

SAP Edge Integration Cell on SUSE

SUSE Linux Enterprise Micro 6.0

Rancher Kubernetes Engine 2

SUSE Storage (Longhorn)

SUSE Rancher Prime

SAP Integration Suite

Kevin Klinger, SAP Solution Architect (SUSE)

Dominik Mathern, SAP Solution Architect (SUSE)

Dr. Ulrich Schairer, SAP Solution Architect (SUSE)

Date: 2025-08-28

SUSE® offers a full stack for your container workloads. This best practice document describes how you can make use of this offerings for your installation of Edge Integration Cell included with SAP Integration Suite. The operations of SAP Edge Integration Cell and/or SAP Integration Suite are not covered in this document.

Disclaimer: Documents published as part of the SUSE Best Practices series have been contributed voluntarily by SUSE employees and third parties. They are meant to serve as examples of how particular actions can be performed. They have been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. SUSE cannot verify that actions described in these documents do what is claimed or whether actions described have unintended consequences. SUSE LLC, its affiliates, the authors, and the translators may not be held liable for possible errors or the consequences thereof.

Contents

1	Introduction	4
2	Supported and used versions	5
3	Prerequisites	6
4	Landscape Overview	7
5	Installing SUSE Linux Enterprise Micro 6.0	9
6	Installing SUSE Rancher Prime cluster	13
7	Installing RKE2 using SUSE Rancher Prime	19
8	Preparing storage	26
9	Installing MetallB and databases	31
10	Installing Edge Integration Cell	39
11	Appendix	40
12	Legal notice	48
13	GNU Free Documentation License	49

1 Introduction

This guide describes how to prepare your infrastructure for the installation of Edge Integration Cell on Rancher Kubernetes Engine 2 using SUSE Rancher Prime. It will guide you through the steps of:

- Installing SUSE Rancher Prime
- Setting up Rancher Kubernetes Engine 2 clusters
- Deploying mandatory components for Edge Integration Cell



Note

This guide does not contain information about sizing your landscapes. Visit <https://help.sap.com/docs/integration-suite?locale=en-US> and search for the "Edge Integration Cell Sizing Guide".

2 Supported and used versions

The support matrix below shows which versions of the given software we will use in this guide.

Product	Version
SUSE Linux Enterprise Micro	6.0
Rancher Kubernetes Engine 2	1.31
SUSE Rancher Prime	2.10.1
SUSE Storage	1.7.2
cert-manager	1.15.2
MetalLB	0.14.7
PostgreSQL	15.7
Redis	7.2.5



Important

If you want to use different versions of SUSE Linux Enterprise Micro or SUSE Linux Micro, SUSE Rancher Prime, Rancher Kubernetes Engine 2, or SUSE Storage, make sure to check the support matrix for the related solutions you want to use: <https://www.suse.com/suse-rancher/support-matrix/all-supported-versions/> ↗

For Redis and PostgreSQL, make sure to pick versions compatible to Edge Integration Cell, which can be found at <https://me.sap.com/notes/3247839> ↗ .

Other versions of MetalLB or cert-manager can be used, but they may not have been tested.

3 Prerequisites

- Get subscriptions for:
 - Rancher for SAP applications *
 - SUSE Linux Enterprise High Availability **

* The Rancher for SAP applications subscription holds support for all required components like SUSE Linux Enterprise Micro, SUSE Rancher Prime and SUSE Storage.

** Only needed if you want to set up SUSE Rancher Prime in a high availability setup
Additionally,

- check the storage requirements.
- create a or get access to a private container registry.
- get an SAP S-user ID to access software and documentation from SAP.
- read the relevant SAP documentation:
 - Release Note for SAP Edge Integration Cell (<https://me.sap.com/notes/3247839>) ↗
 - Release Note for SAP ELM Bridge (<https://me.sap.com/notes/2946788>) ↗
 - Installation Guide at help.sap.com (<https://help.sap.com/docs/integration-suite/sap-integration-suite/setting-up-and-managing-edge-integration-cell>) ↗

4 Landscape Overview

To run Edge Integration Cell in a production-ready and supported way, you need to set up multiple Kubernetes clusters and their nodes. Those comprise a Kubernetes cluster where you will install SUSE Rancher Prime to set up and manage the production and non-production clusters. For this SUSE Rancher Prime cluster, we recommend using three Kubernetes nodes and a load balancer.

The Edge Integration Cell will need to run in a dedicated Kubernetes cluster. For an HA setup of this cluster, we recommend using three Kubernetes control planes and three Kubernetes worker nodes.

For a graphical overview of what is needed, take a look at the landscape overview:

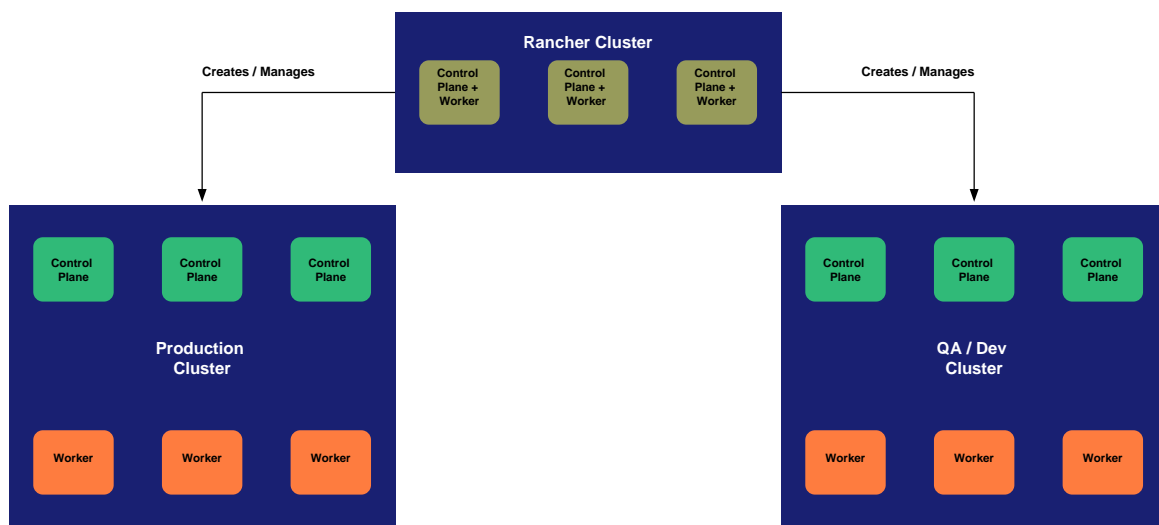


FIGURE 1: ARCHITECTURE OVERVIEW

- The dark blue rectangles represent Kubernetes clusters.
- The olive rectangles represent Kubernetes nodes that hold the roles of Control Plane and Worker combined.
- The green rectangles represent Kubernetes Control Plane nodes.
- The orange rectangles represent Kubernetes Worker nodes.

We will use this graphic overview in the guide to illustrate what the next step is and what it is for.

Starting with installing the operating system of each machine or Kubernetes node, we will walk you through all the steps you need to take to get a fully set-up Kubernetes landscape for deploying Edge Integration Cell.

5 Installing SUSE Linux Enterprise Micro 6.0

There are several ways to install SUSE Linux Enterprise Micro 6.0. For this best practice guide, we use the installation method via graphical installer. But in cloud-native deployments it is highly recommended to use Infrastructure-as-Code technologies to fully automate the deployment and lifecycle processes.

5.1 Installing and configuring SUSE Linux Enterprise Micro

On each server in your environment for Edge Integration Cell and SUSE Rancher Prime, install SUSE Linux Enterprise Micro 6.0 as the operating system. There are several methods to install SUSE Linux Enterprise Micro 6.0 on your hardware or virtual machine. A list of all possible solutions are available in our Documentation [SLE Micro 6.0 \(https://documentation.suse.com/sle-micro/6.0/\)](https://documentation.suse.com/sle-micro/6.0/).

At the end of the installation process, in the summary window, you need to verify that the following security settings are configured:

- The firewall will be disabled.
- The SSH service will be enabled.
- SELinux will be set in permissive mode.

Set SELinux to *permissive* mode, because otherwise, some components of the Edge Integration Cell will violate SELinux rules, and the application will not work.



Tip

If you have already set up all machines and the operating system, skip this chapter.

5.2 Registering your system

To get your system up-to-date, you need to register it with SUSE Manager, an RMT server, or directly with the SCC Portal. Find the registration process with a direct connection to SCC described in the instructions below. For more information, see the SUSE Linux Enterprise Micro documentation.

Registering the system is possible from the command line using the `transactional-update register` command. For information that goes beyond the scope of this section, refer to the inline documentation with **SUSEConnect --help**.

To register SUSE Linux Enterprise Micro with SUSE Customer Center, run `transactional-update register` as follows:

```
sudo transactional-update register -r REGISTRATION_CODE -e EMAIL_ADDRESS
```

To register with a local registration server, additionally specify the URL to the server:

```
sudo transactional-update register -r REGISTRATION_CODE -e EMAIL_ADDRESS \
--url "https://suse_register.example.com/"
```

Do not forget to replace

- **REGISTRATION_CODE** with the registration code you received with your copy of SUSE Linux Enterprise Micro.
- **EMAIL_ADDRESS** with the e-mail address associated with the SUSE account you or your organization uses to manage subscriptions.

Reboot your system to switch to the latest snapshot. SUSE Linux Enterprise Micro is now registered.

Find more information about registering your system in the SUSE Linux Enterprise Micro 6.0 Deployment Guide section [Deploying selfinstall images \(https://documentation.suse.com/sle-micro/5.4/html/SLE-Micro-all/cha-selfinstal-procedure.html\)](https://documentation.suse.com/sle-micro/5.4/html/SLE-Micro-all/cha-selfinstal-procedure.html) ↗.

5.3 Updating your system

Log in to the system. After your system is registered, you can update it with the `transactional-update` command.

```
sudo transactional-update
```

5.4 Disabling automatic reboot

By default SUSE Linux Enterprise Micro runs a timer for `transactional-update` in the background which could automatically reboot your system. Disable it with the following command:

```
sudo systemctl --now disable transactional-update.timer
```

5.5 Preparing for SUSE Storage

For SUSE Storage, some preparation steps are required. First, install some additional packages on all worker nodes. Then, attach a second disk to the worker nodes, create a file system on top of it, and mount it to the default SUSE Storage location. The size of the second disk will depend on your use case.

Install some packages as a requirement for SUSE Storage and Logical Volume Management for adding a file system to SUSE Storage.

```
sudo transactional-update pkg install lvm2 jq nfs-client cryptsetup open-iscsi
```

After the required packages are installed, you need to reboot your machine.

```
sudo reboot
```

Now you can enable the `iscsid` server.

```
sudo systemctl enable iscsid --now
```

5.5.1 Creating file system for SUSE Storage

The next step is to create a new logical volume with the Logical Volume Management.

First, you need to create a new physical volume. In our case, the second disk is called `vdb`. Use this as SUSE Storage volume.

```
sudo pvcreate /dev/vdb
```

After the physical volume is created, create a volume group called `vgdata`:

```
sudo vgcreate vgdata /dev/vdb
```

Now create the logical volume; use 100% of the disk.

```
sudo lvcreate -n lvlonghorn -l100%FREE vgdata
```

On the logical volume, create the XFS file system. You do not need to create a partition on top of it.

```
sudo mkfs.xfs /dev/vgdata/lvlonghorn
```

Before you can mount the device, you need to create the directory structure.

```
sudo mkdir -p /var/lib/longhorn
```

Add an entry to *fstab* to ensure that the mount of the file system is persistent:

```
sudo echo -e "/dev/vgdata/lvlonghorn /var/lib/longhorn xfs defaults 0 0" >> /etc/fstab
```

Finally, you can mount the file system as follows:

```
sudo mount -a
```

6 Installing SUSE Rancher Prime cluster

By now you should have installed the operating system on every Kubernetes node. You are now ready to install a SUSE Rancher Prime cluster. Taking a look again on the landscape overview, this means, we will now cover how to set up the upper part of the given graphic:

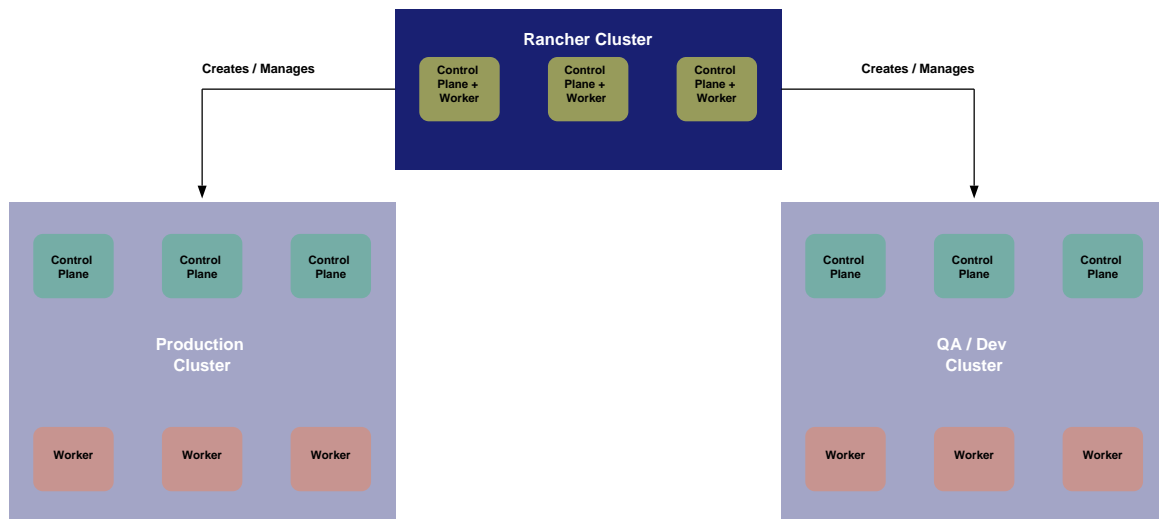


FIGURE 2: ARCHITECTURE SUSE RANCHER PRIME

6.1 Preparation

To provide a highly available SUSE Rancher Prime setup, you need a load balancer for your SUSE Rancher Prime nodes. If you already have a load balancer, you can use that to make SUSE Rancher Prime highly available.

If you do not plan to set up a highly available SUSE Rancher Prime cluster, you can skip this section.

6.1.1 Installing a haproxy-based load balancer

This section describes how to set up a custom load balancer using haproxy.

Set up a virtual machine or a bare metal server with SUSE Linux Enterprise Server and SUSE Linux Enterprise High Availability or use SUSE Linux Enterprise Server for SAP applications. Install the haproxy package.

```
sudo zypper in haproxy
```

Create the configuration for haproxy. Find an example configuration file for haproxy below and adapt for the actual environment.

```
sudo cat <<EOF > /etc/haproxy/haproxy.cfg
global
    log /dev/log      local0
    log /dev/log      local1 notice
    chroot /var/lib/haproxy
    # stats socket /run/haproxy/admin.sock mode 660 level admin
    stats timeout 30s
    user haproxy
    group haproxy
    daemon

    # general hardlimit for the process of connections to handle, this is separate to
backend/listen
    # Added in 'global' AND 'defaults'!!! - global affects only system limits
    (ulimit/maxsock) and defaults affects only listen/backend-limits - hez
    maxconn 400000

    # Default SSL material locations
    ca-base /etc/ssl/certs
    crt-base /etc/ssl/private

    tune.ssl.default-dh-param 2048

    # Default ciphers to use on SSL-enabled listening sockets.
    # For more information, see ciphers(1SSL). This list is from:
    # https://hynek.me/articles/hardening-your-web-servers-ssl-ciphers/
    ssl-default-bind-ciphers ECDH+AESGCM:DH+AESGCM:ECDH+AES256:DH+AES256:ECDH
+AES128:DH+AES:ECDH+3DES:DH+3DES:RSA+AESGCM:RSA+AES:RSA+3DES:!aNULL:!MD5:!DSS
    ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    mode tcp
    log      global
    option   tcplog
    option   redispatch
    option   tcpka
    option   dontlognull
```

```

    retries 2
    timeout connect 5s
    timeout client 5s
    timeout server 5s
    timeout tunnel 86400s
    maxconn 400000

listen stats
    bind *:9000
    mode http
    stats hide-version
    stats uri /stats

listen rancher_apiserver
    bind my_lb_address:6443
    option httpchk GET /healthz
    http-check expect status 401
    server mynode1 mynode1.domain.local:6443 check check-ssl verify none
    server mynode2 mynode2.domain.local:6443 check check-ssl verify none
    server mynode3 mynode3.domain.local:6443 check check-ssl verify none

listen rancher_register
    bind my_lb_address:9345
    option httpchk GET /ping
    http-check expect status 200
    server mynode1 mynode1.domain.local:9345 check check-ssl verify none
    server mynode2 mynode2.domain.local:9345 check check-ssl verify none
    server mynode3 mynode3.domain.local:9345 check check-ssl verify none

listen rancher_ingress80
    bind my_lb_address:80
    option httpchk GET /
    http-check expect status 404
    server mynode1 mynode1.domain.local:80 check
    server mynode2 mynode2.domain.local:80 check
    server mynode3 mynode3.domain.local:80 check

listen rancher_ingress443
    bind my_lb_address:443
    option httpchk GET /
    http-check expect status 404
    server mynode1 mynode1.domain.local:443 check check-ssl verify none
    server mynode2 mynode2.domain.local:443 check check-ssl verify none
    server mynode3 mynode3.domain.local:443 check check-ssl verify none

EOF

```

Check the configuration file:

```
haproxy -f /path/to/your/haproxy.conf -c
```

Enable and start the haproxy load balancer:

```
sudo systemctl enable haproxy
sudo systemctl start haproxy
```

Do not forget to restart or reload haproxy if any changes are made to the haproxy configuration file.

6.2 Installing RKE2

To install RKE2, the script provided at <https://get.rke2.io>  can be used as follows:

```
sudo curl -sL https://get.rke2.io | INSTALL_RKE2_VERSION=v1.31.7+rke2r1 sh
```

For HA setups, you must create RKE2 cluster configuration files in advance. On the first master node, do the following:

```
sudo mkdir -p /etc/rancher/rke2
cat <<EOF > /etc/rancher/rke2/config.yaml
token: 'your cluster token'
system-default-registry: registry.rancher.com
tls-san:
  - FQDN of fixed registration address on load balancer
  - other hostname
  - IP v4 address
EOF
```

Create configuration files for additional cluster nodes:

```
cat <<EOF > /etc/rancher/rke2/config.yaml
server: https://"FQDN of registration address":9345
token: 'your cluster token'
system-default-registry: registry.rancher.com
tls-san:
  - FQDN of fixed registration address on load balancer
  - other hostname
  - IP v4 address
EOF
```


Important

You also need to consider taking etcd snapshots and perform backups of your Rancher instance. These topics are not covered in this document, but you can find more information in our official documentation. Helpful links are https://documentation.suse.com/cloudnative/rke2/latest/en/backup_restore.html and <https://documentation.suse.com/cloudnative/rancher-manager/latest/en/rancher-admin/back-up-restore-and-disaster-recovery/back-up-restore-and-disaster-recovery.html>.
IMPORTANT: For security reasons, we generally recommend activating the CIS profile when installing RKE2. This is currently still being validated and will be included in the documentation at a later date.

Now enable and start the RKE2 components and run the following command on each cluster node:

```
sudo systemctl enable rke2-server --now
```

To verify the installation, run the following command:

```
/var/lib/rancher/rke2/bin/kubectl --kubeconfig /etc/rancher/rke2/rke2.yaml get nodes
```

For convenience, you can add the `kubectl` binary to the **\$PATH** and set the specified `kubeconfig` via an environment variable:

```
export PATH=$PATH:/var/lib/rancher/rke2/bin/  
export KUBECONFIG=/etc/rancher/rke2/rke2.yaml
```

6.3 Installing Helm

To install SUSE Rancher Prime and some of its required components, you need to use Helm. One way to install Helm is to run:

```
curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
```

6.4 Installing cert-manager

To install the `cert-manager` package, do the following:

```
kubectl create namespace cert-manager
```

If you want to install `cert-manager` from the *application-collection* , you must create an `imagePullSecret`.

How to create the `imagePullSecret` is described in the [Section 11.1, “Creating an imagePullSecret for the Rancher Application Collection”](#).

6.4.1 Installing the application

Before you can install the application, you need to login into the registry. You can find the instruction in [Section 11.2, “Logging in to the Application Collection Registry”](#).

```
helm install cert-manager oci://dp.apps.rancher.io/charts/cert-manager \
  --set crds.enabled=true \
  --set-json 'global.imagePullSecrets=[{"name":"application-collection"}]' \
  --namespace=cert-manager \
  --version 1.15.2
```

6.5 Installing SUSE Rancher Prime

To install SUSE Rancher Prime, you need to add the related Helm repository. To achieve that, use the following command:

```
helm repo add rancher-prime https://charts.rancher.com/server-charts/prime
```

Next, create the `cattle-system` namespace in Kubernetes as follows:

```
kubectl create namespace cattle-system
```

The Kubernetes cluster is now ready for the installation of SUSE Rancher Prime:

```
helm install rancher rancher-prime/rancher \
  --namespace cattle-system \
  --set hostname=<your.domain.com> \
  --set replicas=3
```

During the rollout of SUSE Rancher Prime, you can monitor the progress using the following command:

```
kubectl -n cattle-system rollout status deploy/rancher
```

When the deployment is done, you can access the SUSE Rancher Prime cluster at `https://<your.domain.com>`. Here you will also find a description about how to log in for the first time.

7 Installing RKE2 using SUSE Rancher Prime

After having installed the SUSE Rancher Prime cluster, you can now use it to create the Rancher Kubernetes Engine 2 clusters for Edge Integration Cell. SAP recommends to set up not only a production landscape, but to have QA / Dev systems for Edge Integration Cell. Both can be set up the same way using SUSE Rancher Prime. How to do this is covered in this chapter. Returning to the landscape overview, we will now focus on setting up the lower part of the graphic below:

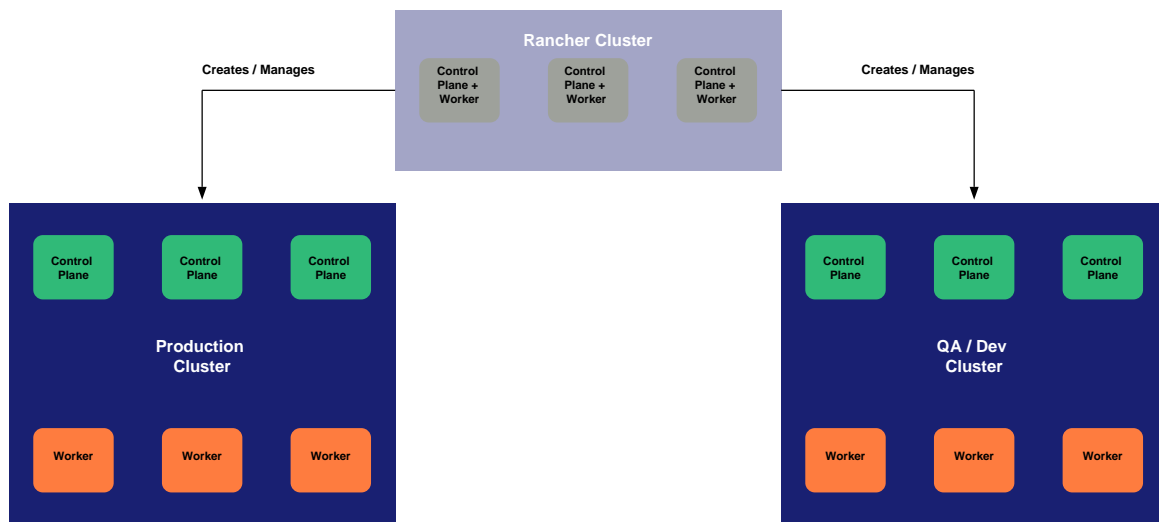


FIGURE 3: ARCHITECTURE OVERVIEW RKE2

Creating new RKE2 clusters is straightforward when using SUSE Rancher Prime.

Navigate to the home menu of your SUSE Rancher Prime instance and click the **Create** button on the right side of the screen, as shown below:

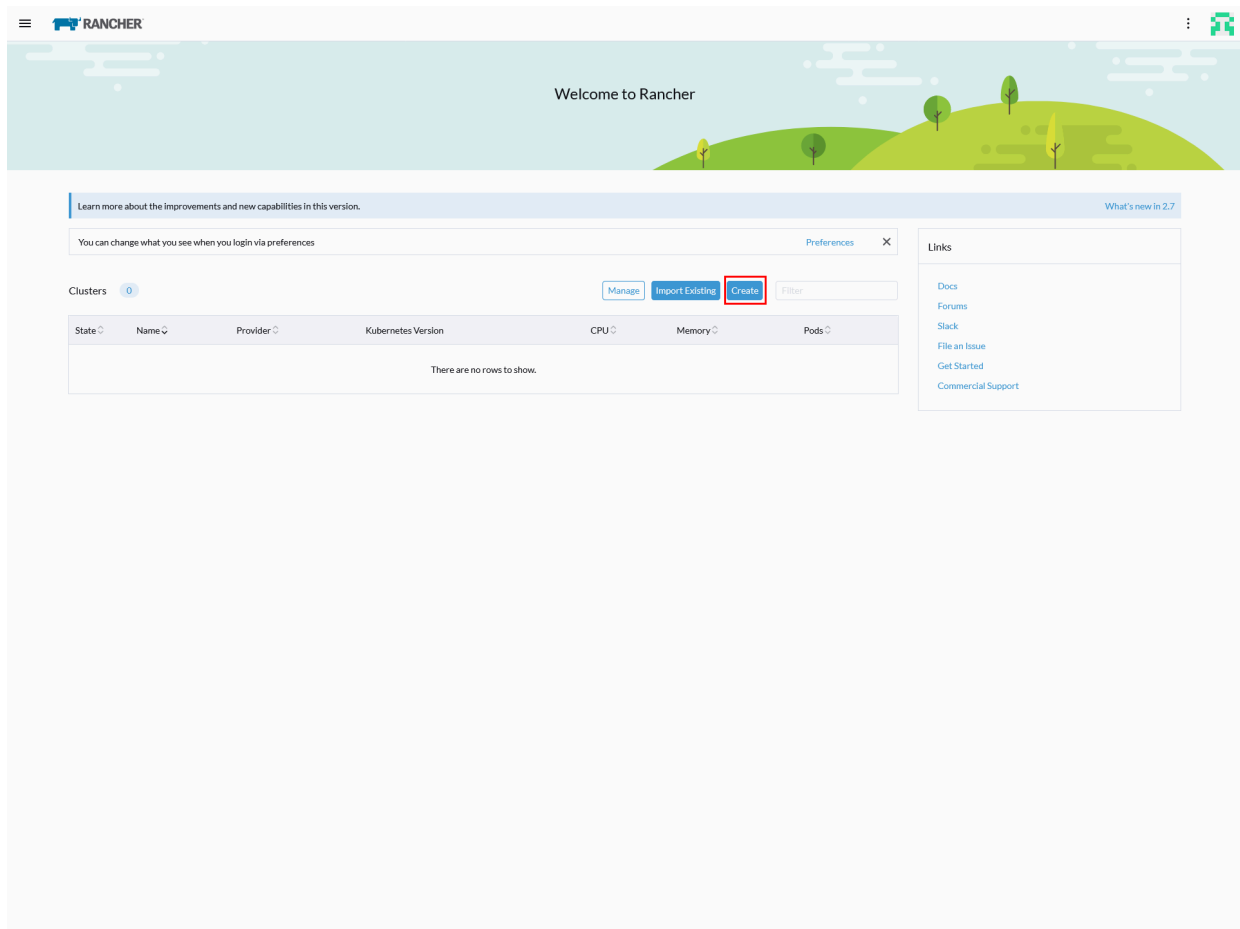


FIGURE 4: RANCHER HOME MENU

The window displays the available options for creating new Kubernetes clusters. Make sure the toggle button on the right side of the screen is set to **RKE2/K3s**, as shown below:

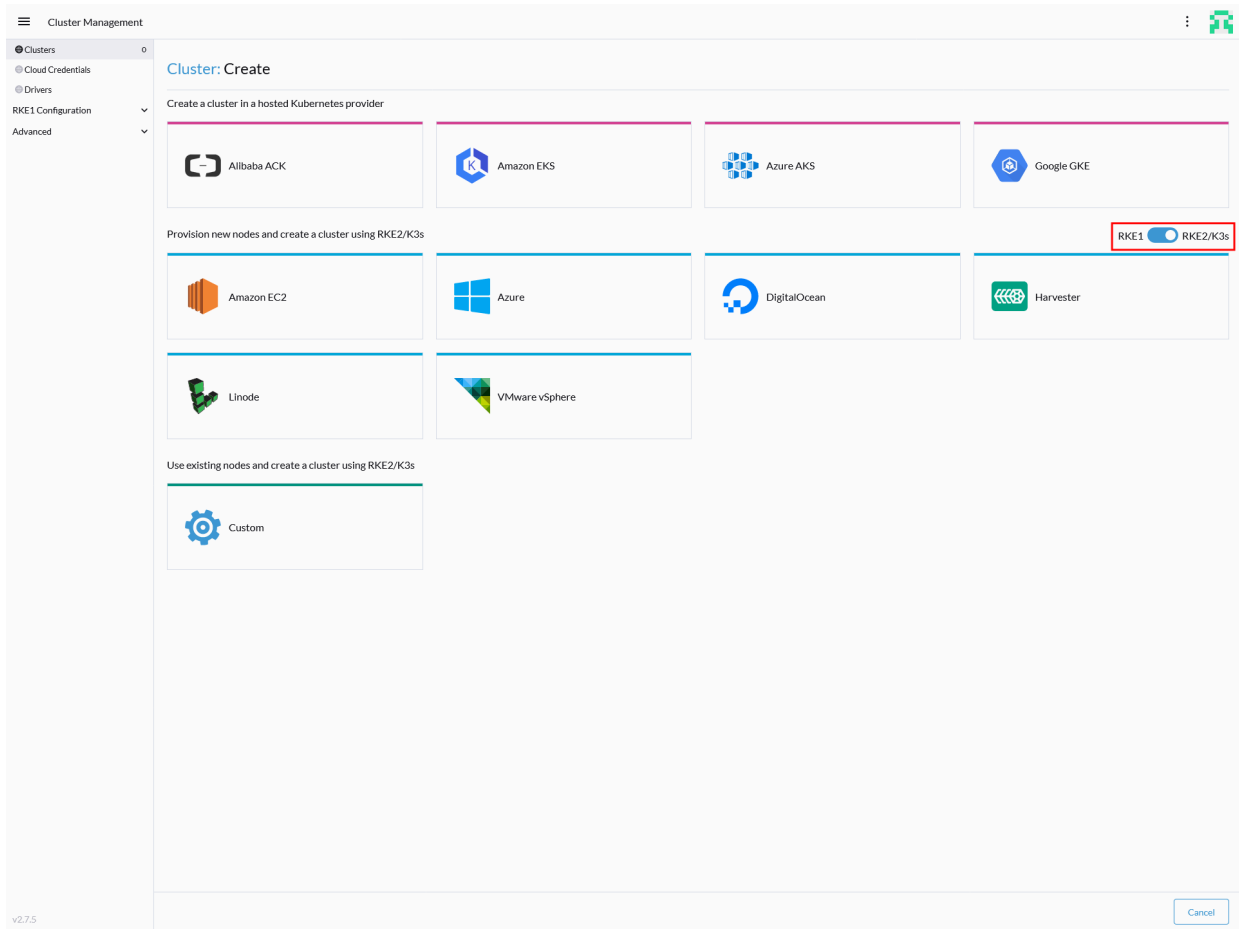


FIGURE 5: RANCHER RKE VERSION SELECTION

If you want to create Kubernetes clusters on existing (virtual) machines, select the **Custom** option at the very bottom, as shown in the image below:

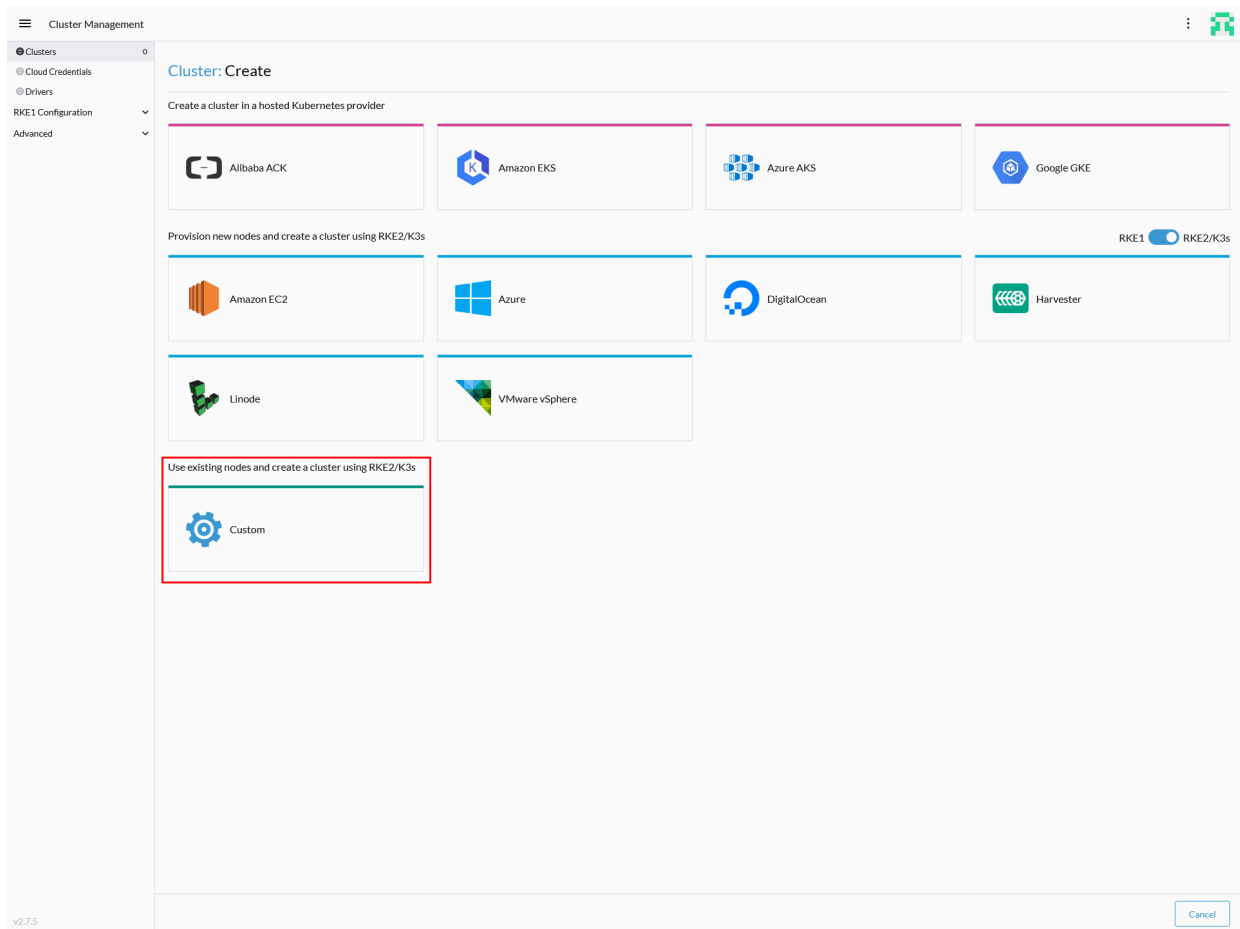


FIGURE 6: RANCHER CREATE CUSTOM CLUSTER

Next, a window will appear where you can configure your Kubernetes cluster. It will look similar to the image below:

The screenshot shows the 'Cluster: Create Custom' configuration window in Rancher. The interface includes a left-hand navigation menu with options like 'Clusters', 'Cloud Credentials', 'Drivers', 'RKE1 Configuration', and 'Advanced'. The main area is titled 'Cluster Configuration' and contains several sections: 'Basics' with fields for 'Cluster Name' (set to 'example') and 'Cluster Description', a 'Kubernetes Version' dropdown (set to 'v1.26.13+rke2r1'), a 'Cloud Provider' dropdown (set to 'Default - RKE2 Embedded'), and a 'Container Network' dropdown (set to 'calico'); 'Security' with a warning about Pod Security Policies, a 'CIS Profile' dropdown (set to 'None'), a 'Pod Security Admission Configuration Template' dropdown (set to 'Default - RKE2 Embedded'), and a 'Project Network Isolation' checkbox; and 'System Services' with checkboxes for 'CoreDNS', 'NGINX Ingress', and 'Metrics Server' (all checked). At the bottom right, there are 'Cancel', 'Edit as YAML', and 'Create' buttons. The version 'v2.7.5' is visible in the bottom left corner.

FIGURE 7: RANCHER CREATE CUSTOM CLUSTER CONFIG

Here, you need to name the cluster. The name will only be used within SUSE Rancher Prime. It will not affect your workloads. In the next step, make sure you select a Kubernetes version that is supported by the workload you want to deploy.

If you do not have any further requirements to Kubernetes, you can click the **Create** button at the very bottom. In any other cases, talk to your administrators before making adjustments. After you clicked **Create**, you should see a screen similar to the below:

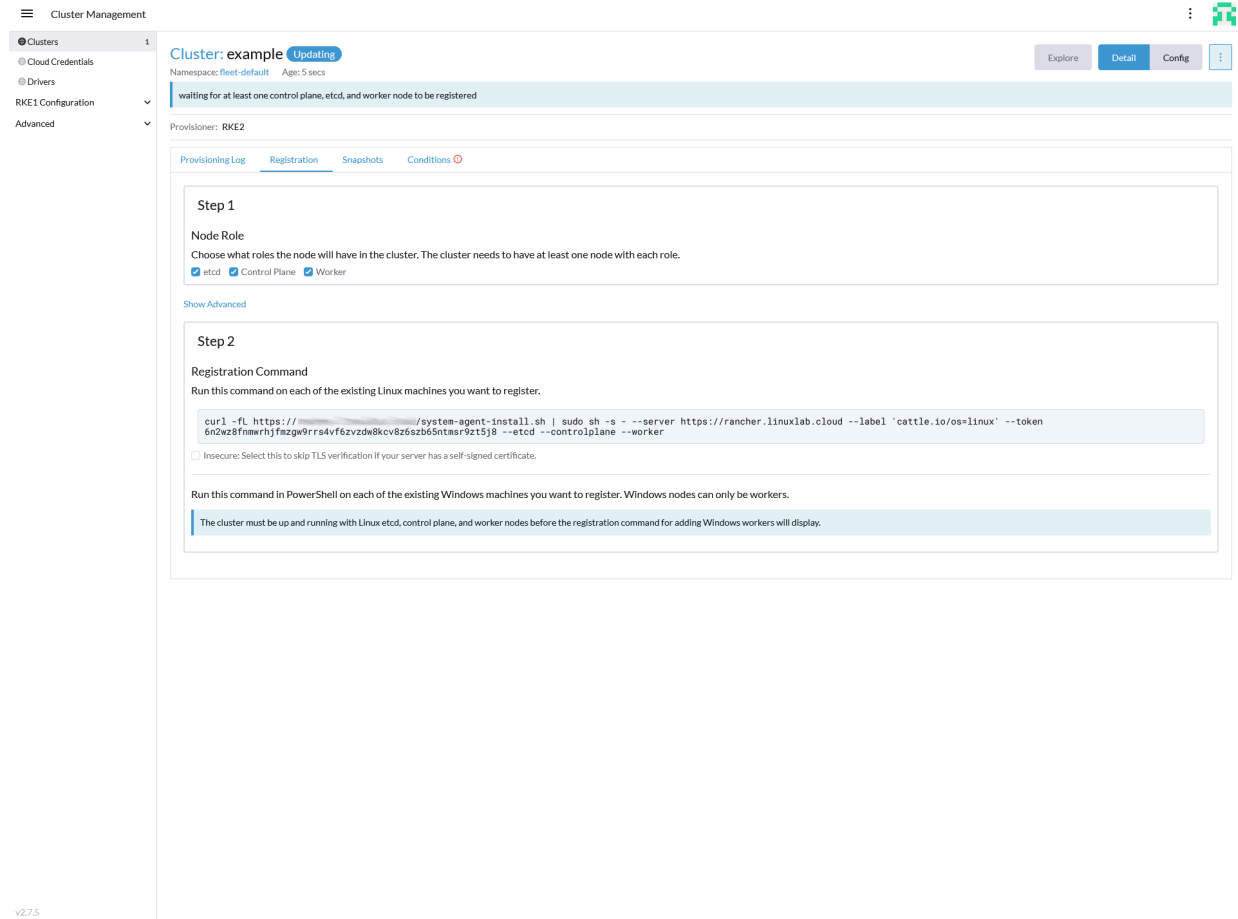


FIGURE 8: RANCHER CREATE REGISTRATION

Now, in a first step, select the roles your node(s) should receive. A common high availability setup holds:

- 3 x etcd / control plane nodes
- 3 x worker nodes

The next step is to copy the registration command to the target machines' shell and execute it. If your SUSE Rancher Prime instance holds a self-signed certificate, make sure to activate the text bar holding the registration command in the check box below.

You can run the command on all nodes in parallel. You do not need to wait until a single node is down. When all machines are registered, you can see the cluster status at the top, changing from "updating" to "active". At this point in time, your Kubernetes cluster is ready to be used.


8 Preparing storage

To make storage available for Kubernetes workloads, prepare the storage solution you want to use. In this chapter, we will describe how to set this up and how to prepare it for consumption by Edge Integration Cell.

The storage solutions tested by SAP and SUSE are presented below, along with links to chapters detailing their setup and configuration.

- SUSE Storage [Section 8.1, "Installing SUSE Storage"](#)
- NetApp Trident [Section 8.2, "Installing Trident"](#)

8.1 Installing SUSE Storage

This chapter details the minimum requirements to install SUSE Storage and describes three different ways for the installation. For more details, visit <https://longhorn.io/docs/1.6.2/deploy/install/> .

8.1.1 Requirements

To ensure a node is prepared for SUSE Storage, you can use the following script to check:

```
curl -sSfL https://raw.githubusercontent.com/longhorn/longhorn/v1.6.2/scripts/
environment_check.sh | bash
```

8.1.2 Installing SUSE Storage using SUSE Rancher Prime

Up-to-date and detailed instructions how to install SUSE Storage using SUSE Rancher Prime can be found at <https://longhorn.io/docs/1.6.2/deploy/install/install-with-rancher/> .

8.1.3 Installing SUSE Storage using Helm

To install Longhorn using Helm, run the following commands:

```
helm repo add rancher-v2.8-charts https://raw.githubusercontent.com/rancher/charts/
release-v2.8
helm repo update
helm upgrade --install longhorn-crd rancher-v2.8-charts/longhorn-crd \
--namespace longhorn-system \
--create-namespace
helm upgrade --install longhorn rancher-v2.8-charts/longhorn \
--namespace longhorn-system
```

For more details, visit <https://longhorn.io/docs/1.6.2/deploy/accessing-the-ui/longhorn-ingress/> .

8.2 Installing Trident

This chapter describes how to install Trident in a RKE2 cluster using Helm. For more details how to install Trident with Helm, visit <https://docs.netapp.com/us-en/trident/trident-get-started/kubernetes-deploy-helm.html#critical-information-about-trident-25-02> ↗

8.2.1 Preparing the OS

The Kubernetes worker nodes must be prepared so PVCs can later be provisioned properly. Thus, you need to install the following packages:

```
sudo zypper in -y lsscsi multipath-tools open-iscsi
```

As multipathd is known to have problems on OS using the kernel-default-base packages, replace them with kernel-default:

```
sudo zypper remove -y kernel-default-base
sudo zypper in -y kernel-default
```

Afterwards you can enable iscsi and multipath:

```
sudo systemctl enable --now iscsi
sudo systemctl enable --now multipathd
```

8.2.2 Deploying the Trident operator

The Trident operator is responsible to establish the connection between your NetApp storage system and the Kubernetes cluster. An example of its deployment is shown below:

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
helm install my-trident-operator netapp-trident/trident-operator --version 100.2502.1 --
create-namespace --namespace trident
```

8.2.3 Establishing the connection between Kubernetes and the NetApp storage

Before creating the link to the backend, you should store the user and password information in a Secret. To create such a Secret, follow the example below:

```
apiVersion: v1
kind: Secret
metadata:
```

```
name: backend-tbc-ontap-secret
namespace: trident
type: Opaque
stringData:
  username: <cluster-admin>
  password: <password>
```

To establish the connection between the target Kubernetes cluster and the NetApp storage system, a so-called **TridentBackendConfig** is required. For more information how to set up the backend configuration, refer to <https://docs.netapp.com/us-en/trident/trident-use/backend-kubectrl.html#tridentbackendconfig> ↗

Below are two examples demonstrating the configuration of SAN/iSCSI and NAS backends:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
  namespace: trident
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: <Cluster IP>
  dataLIF: <Storage-VM-IP>
  svm: <Storage-VM-FQDN>
  credentials:
    name: backend-tbc-ontap-secret
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  backendName: ontap-nas-backend
  storageDriverName: ontap-nas
  managementLIF: <Cluster-IP>
  dataLIF: <Storage-VM-IP>
  svm: <Storage-VM-FQDN>
  credentials:
    name: backend-tbc-ontap-secret
```

To verify the backend was configured successfully, check the output of:

```
kubectl -n trident get tbc backend-tbc-ontap-san
```

If the connection was established, the **State** should be active and you should see a **Backend UUID**.

When the backend is configured, you can create a **StorageClass** to provision Volumes to be consumed by a Persistent Volume Claim. Here's an example for creating a StorageClass that uses a SAN/iSCSI backend:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-san
```

9 Installing MetalLB and databases

In the following chapter we present an example for setting up MetalLB, Redis and PostgreSQL.



Note

Keep in mind that the descriptions and instructions below might differ from the deployment you need for your specific infrastructure and use cases.

9.1 Logging in to Rancher Application Collection

To access the Rancher Application Collection, you need to log in. You can do this using the console and Helm client. The easiest way to do so is to use the built-in shell in SUSE Rancher Prime. To access it, navigate to your cluster and click **Kubectl Shell** as shown below:

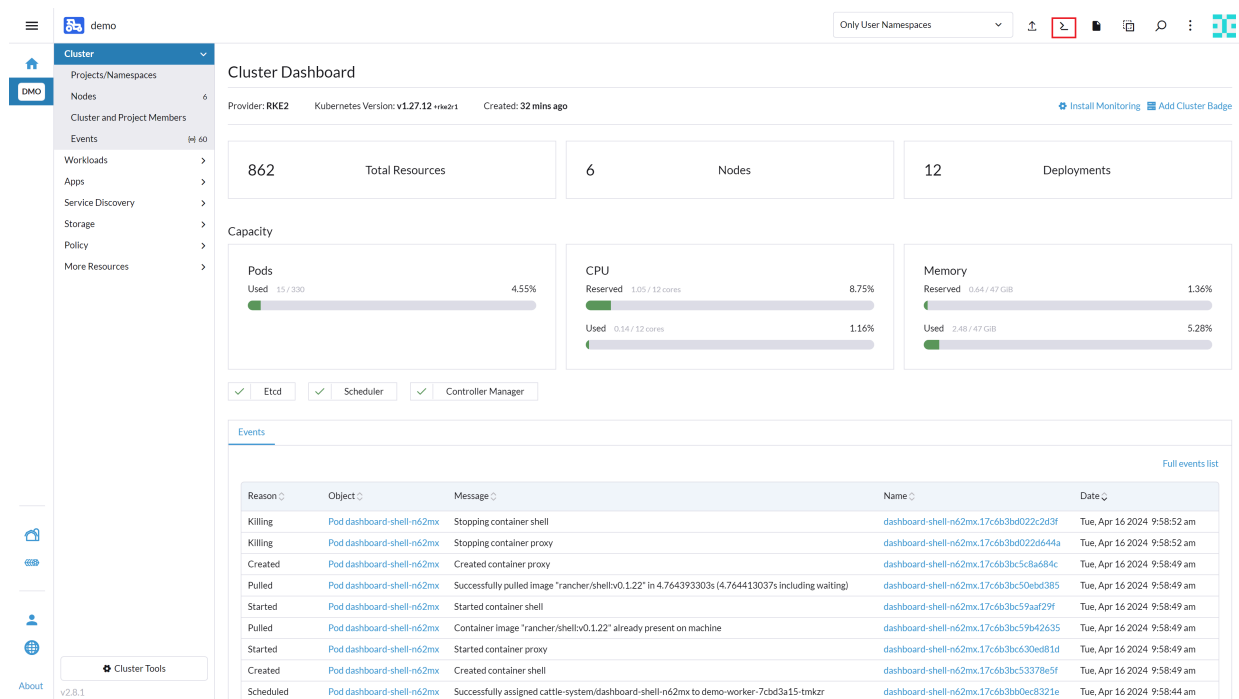


FIGURE 9: RANCHER SHELL ACCESS

A shell will open as shown in the image below:

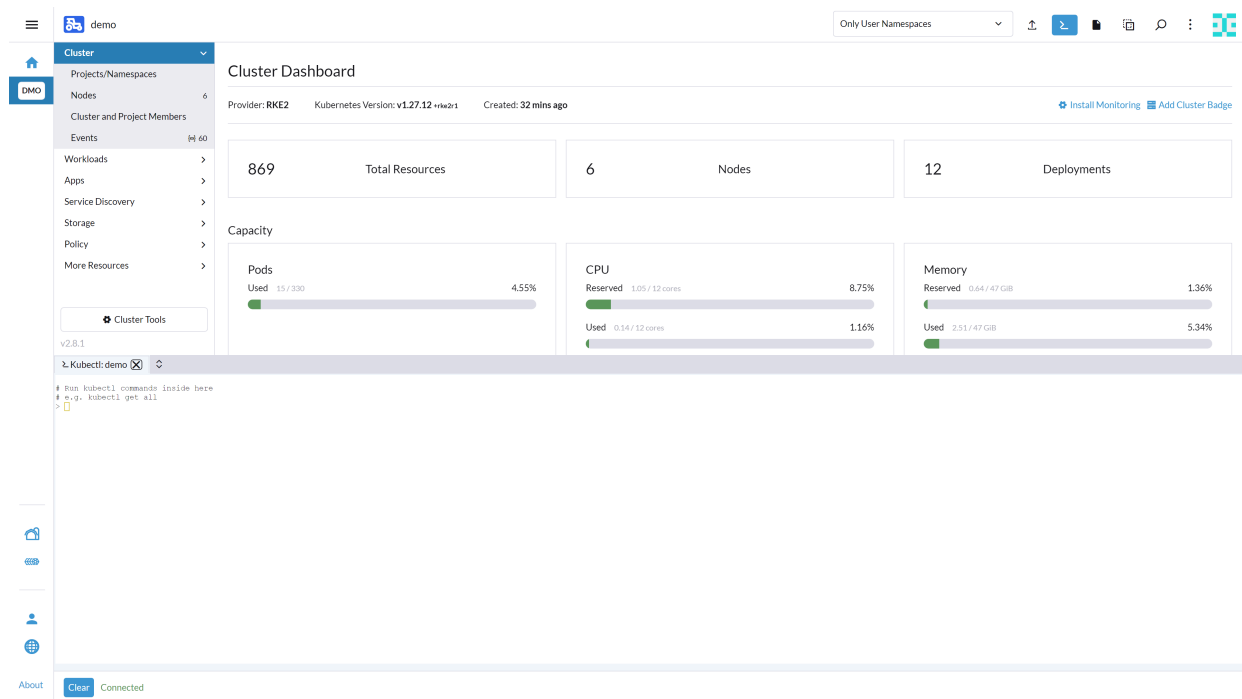


FIGURE 10: RANCHER SHELL OVERVIEW

You must log in to Rancher Application Collection. This can be done as follows:

```
helm registry login dp.apps.rancher.io/charts -u <yourUser> -p <your-token>
```

9.2 Installing MetalLB on Kubernetes cluster

The following chapter should guide you through the installation and configuration of MetalLB on your Kubernetes cluster used for Edge Integration Cell.

9.2.1 Installing and configuring MetalLB

There are multiple ways to install the MetalLB software. In this guide, we will cover how to install MetalLB using `kubectl` or Helm. A complete overview and more details about MetalLB can be found on the [official website for MetalLB \(https://metallb.universe.tf/\)](https://metallb.universe.tf/).

9.2.1.1 Prerequisites

Before starting the installation, ensure that all requirements are met. In particular, you should pay attention to network add-on compatibility. If you are trying to run MetalLB on a cloud platform, you should also look at the cloud compatibility page and make sure your cloud platform works with MetalLB (note that most cloud platforms do **not**).

There are several ways to deploy MetalLB. In this guide, we will describe how to use the Rancher Application Collection to deploy MetalLB.

Make sure to have one IP address available for configuring MetalLB.

Before you can deploy MetalLB from Rancher Application Collection, you need to create the namespace and an `imagePullSecret`. To create the related namespace, run:

```
kubectl create namespace metallb
```

Instructions how to create the `imagePullSecret` can be found in [Section 11.1, "Creating an imagePullSecret for the Rancher Application Collection"](#).

9.2.1.2 Installing MetalLB

Before you can install the application, you need to log in to the registry. You can find the instructions in [Section 11.2, "Logging in to the Application Collection Registry"](#).

Create a `values.yaml` file with the following configuration:

```
imagePullSecrets:
  - name: application-collection
```

Then install the `metallb` application.

```
helm install metallb oci://dp.apps.rancher.io/charts/metallb \
-f values.yaml \
--namespace=metallb \
--version 0.14.7
```

9.2.1.3 Configuring MetalLB

MetalLB needs two configurations to function properly:

- IP address pool
- L2 advertisement configuration

Create the configuration files for the MetalLB IP address pool:

```
cat <<EOF >iprange.yaml
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: first-example-pool
  namespace: metallb
spec:
  addresses:
    - 192.168.1.240/32
EOF
```

Create the layer 2 network advertisement:

```
cat <<EOF > l2advertisement.yaml
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: example
  namespace: metallb
EOF
```

Apply the configuration:

```
kubectl apply -f iprange.yaml
kubectl apply -f l2advertisement.yaml
```

9.3 Installing Redis

Before deploying Redis, ensure that the requirements described at <https://me.sap.com/notes/3247839> are met.

Also, make sure you understand what grade of persistence you want to achieve for your Redis cluster. For more information about persistence in Redis, see <https://redis.io/docs/management/persistence/>.

IMPORTANT

SUSE does not offer database support for Redis. For support requests, contact [Redis Ltd.](https://redis.com/) (<https://redis.com/>).

IMPORTANT

The following instructions describe only one variant of installing Redis which is called Redis Cluster. There are other possible ways to set up Redis that are not covered in this guide. Check if you require [Redis Sentinel](https://redis.io/docs/management/sentinel/) instead of [Redis Cluster](https://redis.io/docs/management/scaling/).

9.3.1 Deploying Redis

Although Redis is available for deployment using the SUSE Rancher Prime Apps, we recommend using the Rancher Application Collection. The Redis chart can be found at <https://apps.rancher.io/applications/redis>.

9.3.1.1 Deploying the chart

To deploy the chart, create the related namespace and **imagePullSecret** first. To create the namespace, run:

```
kubectl create namespace redis
```

Instructions how to create the **imagePullSecret** can be found in [Section 11.1, "Creating an imagePullSecret for the Rancher Application Collection"](#).

If you want to use self-signed certificates, you can find instructions how to create such in [Section 11.3.1, "Creating self-signed certificates"](#).

Before you can install the application, you need to log in to the registry. You can find the instructions in [Section 11.2, "Logging in to the Application Collection Registry"](#).

Create a file *values.yaml* to store some configurations for the Redis Helm chart. The configuration might look like the following:

```
images:
  redis:
    # -- Image registry to use for the Redis container
    registry: dp.apps.rancher.io
    # -- Image repository to use for the Redis container
    repository: containers/redis
    # -- Image tag to use for the Redis container
    tag: 7.2.5
storageClassName: "longhorn"
global:
```

```

imagePullSecrets: ["application-collection"]
architecture: cluster
nodeCount: 3
auth:
  password: <redisPW>
tls:
  # -- Enable TLS
  enabled: true
  # -- Whether to require Redis clients to authenticate with a valid certificate
  (authenticated against the trusted root CA certificate)
  authClients: false
  # -- Name of the secret containing the Redis certificates
  existingSecret: "redisCert"
  # -- Certificate filename in the secret
  certFilename: "server.pem"
  # -- Certificate key filename in the secret
  keyFilename: "server.key"
  # CA certificate filename in the secret - needs to hold the CA.crt and the server.pem
  caCertFilename: "root.pem"

```

To install the application, run:

```

helm install redis oci://dp.apps.rancher.io/charts/redis \
-f values.yaml \
--namespace=redis

```

9.4 Installing PostgreSQL

Before deploying PostgreSQL, ensure that the requirements described at <https://me.sap.com/notes/3247839> are met.

IMPORTANT

SUSE does **not** offer database support for PostgreSQL on Kubernetes. Find information about support options at [The PostgreSQL Global Development Group \(https://www.postgresql.org/support/\)](https://www.postgresql.org/support/).

IMPORTANT

The instructions below describe only one variant of installing PostgreSQL. There are other possible ways to set up PostgreSQL which are not covered in this guide. It is also possible to install PostgreSQL as a single instance on the operating system. We will focus on installing PostgreSQL in a Kubernetes cluster, as we also need a Redis database, and we will clustering that together.

9.4.1 Deploying PostgreSQL

Even though PostgreSQL is available for deployment using the SUSE Rancher Prime Apps, we recommend to use the Rancher Application Collection. The PostgreSQL chart can be found at <https://apps.rancher.io/applications/postgresql>.

9.4.2 Creating secret for Rancher Application Collection

First, create a namespace and the **imagePullSecret** for installing the PostgreSQL database onto the cluster.

```
kubectl create namespace postgresql
```

How to create the **imagePullSecret** is described in *Section 11.1, "Creating an imagePullSecret for the Rancher Application Collection"*.

9.4.2.1 Creating secret with certificates

Second, create the Kubernetes secret with the certificates. You will find an example how to do this in *Section 11.3.1, "Creating self-signed certificates"*.

9.4.2.2 Installing the application

Before you can install the application, you need to log in to the registry. You can find the instruction in *Section 11.2, "Logging in to the Application Collection Registry"*.

Create a file **values.yaml** which holds some configurations for the PostgreSQL Helm chart. The configuration may look like:

```
global:
  # -- Global override for container image registry pull secrets
  imagePullSecrets: ["application-collection"]
images:
  postgresql:
    # -- Image registry to use for the PostgreSQL container
    registry: dp.apps.rancher.io
    # -- Image repository to use for the PostgreSQL container
    repository: containers/postgresql
    # -- Image tag to use for the PostgreSQL container
    tag: "15.7"
auth:
```

```

# -- PostgreSQL username for the superuser
postgresUsername: postgres
# -- PostgreSQL password for the superuser
postgresPassword: "<your_password>"
# -- Replication username
replicationUsername: replication
# -- Replication password
replicationPassword: "<your_password>"
tls:
# -- Enable SSL/TLS
enabled: false
# -- Name of the secret containing the PostgreSQL certificates
existingSecret: "postgresqlcert"
# -- Whether or with what priority a secure SSL TCP/IP connection will be negotiated
with the server. Valid values: prefer (default), disable, allow, require, verify-ca,
verify-full
sslMode: "verify-full"
# -- Certificate filename in the secret (will be ignored if empty)
certFilename: "server.pem"
# -- Certificate key filename in the secret (will be ignored if empty)
keyFilename: "server.key"
# -- CA certificate filename in the secret (will be ignored if empty)
caCertFilename: "root.pem"
statefulset:
# -- Enable the StatefulSet template for PostgreSQL standalone mode
enabled: true
# -- Lifecycle of the persistent volume claims created from PostgreSQL
volumeClaimTemplates
persistentVolumeClaimRetentionPolicy:
## -- Volume behavior when the StatefulSet is deleted
whenDeleted: Delete
podSecurityContext:
# -- Enable pod security context
enabled: true
# -- Group ID for the pod
fsGroup: 1000

```

To install the application, run:

```

helm install postgresql oci://dp.apps.rancher.io/charts/postgresql -f values.yaml --
namespace=postgresql

```

10 Installing Edge Integration Cell

At this point, you should be able to deploy Edge Integration Cell. Follow the instructions at <https://help.sap.com/docs/integration-suite/sap-integration-suite/setting-up-and-managing-edge-integration-cell> to install Edge Integration Cell in your prepared environments.

11 Appendix

11.1 Creating an imagePullSecret for the Rancher Application Collection

To make the resources available for deployment, you need to create an imagePullSecret. In this guide, we use the name *application-collection* for it.

11.1.1 Creating an imagePullSecret using kubectl

Using `kubectl` to create the imagePullSecret is quite easy. Get your user name and your access token for the Rancher Application Collection. Then run:

```
kubectl -n <namespace> create secret docker-registry application-collection --docker-server=dp.apps.rancher.io --docker-username=<yourUser> --docker-password=<yourPassword>
```

As secrets are namespace-sensitive, you need to create this for every required namespace.

The related secret can then be used for the components:

- Cert-Manager ([Section 6.4, “Installing cert-manager”](#) (page 18))
- MetalLB ([Section 9.2.1.1, “Prerequisites”](#) (page 33))
- Redis ([Section 9.3.1.1, “Deploying the chart”](#) (page 35))
- PostgreSQL ([Section 9.4.2, “Creating secret for Rancher Application Collection”](#) (page 37))

11.1.2 Creating an imagePullSecret using SUSE Rancher Prime

You can also create an imagePullSecret using SUSE Rancher Prime. To do so, open SUSE Rancher Prime and enter your cluster.

Navigate to **Storage** → **Secrets** as shown below:

The screenshot shows the SUSE Rancher Prime interface. On the left, a sidebar contains a menu with the following items: Cluster, Workloads, Apps, Service Discovery, Storage (highlighted with a red box), PersistentVolumes, StorageClasses, ConfigMaps, PersistentVolumeClaims, Secrets (highlighted with a red box), Policy, and More Resources. The main area displays the 'Cluster Dashboard' for a cluster named 'demo'. It shows the following metrics: 871 Total Resources, 6 Nodes, and 12 Deployments. Below these are three capacity charts: Pods (Used: 15 / 330, 4.55%), CPU (Reserved: 1.05 / 12 cores, 8.75%; Used: 0.17 / 12 cores, 1.42%), and Memory (Reserved: 0.64 / 47 GiB, 1.36%; Used: 2.49 / 47 GiB, 5.30%). At the bottom, there is an 'Events' section with a table of recent events.

Reason	Object	Message	Name	Date
Killing	Pod dashboard-shell-hdr5j	Stopping container shell	dashboard-shell-hdr5j.17c6b4c012425e89	Tue, Apr 16 2024 10:17:25 am
Killing	Pod dashboard-shell-hdr5j	Stopping container proxy	dashboard-shell-hdr5j.17c6b4c0124393af	Tue, Apr 16 2024 10:17:25 am
Created	Pod dashboard-shell-hdr5j	Created container proxy	dashboard-shell-hdr5j.17c6b4284c5b4e89	Tue, Apr 16 2024 10:06:33 am
Started	Pod dashboard-shell-hdr5j	Started container proxy	dashboard-shell-hdr5j.17c6b42852de50f3	Tue, Apr 16 2024 10:06:33 am
Created	Pod dashboard-shell-hdr5j	Created container shell	dashboard-shell-hdr5j.17c6b428432c7fef	Tue, Apr 16 2024 10:06:33 am
Started	Pod dashboard-shell-hdr5j	Started container shell	dashboard-shell-hdr5j.17c6b428497f6c26	Tue, Apr 16 2024 10:06:33 am
Pulled	Pod dashboard-shell-hdr5j	Container image "rancher/shell:v0.1.22" already present on machine	dashboard-shell-hdr5j.17c6b4284989be1e	Tue, Apr 16 2024 10:06:33 am
Pulled	Pod dashboard-shell-hdr5j	Container image "rancher/shell:v0.1.22" already present on machine	dashboard-shell-hdr5j.17c6b42840150778	Tue, Apr 16 2024 10:06:33 am
Scheduled	Pod dashboard-shell-hdr5j	Successfully assigned cattle-system/dashboard-shell-hdr5j to demo-worker-7cbd3a15-tmkrz	dashboard-shell-hdr5j.17c6b42819ed3c15	Tue, Apr 16 2024 10:06:32 am

FIGURE 11: SECRETS MENU

Click the **Create** button in the top right corner.

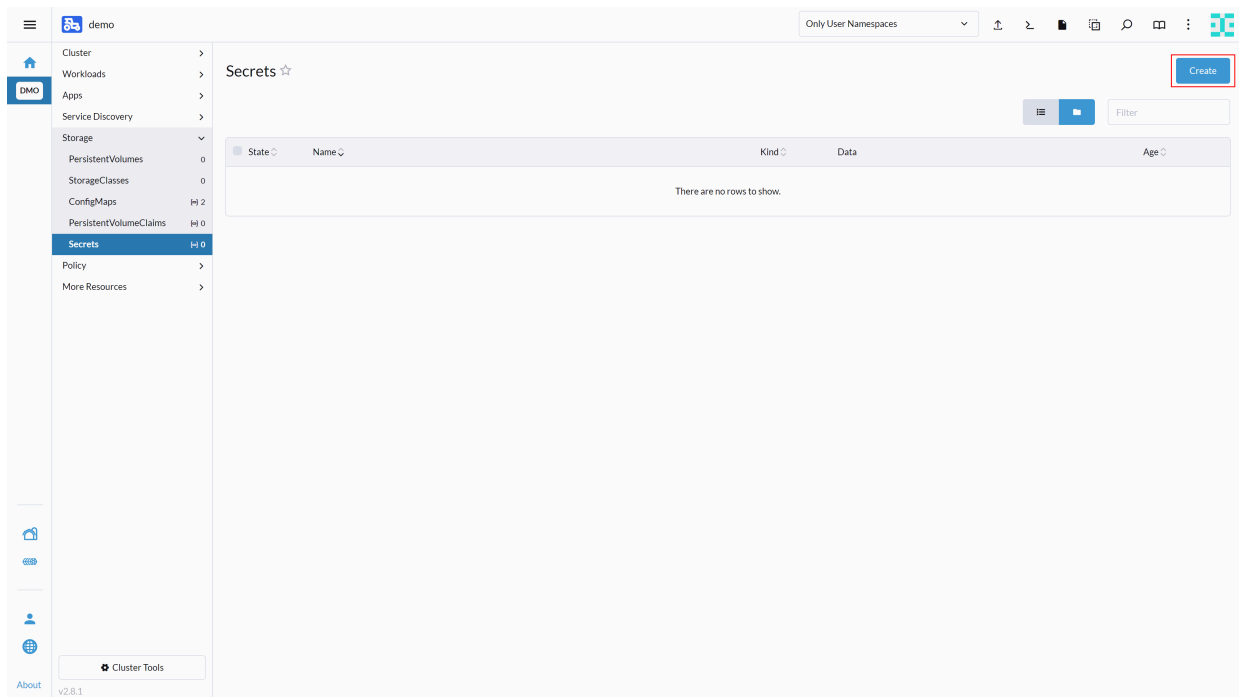


FIGURE 12: SECRETS OVERVIEW

A window will appear asking you to select the secret type. Select **Registry** as shown here:

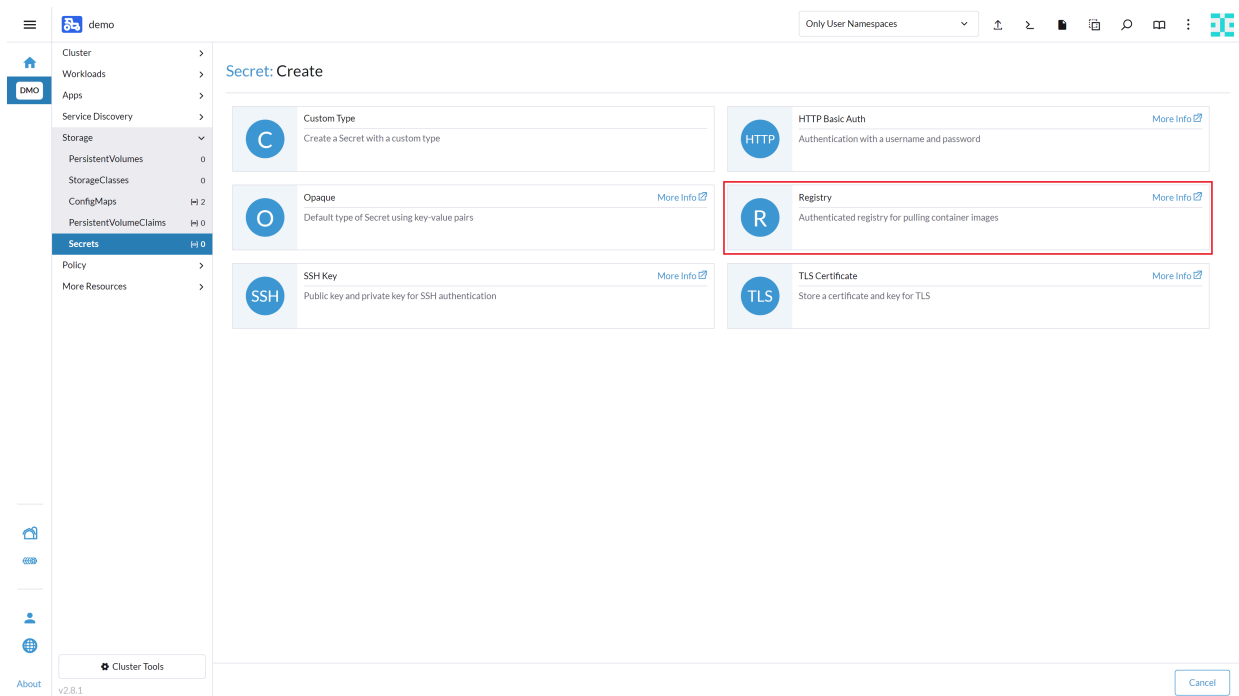


FIGURE 13: SECRETS TYPE SELECTION

Enter a name such as *application-collection* for the secret. In the text box **Registry Domain Name**, enter *dp.apps.rancher.io*. Enter your user name and password and click the **Create** button at the bottom right.

The screenshot shows the Rancher UI interface for creating a registry secret. On the left is a sidebar with a navigation menu including Cluster, Workloads, Apps, Service Discovery, Storage, PersistentVolumes, StorageClasses, ConfigMaps, PersistentVolumeClaims, Secrets (highlighted), Policy, and More Resources. The main panel is titled 'Secret: Create Registry'. It contains a form with the following fields: 'Namespace' (set to 'default'), 'Name' (set to 'application-collection'), and 'Description' (set to 'Image Pull Secret for the Rancher Application Collection'). Below these is a 'Data' section with a 'Labels & Annotations' tab and a 'Data' tab. The 'Data' tab has radio buttons for 'Custom' (selected), 'DockerHub', 'Quay.io', and 'Artifactory'. Below these are input fields for 'Registry Domain Name' (set to 'dp.apps.rancher.io'), 'Username' (set to 'example@demo.com'), and 'Password' (masked with dots). At the bottom right are three buttons: 'Cancel', 'Edit as YAML', and 'Create'. The bottom left of the interface shows 'About v2.8.1' and 'Cluster Tools'.

FIGURE 14: SECRETS CREATION STEP

11.2 Logging in to the Application Collection Registry

To install the Helm Charts from the *application-collection*, you need to log in in to the registry. This needs to be done with the Helm client.

To log in to the Rancher Application Collection, run:

```
helm registry login dp.apps.rancher.io/charts -u <yourUser> -p <your-token>
```

The login process is needed for the following application installations:

- Cert-Manager [Section 6.4.1, “Installing the application”](#) (page 18)
- MetalLB [Section 9.2.1.2, “Installing MetalLB”](#) (page 33)
- Redis [Section 9.3.1.1, “Deploying the chart”](#) (page 35)
- PostgreSQL [Section 9.4.2.2, “Installing the application”](#) (page 37)

11.3 Using self-signed certificates

In this chapter we will explain how to create self-signed certificates and how to make them available within Kubernetes. We will describe two possible solutions to do this. You can create everything on the operation system layer or you also can use cert-manager in your downstream cluster.

11.3.1 Creating self-signed certificates



Warning

We strongly advise against using self-signed certificates in production environments.

The first step is to create a certificate authority (hereinafter referred to as CA) with a key and certificate. The following excerpt provides an example of how to create a CA with a passphrase of your choice:

```
openssl req -x509 -sha256 -days 1825 -newkey rsa:2048 -keyout rootCA.key -out rootCA.crt  
-passout pass:<ca-passphrase> -subj "/C=DE/ST=BW/L=Nuremberg/O=SUSE"
```

This will generate the files *rootCA.key* and *rootCA.crt*. The server certificate requires a certificate signing request (hereinafter referred to as CSR). The following excerpt shows how to create such a CSR:

```
openssl req -newkey rsa:2048 -keyout domain.key -out domain.csr -passout pass:<csr-passphrase> -subj "/C=DE/ST=BW/L=Nuremberg/O=SUSE"
```

Before you can sign the CSR, you need to add the DNS names of your Kubernetes Services to the CSR. Therefore, create a file with the content below and replace the **<servicename>** and **<namespace>** with the name of your Kubernetes service and the namespace in which it is placed:

```
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
subjectAltName = @alt_names
[alt_names]
DNS.1 = <servicename>.<namespace>.svc.cluster.local
DNS.2 = <AltService>.<AltNamespace>.svc.cluster.local
```

You can now use the previously created files *rootCA.key* and *rootCA.crt* with the extension file to sign the CSR. The example below shows how to do that by passing the extension file (here called *domain.ext*):

```
openssl x509 -req -CA rootCA.crt -CAkey rootCA.key -in domain.csr -out server.pem -days 365 -CAcreateserial -extfile domain.ext -passin pass:<ca-passphrase>
```

This creates a file called *server.pem*, which is the certificate to be used for your application. Your *domain.key* is still encrypted at this point, but the application requires an unencrypted server key. To decrypt it, run the provided command, which will generate the *server.key*.

```
openssl rsa -passin pass:<csr-passphrase> -in domain.key -out server.key
```

Some applications (like Redis) require a full certificate chain to operate. To get a full certificate chain, link the generated file *server.pem* with the file *rootCA.crt* as follows:

```
cat server.pem rootCA.crt > chained.pem
```

You should then have the files *server.pem*, *server.key* and *chained.pem* that can be used for your applications such as Redis or PostgreSQL.

11.3.2 Uploading certificates to Kubernetes

To use certificate files in Kubernetes, you need to save them as so-called **secrets**. For an example of uploading your certificates to Kubernetes, see the following excerpt:

```
kubectl -n <namespace> create secret generic <certName> --from-file=./root.pem --from-file=./server.pem --from-file=./server.key
```



Note

All applications are expecting to have the secret to be used in the same namespace as the application.

11.3.3 Using cert-manager

cert-manager needs to be available in your Downstream Cluster. To install cert-manager in your downstream cluster, you can follow the same installation steps outlined in the Rancher Prime installation section. First, create a *selfsigned-issuer.yaml* file:

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: selfsigned-issuer
spec:
  selfSigned: {}
```

Then create a Certificate Ressource for the CA called *my-ca-cert.yaml*:

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: my-ca-cert
  namespace: cert-manager
spec:
  isCA: true
  commonName: <cluster-name>.cluster.local
  secretName: my-ca-secret
  issuerRef:
    name: selfsigned-issuer
    kind: ClusterIssuer
  dnsNames:
    - "<cluster-name>.cluster.local"
    - "*,<cluster-name>.cluster.local"
```

For creating a *ClusterIssuer* using the Generated CA, create the *my-ca-issuer.yaml* file:

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: my-ca-issuer
spec:
  ca:
    secretName: my-ca-secret
```

The last resource you need to create is the certificate itself. This certificate is signed by your created CA. You can name the yaml file *application-name-certificate.yaml*.

```
kind: Certificate
metadata:
  name: <application-name>
  namespace: <application namespace> // need to be created manually.
spec:
  dnsNames:
    - <application-name>.cluster.local
  issuerRef:
    group: cert-manager.io
    kind: ClusterIssuer
    name: my-ca-issuer
  secretName: <application-name>
  usages:
    - digital signature
    - key encipherment
```

Apply the yaml file to your Kubernetes cluster.


```
kubectl apply -f selfsigned-issuer.yaml
kubectl apply -f my-ca-cert.yaml
kubectl apply -f my-ca-issuer.yaml
kubectl apply -f application-name-certificate.yaml
```

When you deploy your applications via Helm Charts, you can use the generated certificate. In the Kubernetes Secret Certificate, three files are stored. These are the files *tls.crt*, *tls.key* and *ca.crt*, which you can use in the *values.yaml* file of your application.

12 Legal notice

Copyright © 2006–2025 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".

SUSE, the SUSE logo and YaST are registered trademarks of SUSE LLC in the United States and other countries. For SUSE trademarks, see <https://www.suse.com/company/legal/> .

Linux is a registered trademark of Linus Torvalds. All other names or trademarks mentioned in this document may be trademarks or registered trademarks of their respective owners.

Documents published as part of the SUSE Best Practices series have been contributed voluntarily by SUSE employees and third parties. They are meant to serve as examples of how particular actions can be performed. They have been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. SUSE cannot verify that actions described in these documents do what is claimed or whether actions described have unintended consequences. SUSE LLC, its affiliates, the authors, and the translators may not be held liable for possible errors or the consequences thereof.

Below we draw your attention to the license under which the articles are published.

13 GNU Free Documentation License

Copyright © 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 noncommercially. Secondly, 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 in order 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 noncommercially, 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 <http://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.