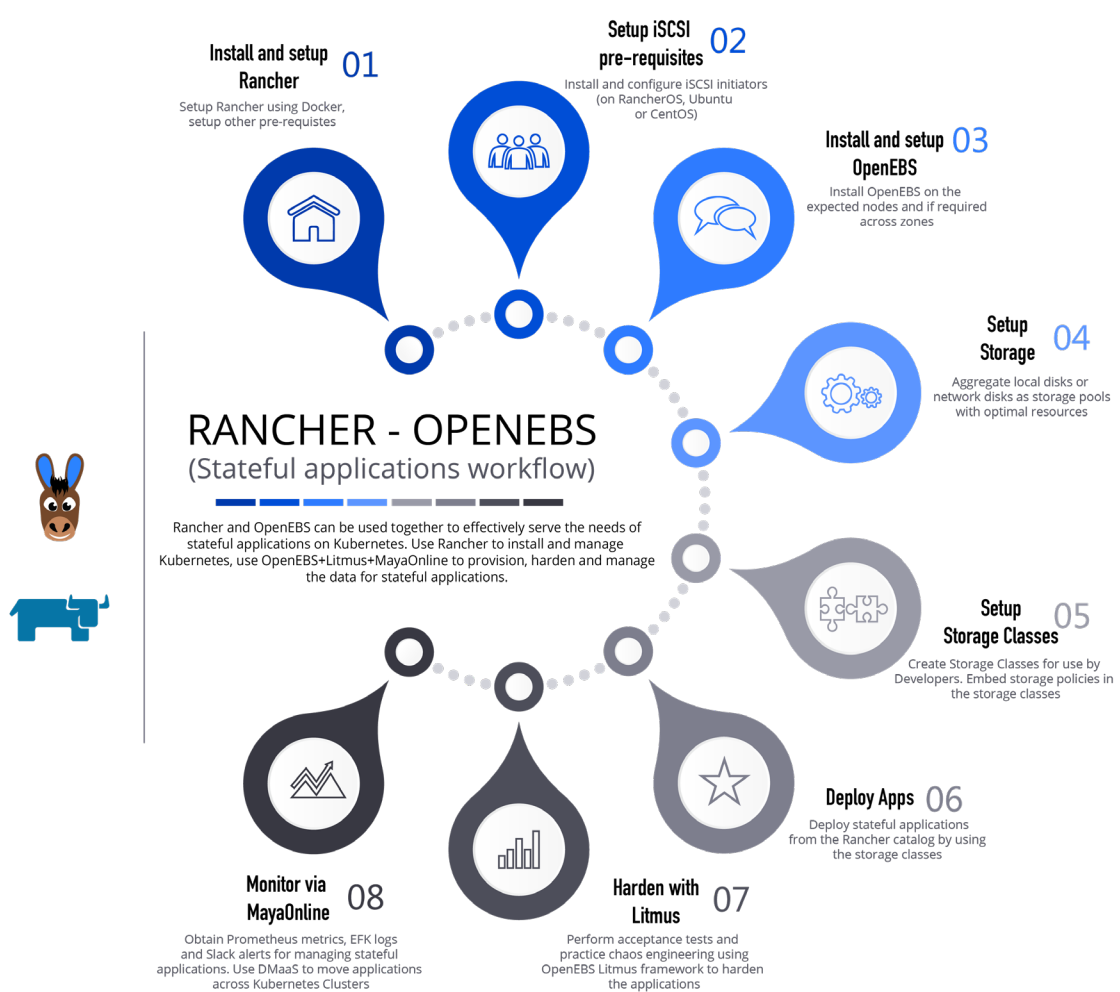
**Following are some of the benefits as observed from the OpenEBS community who are using Rancher as their orchestration platform :**

- **Management of physical disks:** Physical or cloud or network disks are simplified by OpenEBS using its node disk manager or NDM.
- **Easy Provisioning:** Extremely easy to provision and manage data for stateful applications as interfaces to applications are native to Kubernetes, as all resources of OpenEBS are defined as Kubernetes Custom Resources.
- **Thin provisioning:** The volumes for stateful applications can be thin provisioned. Administrators can define storage classes with larger capacities for use by developers or developers can request larger capacities through the PVC interfaces without having to worry about the actual capacity being available. As the storage gets filled in, the physical capacity is extended without any storage service disruptions.
- **Pool aggregation to provide similar storage services as EBS:** With OpenEBS, storage services like AWS EBS are provided for On-Premise Kubernetes deployments. The disks on Kubernetes nodes are pooled together, and volumes are carved out on demand while providing enterprise capabilities to the volumes such as snapshots/cloning/replication.
- **Backup and Restore:** Backup and restore capabilities of OpenEBS volumes can make the stateful application volumes truly multi-cloud. With this feature, the application stateful application are moved across any Kubernetes cluster when needed.

## STATEFUL APPLICATIONS WORKFLOW

Kubernetes clusters are designed to be very scalable while applications are increasingly following the multi-cloud pattern. For these reasons, the infrastructure needs to be designed in such a way that stateful applications can come-in and go-out of the Kubernetes cluster on demand basis.

Following workflow describes various stages for design and implementation considerations. Careful consideration is needed for Storage pools and classes design and hardening methods of applications for achieving desired levels of resiliency against failures.



## INSTALL AND SETUP RANCHER

01

Updated Rancher quick start and configuration guides are available at their docs site <https://rancher.com/docs/rancher/v2.x/en/quick-start-guide/> . However, here we cover short steps to get started with Rancher manual install.

The typical quick start for the manual install of Rancher involve the following steps:

1. Provision a Linux Host
2. Install Rancher
3. Log In

OpenEBS works on any Kubernetes cluster running on Linux hosts. As the primary function of OpenEBS is to provide persistent storage, setting up of storage pools differ slightly from environment to environment depending on the type of disks available to the Kubernetes nodes. To use OpenEBS with Rancher, proceed with the above three steps to install Rancher and login into it. Before creating a Kubernetes cluster using Rancher, consider the following recommendation for the type of Operating System being chosen on the Linux hosts.

**Note:**

On specific cloud provider hosted Kubernetes services such as GKE, the default operating systems on the Linux hosts do not have iSCSI clients. GKE's COS is an example. Choose Ubuntu as the Operating System for the Linux hosts.

Once the correct Operating System that has the support for the iSCSI client is chosen, go ahead and create the Kubernetes cluster on the chosen platform. Verify that Rancher is running as expected before proceeding to the step of setting up the iSCSI client on the nodes.



## SETUP iSCSI PRE-REQUISITES

02

The iSCSI initiator is a pre-requisite for OpenEBS. Setting up or verifying if iSCSI clients are working as expected depends on the Linux Operating System. Use the following instructions depending on the type of Operating System.

### On RancherOS :

If you are using RancherOS as the operating system for your Kubernetes cluster, you just need to enable the iSCSI service and start it on all the hosts or nodes.

```
sudo ros s enable open-iscsi
sudo ros s up open-iscsi
```

Run the below commands on all the nodes to make sure the below directories are persistent, by default these directories are ephemeral.

```
ros config set rancher.services.user-volumes.volumes [ /home:/home, /opt:/opt, /var/lib/
kubelet:/var/lib/kubelet, /etc/kubernetes:/etc/kubernetes, /var/openeps ]
system-docker rm all-volumes
reboot
```

### On Ubuntu/Debian:

If you are running Ubuntu or Debian on all your Linux hosts, first verify if the iSCSI initiator is installed and iSCSI services are running

```
sudo apt install open-iscsi
sudo systemctl enable iscsid && sudo systemctl start iscsid
modprobe iscsi_tcp
```

After installing the initiator tool on your nodes, edit the YAML for your cluster from Rancher web console, and edit the Kubelet configuration to mount the iSCSI binary and configuration, as shown in the sample below.

```
services:
  kubelet:
    extra_binds:
      - "/etc/iscsi:/etc/iscsi"
      - "/sbin/iscsiadm:/sbin/iscsiadm"
      - "/var/lib/iscsi:/var/lib/iscsi"
      - "/lib/modules"
```

## On RHEL/CentOS:

If you are running RHEL/CentOS on all your linux hosts, setup `iscsi_tcp` module to be loaded by default.

The `iscsi_tcp` module is not loaded by default. Use the following command to load it instantly and then create a config file in `/etc/modules-load.d/` directory to make it load at the next reboot of the node.

```
modprobe iscsi_tcp
echo "iscsi_tcp" >/etc/modules-load.d/iscsi-tcp.conf
```

On RHEL/CentOS, it is possible that iSCSI initiator is installed on both the hosts and on the Kubelet. In such cases, the iSCSI initiator on the host needs to be removed or uninstalled. Use the following commands to remove iSCSI initiator configuration on each of the nodes/hosts.

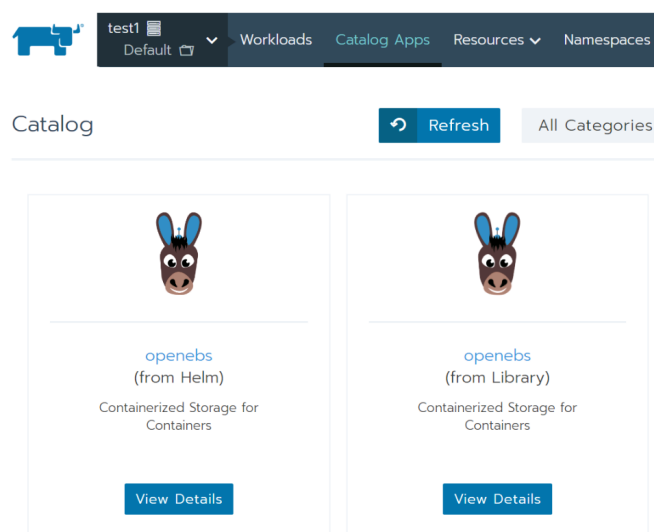
```
systemctl disable iscsid
systemctl stop iscsid
yum erase iscsi-initiator-utils -y
```

## INSTALL AND SETUP OPENEBS

03

As a certified partner, OpenEBS is well tested with Rancher and is inserted into the Rancher Global Catalog. OpenEBS can also be installed from the stable helm chart, which as an OSS software enjoys large community usage, preferred by hundreds of users.

For more detailed instructions on OpenEBS installation and setting up the NodeSelectors before installation, refer to OpenEBS documentation at [https:// docs.openebs.io/docs/next/installation.html](https://docs.openebs.io/docs/next/installation.html)



In the Rancher catalog, search for OpenEBS and launch it. OpenEBS installation is quick and straightforward, typically takes a few minutes. When the installation is complete, you should see the following components in the openebs namespace.

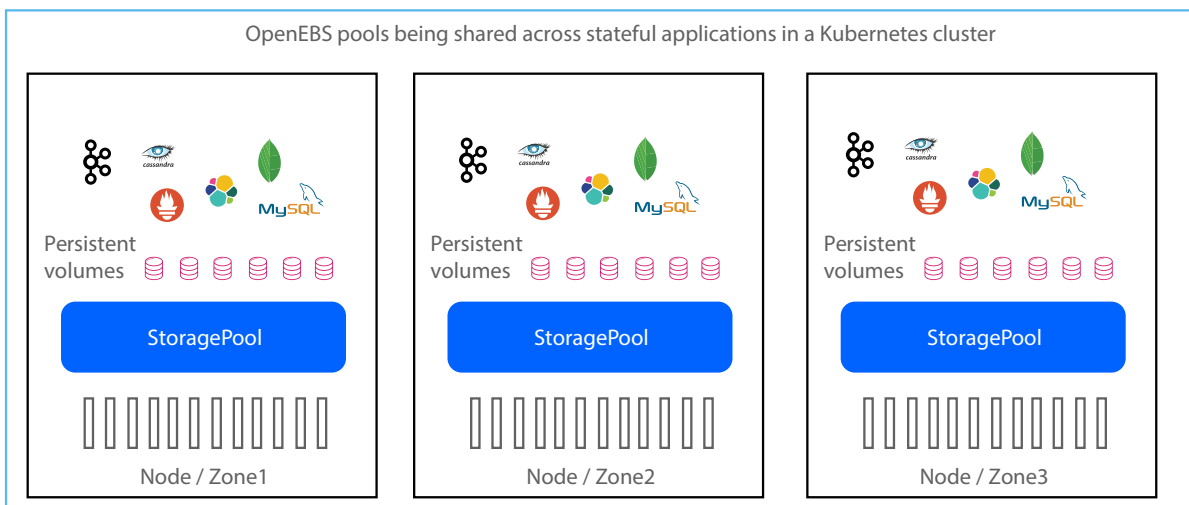
```
root@clone1:~# kubectl get pods -n openebs
```

NAME		READY	STATUS	RESTARTS	AGE
ctor-disk-poo11-2onz-694848ccf8-jljj2	2/2	Running	0	22m	
ctor-disk-poo11-d7ot-b4bc9d9f9-dkzns	2/2	Running	0	22m	
ctor-disk-poo11-jd1v-5b576fccb5-824kk	2/2	Running	0	22m	
ctor-disk-poo11-s4p4-6ff5b8ffd4-s5srd	2/2	Running	0	22m	
ctor-sparse-pool-6btv-65dd8f49d8-268x9	2/2	Running	0	25m	
ctor-sparse-pool-s8qd-67d57776bb-jzfpz	2/2	Running	0	25m	
ctor-sparse-pool-tpg2-5dc48c456c-bf4n5	2/2	Running	0	25m	
openebs-apiserver-5f5cbbb969-52mhq	1/1	Running	0	26m	
openebs-ndm-4xp7c	1/1	Running	0	26m	
openebs-ndm-6fd54	1/1	Running	0	26m	
openebs-ndm-gjkkb	1/1	Running	1	26m	
openebs-ndm-xmpl8	1/1	Running	0	26m	
openebs-provisioner-554bbd965d-tjqx5	1/1	Running	0	26m	
openebs-snapshot-operator-55859594c8-p4lzb	2/2	Running	0	26m	

## SETUP STORAGE

## 04

OpenEBS storage pools are typically a shared resource for many stateful applications running on the Kubernetes cluster. The physical capacity of a pool can be extended dynamically without any disruption to the volumes provisioned on it. Setting up OpenEBS storage involves planning or storage of storage pools and creating storage pools.



A storage pool configuration has multiple individual instances of storage pools, which each pool configured on a set of disks discovered on a node. A storage pool instance hosts one replica of an OpenEBS volume. More details on how replication works in OpenEBS are obtained from OpenEBS documentation (<https://docs.openebs.io/docs/next/cstor.html#cstor-pools>).

**For configuring a storage pool configuration, perform the following steps:**

1. Select the nodes on which pool instances are going to exist.
2. Select the disks on each node that participate in the pool instance.
3. Create a config file with the required pool spec (spc-config.yaml) and do `kubectl apply -f <spc-config.yaml>`.
4. Verify if the pool instances have been created by listing the custom resources SP, SPC, CSP.

```
kubectl get sp,spc,csp
```

A sample YAML for storage pool configuration is shown below. Note that you need to avoid using sparse disks for production usage. Sparse disks examples are provided as a way to setup and understand the pools.

```

---
apiVersion: openebs.io/v1alpha1
kind: StoragePoolClaim
metadata:
  name: cstor-disk-pool1
  annotations:
    cas.openebs.io/config: |
      - name: PoolResourceRequests
        value: |-
          memory: 1Gi
          cpu: 100m
      - name: PoolResourceLimits
        value: |-
          memory: 4Gi
      - name: AuxResourceLimits
        value: |-
          memory: 0.5Gi
          cpu: 50m
spec:
  name: cstor-disk-pool1
  type: disk
  maxPools: 4
  poolSpec:
    poolType: striped
    # NOTE - Appropriate disks need to be fetched using `kubectl get disks`
    #
    # `Disk` is a custom resource supported by OpenEBS with `node-disk-man-
    ager`
    # as the disk operator
    disks:
      diskList:
        - disk-08ff0589098324dcd86b3b3bd5548dc7
        - disk-28ab40402d2651fd49a4f517ec04e354
        - disk-9f34e20c5a5e9afafa4730e7c889d673
        - disk-f000c471bdd562f1d9b66a5ae5931f73
---

```

**Note:**

OpenEBS NDM (Node Disk Manager) discovers all disks attached to the node, both physical disks, and network disks. It is a good practice not to mix these two types of disks. For expanding the pool instance, add the physical or network disks, update the spc-config. yml and rerun the kubectl command.

**Note:**

cStorPools use memory for read cache and hence it is recommended to allocate enough memory (greater than 8G) to cStorPool pods. See <https://docs.openebs.io/docs/next/configurepools.html#poolpolicies> for more details on pool policies.

## SETUP STORAGE CLASSES

## 05

After the storage pools are setup and configured with the required pool resource limits, the next step is to create the storage classes. Each StorageClass refers to the storage pool from where the persistent volumes should be dynamically provisioned.

The storage classes are created for each application with the storage policies differing between each StorageClass. The details of various storage policies that you can apply on storage classes are mentioned in the OpenEBS documentation at <https://docs.openebs.io/docs/next/config-uresc.html#cstor-storage-policies>.

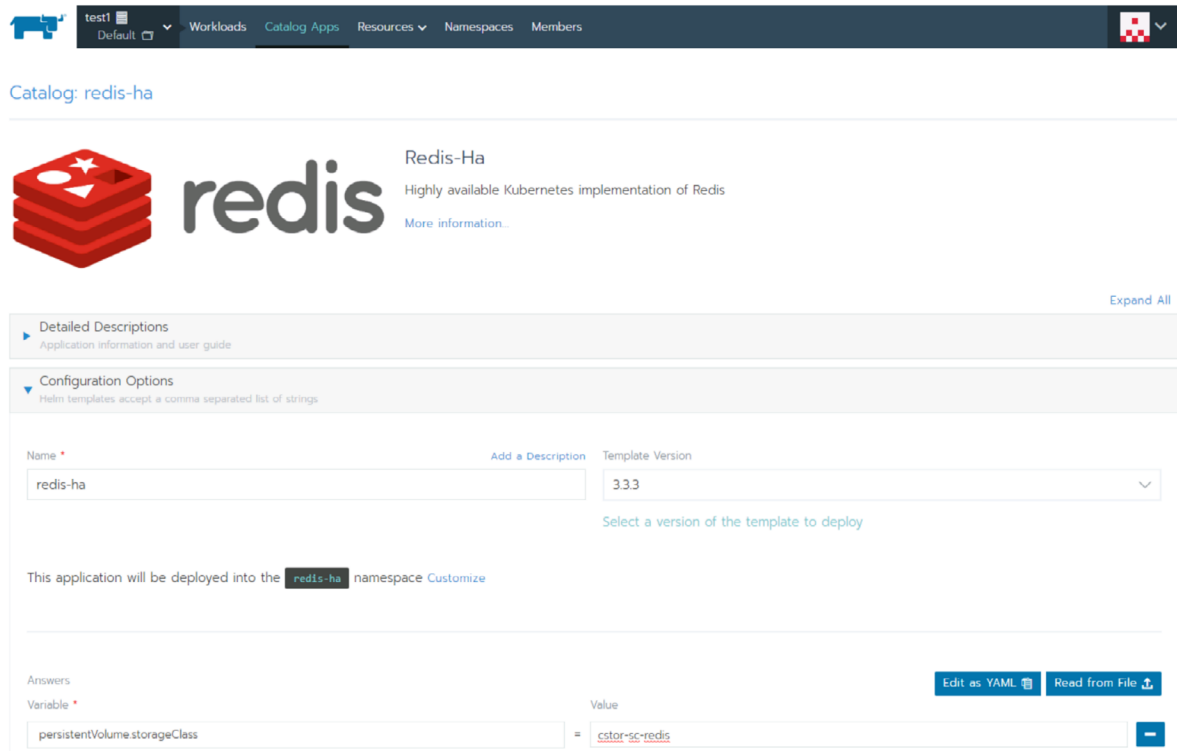
Attention needs to be given to policies such as ReplicaCount, TargetTolerations, TargetAffinity, etc. A sample StorageClass YAML spec is given below.

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cstor-sc-redis
  annotations:
    openebs.io/cas-type: cstor
    cas.openebs.io/config: |
      - name: StoragePoolClaim
        value: "cstor-disk-pool1"
      - name: ReplicaCount
        value: "3"
      - name: TargetResourceLimits
        value: |-
          memory: 1Gi
          cpu: 100m
      - name: AuxResourceLimits
        value: |-
          memory: 0.5Gi
          cpu: 50m
    provisioner: openebs.io/provisioner-iscsi
```


## DEPLOY APPS FOR CATALOG

06

Once the storage classes are created, deploying applications on Kubernetes cluster is easy. You can deploy them from Rancher Catalog by choosing the required StorageClass. An example of Redis being deployed with OpenEBS cStor from Rancher catalog is shown below.



Catalog: redis-ha

 **redis** Redis-Ha  
Highly available Kubernetes implementation of Redis  
[More information...](#)

[Expand All](#)

**Detailed Descriptions**  
Application information and user guide

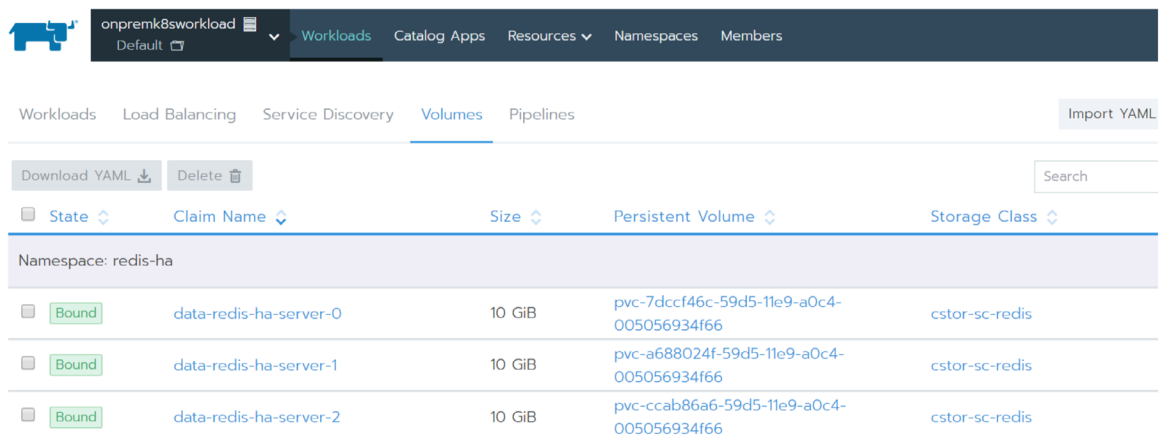
**Configuration Options**  
Helm templates accept a comma separated list of strings

Name:  Add a Description Template Version:   
Select a version of the template to deploy

This application will be deployed into the **redis-ha** namespace [Customize](#)

Answers

Variable:  Value:  [Edit as YAML](#) [Read from File](#)



onpremk8sworkload

Workloads Catalog Apps Resources Namespaces Members

Workloads Load Balancing Service Discovery **Volumes** Pipelines [Import YAML](#)

[Download YAML](#) [Delete](#)

State	Claim Name	Size	Persistent Volume	Storage Class
Namespace: redis-ha				
<input type="checkbox"/> Bound	data-redis-ha-server-0	10 GiB	pvc-7dccb46c-59d5-11e9-a0c4-005056934f66	cstor-sc-redis
<input type="checkbox"/> Bound	data-redis-ha-server-1	10 GiB	pvc-a688024f-59d5-11e9-a0c4-005056934f66	cstor-sc-redis
<input type="checkbox"/> Bound	data-redis-ha-server-2	10 GiB	pvc-ccab86a6-59d5-11e9-a0c4-005056934f66	cstor-sc-redis

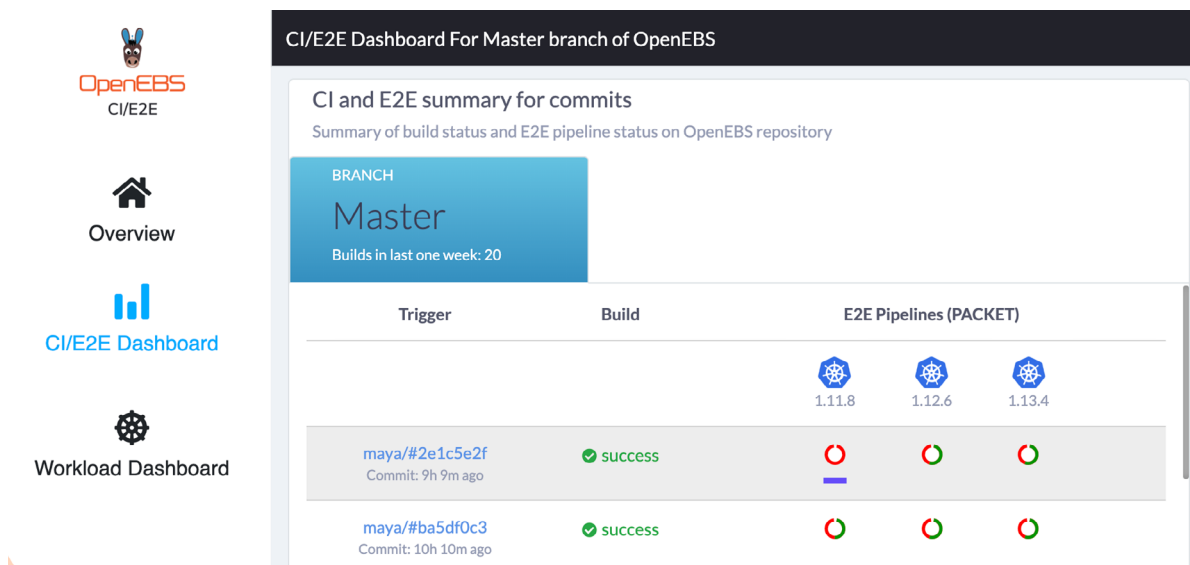
```
# kubectl get sts -n redis-ha
NAME           DESIRED   CURRENT   AGE
redis-ha-server 3         3         29m
```

## HARDEN WITH LITMUS

07

Chaos engineering is an essential discipline towards achieving resilience to failures. Litmus is a chaos engineering framework that works with OpenEBS storage and any other stateful applications on Kubernetes. Using Litmus failures are injected into various layers of the cloud-native ecosystem such as into database components, Kubernetes nodes, Kubelet, OpenEBS components, and physical disks.

Litmus is an open source project that provides an easy to use API for injecting chaos. Litmus can be customized easily to suit specific application or database configurations. The reference implementation for building E2E pipelines for various platforms that use Litmus as the core building block can be found at <https://openebs.ci>



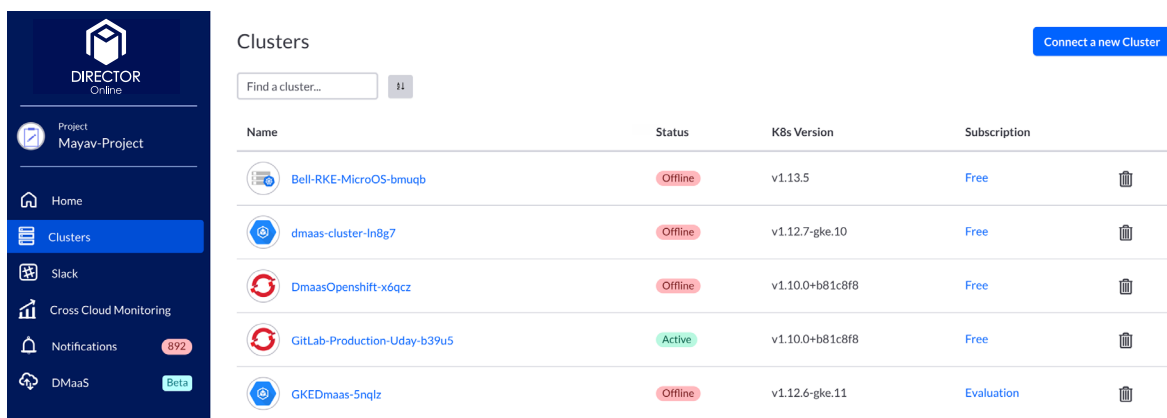
Before moving the Rancher managed clusters into production and while running in production, Litmus can be configured to run Litmus experiments or to inject failures to verify if the system has the expected level of resilience.



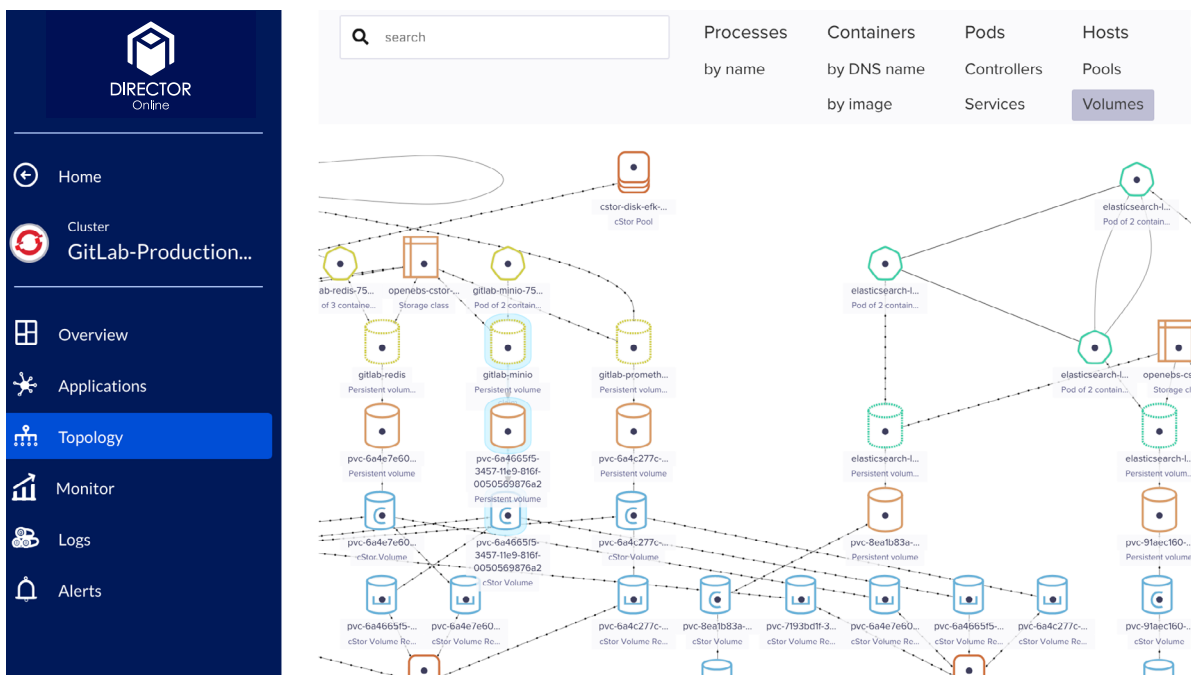
# MONITOR WITH DIRECTOR ONLINE

# 08

Director Online is a SaaS platform that provides cross-cloud visibility and data management capabilities to stateful applications running on Kubernetes. Director Online complimentary capabilities to Rancher console while managing the data needs of stateful applications. It helps in managing the storage resources, achieving granular visibility, ChatOps and data migration across clusters/clouds.



MayaOnline provides granular visibility into Kubernetes resources, especially into OpenEBS storage resources. The logging, monitoring and resources views of stateful applications are helpful to both the developers and administrators for debugging. An example screenshot of the topology view of storage resources is shown below.



## SUMMARY

Rancher introduces a centralized control plane that unifies management of every Kubernetes cluster running across your organization. OpenEBS can be easily installed, configured and managed to achieve the persistent storage needs of stateful applications on Rancher managed Kubernetes clusters. With Director Online, resource monitoring and data migration needs are remarkably simplified.



MayaData



[linkedin.com/company/mayadata/](https://www.linkedin.com/company/mayadata/)



[medium.com/mayadata](https://medium.com/mayadata)



[@mayadata\\_inc](https://twitter.com/mayadata_inc)