



Deployment Guide



Deployment Guide: This guide describes deployment for SUSE CaaS Platform 4.0.3.

by Markus Napp and Nora Kořánová

Publication Date: 2022-05-02

SUSE LLC

1800 South Novell Place

Provo, UT 84606

USA

<https://documentation.suse.com> 

Contents

v

About This Guide vi

- 1 Required Background vi
- 2 Available Documentation vi
- 3 Feedback vii
- 4 Documentation Conventions vii

1 Requirements 1

- 1.1 Platform 1

2 Nodes 2

- 2.1 Hardware 3
 - Management Workstation 3 • Storage Sizing 3 • Storage Performance 5
- 2.2 Networking 6
 - Sub-Network Sizing 6 • Ports 7 • IP Addresses 8 • IP Forwarding 8 • Communication 9 • Performance 9 • Security 9

3 Deployment Scenarios 10

- 3.1 Airgapped deployment 10
 - Process Checklist 10 • Concepts 11 • Requirements 14 • RPM Repository Mirror 18 • Updating RPM Repository Mirror 19 • Container Registry Mirror 20 • Helm Chart Repository Mirror 26 • Updating Registry Mirror And Helm Charts 27 • Deploying SUSE CaaS Platform 29 • Troubleshooting 30

4 Deployment Instructions 31

- 4.1 Deployment Preparations 31
 - Basic SSH Key Configuration 31 • Product Key 33 • Installation Tools 34 • Load Balancer 35
- 4.2 Deployment on SUSE OpenStack Cloud 44
 - Deploying the Cluster Nodes 45 • Logging in to the Cluster Nodes 50 • Container Runtime Proxy 52
- 4.3 Deployment on VMware 52
 - Environment Description 52 • Setup with AutoYaST 53 • Setup using VMWare Template 53 • Deploying VMs from the Template 77 • Container Runtime Proxy 81
- 4.4 Deployment on Bare Metal 82
 - Environment Description 82 • AutoYaST Preparation 83 • Provisioning the Cluster Nodes 85 • Container Runtime Proxy 86
- 4.5 Deployment on Existing SLES Installation 86
 - Requirements 86 • Adding SUSE CaaS Platform repositories 87

5 Bootstrapping the Cluster 89

- 5.1 Preparation 89
 - Install skuba 89 • Container Runtime Proxy 89
- 5.2 Cluster Deployment 90
 - Initializing the Cluster 90 • Configuring Kubernetes Services 91 • Cluster Configuration 92 • Prevent Nodes with Any Prometheus Alerts from Being Rebooted 97 • Cluster Bootstrap 98
- 5.3 Using kubectl 99

6 Network Security 102

- 6.1 Cilium 102

A GNU Licenses 104

- A.1 GNU Free Documentation License 104



Warning

This document is a work in progress.

The content in this document is subject to change without notice.



Note

This guide assumes a configured SUSE Linux Enterprise 15 SP1 environment.

Copyright © 2006 — 2019 SUSE LLC and contributors. All rights reserved.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or (at your option) version 1.3; with the Invariant Section being this copyright notice and license. A copy of the license version 1.2 is included in the section entitled “GNU Free Documentation License”.

For SUSE trademarks, see <http://www.suse.com/company/legal/>¹. All other third-party trademarks are the property of their respective owners. Trademark symbols (®, ™, etc.) denote trademarks of SUSE and its affiliates. Asterisks (*) denote third-party trademarks.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

About This Guide

1 Required Background

To keep the scope of these guidelines manageable, certain technical assumptions have been made. These documents are not aimed at beginners in Kubernetes usage and require extensive knowledge of

- You have some computer experience and are familiar with common technical terms.
- You are familiar with the documentation for your system and the network on which it runs.
- You have a basic understanding of Linux systems.
- You have an understanding of how to follow instructions aimed at experienced Linux administrators and can fill in gaps with your own research.
- You understand how to plan, deploy and manage Kubernetes applications.

2 Available Documentation

We provide HTML and PDF versions of our books in different languages. Documentation for our products is available at <https://documentation.suse.com>, where you can also find the latest updates and browse or download the documentation in various formats.

The following documentation is available for this product:

Architecture Guide

The SUSE CaaS Platform Architecture Guide gives you a rough overview of the software architecture. It is as of yet incomplete and will change infrequently.

Deployment Guide

The SUSE CaaS Platform Deployment Guide gives you details about installation and configuration of SUSE CaaS Platform along with a description of architecture and minimum system requirements.

Quick Start Guide

The SUSE CaaS Platform Quick Start guides you through the installation of a minimum cluster in the fastest way possible.

Admin Guide

The SUSE CaaS Platform Admin Guide discusses authorization, updating clusters and individual nodes, monitoring, use of Helm and Tiller, the Kubernetes dashboard, and integration with SUSE Enterprise Storage.

3 Feedback

Several feedback channels are available:

Bugs and Enhancement Requests

For services and support options available for your product, refer to <http://www.suse.com/support/>.

To report bugs for a product component, go to <https://scc.suse.com/support/requests>, log in, and click *Create New*.

Mail

We want to hear your comments about and suggestions for this manual and the other documentation included with this product. For feedback on the documentation of this product, you can send a mail to doc-team@suse.com. Make sure to include the document title, the product version and the publication date of the documentation. To report errors or suggest enhancements, provide a concise description of the problem and refer to the respective section number and page (or URL).

4 Documentation Conventions

The following notices and typographical conventions are used in this documentation:

- /etc/passwd : directory names and file names
- <PLACEHOLDER>: replace <PLACEHOLDER> with the actual value
- PATH: the environment variable PATH
- ls, --help: commands, options, and parameters
- user : users or groups
- package name : name of a package

- `Alt`, `Alt-F1` : a key to press or a key combination; keys are shown in uppercase as on a keyboard
- *File > Save As* : menu items, buttons
- *Dancing Penguins* (Chapter *Penguins*, ↑Another Manual): This is a reference to a chapter in another manual.
- Commands that must be run with root privileges. Often you can also prefix these commands with the `sudo` command to run them as non-privileged user.

```
sudo command
```

- Commands that can be run by non-privileged users.

```
command
```

- Notices:



Warning

Vital information you must be aware of before proceeding. Warns you about security issues, potential loss of data, damage to hardware, or physical hazards.



Important

Important information you should be aware of before proceeding.



Note

Additional information, for example about differences in software versions.



Tip

Helpful information, like a guideline or a piece of practical advice.

1 Requirements

1.1 Platform

Currently we support:

- SUSE OpenStack Cloud 8
- VMware ESXi 6.7.0.20000
- Bare Metal x86_64

2 Nodes

You will need at least two machines:

- 1 master node
- 1 worker node

SUSE CaaS Platform 4.0.3 supports deployments with a single or multiple master nodes. Production environments must be deployed with multiple master nodes for resilience.

All communication to the cluster is done through a load balancer talking to the respective nodes. For that reason any failure tolerant environment must provide at least two load balancers for incoming communication.

The minimal viable failure tolerant production environment configuration consists of:

CLUSTER NODES:

- 3 master nodes
- 2 worker nodes



Important: Dedicated Cluster Nodes

All cluster nodes must be dedicated (virtual) machines reserved for the purpose of running SUSE CaaS Platform.

ADDITIONAL SYSTEMS:

- Fault tolerant load balancing solution
(for example SUSE Linux Enterprise High Availability Extension with pacemaker and haproxy)
- 1 management workstation

2.1 Hardware

2.1.1 Management Workstation

In order to deploy and control a SUSE CaaS Platform cluster you will need at least one machine capable of running `skuba`. This typically is a regular desktop workstation or laptop running SLE 15 SP1 or later.

The `skuba` CLI package is available from the SUSE CaaS Platform module. You will need a valid SUSE Linux Enterprise and SUSE CaaS Platform subscription to install this tool on the workstation.

Important: Time Synchronization

It is vital that the management workstation runs an NTP client and that time synchronization is configured to the same NTP servers, which you will use later to synchronize the cluster nodes.

2.1.2 Storage Sizing

The storage sizes in the following lists are absolute minimum configurations.

Sizing of the storage for worker nodes depends largely on the expected amount of container images, their size and change rate. The basic operating system for all nodes might also include snapshots (when using `bt rfs`) that can quickly fill up existing space.

We recommend provisioning a separate storage partition for container images on each (worker) node that can be adjusted in size when needed. Storage for `/var/lib/containers` on the worker nodes should be approximately 50GB in addition to the base OS storage.

2.1.2.1 Master Nodes

Up to 5 worker nodes (**minimum**):

- Storage: 50 GB +
- (v)CPU: 2

- RAM: 4 GB
- Network: Minimum 1Gb/s (faster is preferred)

Up to 10 worker nodes:

- Storage: 50 GB +
- (v)CPU: 2
- RAM: 8 GB
- Network: Minimum 1Gb/s (faster is preferred)

Up to 100 worker nodes:

- Storage: 50 GB +
- (v)CPU: 4
- RAM: 16 GB
- Network: Minimum 1Gb/s (faster is preferred)

Up to 250 worker nodes:

- Storage: 50 GB +
- (v)CPU: 8
- RAM: 16 GB
- Network: Minimum 1Gb/s (faster is preferred)

Important

Using a minimum of 2 (v)CPUs is a hard requirement, deploying a cluster with less processing units is not possible.

2.1.2.2 Worker nodes

Important

The worker nodes must have sufficient memory, CPU and disk space for the Pods/containers/applications that are planned to be hosted on these workers.

A worker node requires the following resources:

- CPU cores: 1.250
- RAM: 1.2 GB

Based on these values, the **minimal** configuration of a worker node is:

- Storage: Depending on workloads, minimum 20-30 GB to hold the base OS and required packages. Mount additional storage volumes as needed.
- (v)CPU: 2
- RAM: 2 GB
- Network: Minimum 1Gb/s (faster is preferred)

Calculate the size of the required (v)CPU by adding up the base requirements, the estimated additional essential cluster components (logging agent, monitoring agent, configuration management, etc.) and the estimated CPU workloads:

- 1.250 (base requirements) + 0.250 (estimated additional cluster components) + estimated workload CPU requirements

Calculate the size of the RAM using a similar formula:

- 1.2 GB (base requirements) + 500 MB (estimated additional cluster components) + estimated workload RAM requirements



Note

These values are provided as a guide to work in most cases. They may vary based on the type of the running workloads.

2.1.3 Storage Performance

For master and worker nodes you must ensure storage performance of at least 500 sequential IOPS with disk bandwidth depending on your cluster size.

"Typically 50 sequential IOPS (for example, a 7200 RPM disk) is required.
For heavily loaded clusters, 500 sequential IOPS (for example, a typical local SSD

or a high performance virtualized block device) is recommended."

"Typically 10MB/s will recover 100MB data within 15 seconds.
For large clusters, 100MB/s or higher is suggested for recovering 1GB data within 15 seconds."

<https://github.com/etcd-io/etcd/blob/master/Documentation/op-guide/hardware.md#disks> ↗

2.2 Networking

The management workstation needs at least the following networking permissions:

- SSH access to all machines in the cluster
- Access to the `apiserver` (the load balancer should expose it, port `6443`), that will in turn talk to any master in the cluster
- Access to Dex on the configured `NodePort` (the load balancer should expose it, port `32000`) so when the OIDC token has expired, `kubectl` can request a new token using the refresh token

2.2.1 Sub-Network Sizing



Important

The service subnet and pod subnet must not overlap.

Please plan generously for workload and the expected size of the networks before bootstrapping.

The default pod subnet is `10.244.0.0/16`. It allows for 65536 IP addresses overall. Assignment of CIDR's is by default `/24` (254 usable IP addresses per node).

The default node allocation of `/24` means a hard cluster node limit of 256 since this is the number of `/24` ranges that fit in a `/16` range.

Depending on the size of the nodes that you are planning to use (in terms of resources), or on the number of nodes you are planning to have, the CIDR can be adjusted to be bigger on a per node basis but the cluster would accommodate less nodes overall.

If you are planning to use more or less pods per node or have a higher number of nodes, you can adjust these settings to match your requirements. Please make sure that the networks are suitably sized to adjust to future changes in the cluster.

You can also adjust the service subnet size, this subnet must not overlap with the pod CIDR, and it should be big enough to accommodate all services.

For more advanced network requirements please refer to: <https://docs.cilium.io/en/v1.6/concepts/ipam/#address-management> ↗

2.2.2 Ports

Node	Port	Protocol	Accessibility	Description
All nodes	22	TCP	Internal	SSH (required in public clouds)
	4240	TCP	Internal	Cilium health check
	8472	UDP	Internal	Cilium VXLAN
	10250	TCP	Internal	Kubelet (API server → kubelet communication)
	10256	TCP	Internal	kube-proxy health check
	30000 - 32767	TCP + UDP	Internal	Range of ports used by Kubernetes when allocating services of type <u>Node-Port</u>
	32000	TCP	External	Dex (OIDC Connect)

Node	Port	Protocol	Accessibility	Description
	32001	TCP	External	Gangway (RBAC Authenticate)
Masters	2379	TCP	Internal	etcd (client communication)
	2380	TCP	Internal	etcd (server-to-server traffic)
	6443	TCP	Internal / External	Kubernetes API server

2.2.3 IP Addresses



Warning

Using IPv6 addresses is currently not supported.

All nodes must be assigned static IPv4 addresses, which must not be changed manually afterwards.



Important

Plan carefully for required IP ranges and future scenarios as it is not possible to reconfigure the IP ranges once the deployment is complete.

2.2.4 IP Forwarding

The [Kubernetes networking model](https://kubernetes.io/docs/concepts/cluster-administration/networking/) (https://kubernetes.io/docs/concepts/cluster-administration/networking/) requires that your nodes have IP forwarding enabled in the kernel. `skuba` checks this value when installing your cluster and installs a rule in `/etc/sysctl.d/90-skuba-net-ipv4-ip-forward.conf` to make it persistent.

Other software can potentially install rules with higher priority overriding this value and causing machines to not behave as expected after rebooting.

You can manually check if this is enabled using the following command:

```
# sysctl net.ipv4.ip_forward  
  
net.ipv4.ip_forward = 1
```

`net.ipv4.ip_forward` must be set to `1`. Additionally, you can check in what order persisted rules are processed by running `sysctl --system -a`.

2.2.5 Communication

Please make sure that all your Kubernetes components can communicate with each other. This might require the configuration of routing when using multiple network adapters per node.

Refer to: <https://kubernetes.io/docs/setup/independent/install-kubeadm/#check-network-adapters>.

Configure firewall and other network security to allow communication on the default ports required by Kubernetes: <https://kubernetes.io/docs/setup/independent/install-kubeadm/#check-required-ports>

2.2.6 Performance

All master nodes of the cluster must have a minimum 1Gb/s network connection to fulfill the requirements for etcd.

```
"1GbE is sufficient for common etcd deployments. For large etcd clusters,  
a 10GbE network will reduce mean time to recovery."
```

<https://github.com/etcd-io/etcd/blob/master/Documentation/op-guide/hardware.md#network>

2.2.7 Security

Do not grant access to the kubeconfig file or any workstation configured with this configuration to unauthorized personnel. In the current state, full administrative access is granted to the cluster.

Authentication is done via the kubeconfig file generated during deployment. This file will grant full access to the cluster and all workloads. Apply best practices for access control to workstations configured to administer the SUSE CaaS Platform cluster.

3 Deployment Scenarios

3.1 Airgapped deployment

An air gapped deployment is defined by not allowing any direct connection to the Internet or external networks from the cluster during setup or runtime.

All data that is transferred to the cluster must be transferred in a secure fashion.



Note

Air gapped deployment can be performed with any of the other deployment types and includes a set of steps that need to be performed before, or during the deployment steps of the concrete deployment.



Important: Scope Of This Document

This document focuses on providing mirrors for the resources provided by SUSE and required for basic SUSE CaaS Platform functionality. If you require additional functionality, you can use these instructions as an example on how to provide additional mirrors.

Providing a full set of mirroring instructions, for all usage scenarios, is beyond the scope of this document.

3.1.1 Process Checklist

The steps that must be performed for an air gapped installation are:

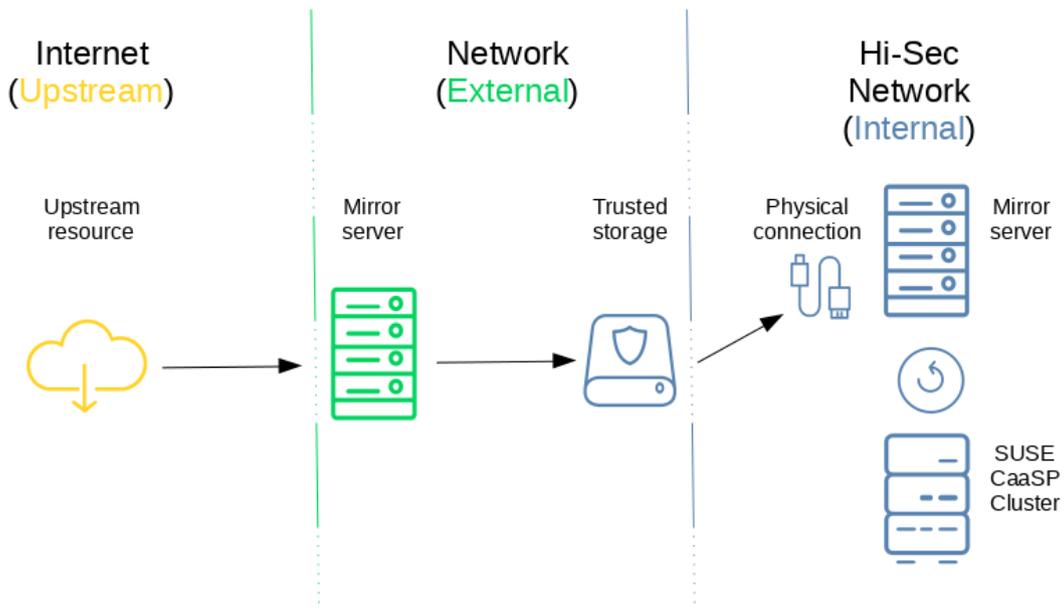
1. Read the concepts section.
Section 3.1.2, "Concepts"
2. Deploy mirror servers on external and internal networks.
Section 3.1.3.1, "Mirror Servers"
3. Install Repository Mirroring Tool (RMT) on servers.
Section 3.1.4, "RPM Repository Mirror"
4. Configure container image registry on servers.
Section 3.1.6, "Container Registry Mirror"

5. Configure Helm Chart repository on internal mirror.
Section 3.1.7, "Helm Chart Repository Mirror"
6. Perform the Repository Mirroring Tool (RMT) update procedure to populate the RPM repository.
Section 3.1.5, "Updating RPM Repository Mirror"
7. Perform the shared update procedure to populate the Helm chart repository and registry services.
Section 3.1.5, "Updating RPM Repository Mirror"
8. Deploy SUSE CaaS Platform and configure the nodes to use the respective services on the internal network.
Section 3.1.9, "Deploying SUSE CaaS Platform"
RPM Packages: *Section 3.1.4.2, "Client Configuration"*
Helm Charts: *Section 3.1.7.2, "Client Configuration"*
Container Images: *Section 3.1.6.2, "Client Configuration"*

3.1.2 Concepts

3.1.2.1 Network Separation

For an air gapped scenario we assume a network separation into three logical parts.



Upstream

Outside the controlled network.

External

Inside the controlled network, outside the air gapped network.

Internal

Inside the air gapped network.

The following instructions will use these three terms to refer to parts of the infrastructure. For example: "internal mirror" refers to the mirroring server on the air gapped network. The terms air gapped and internal will be used interchangeably.

3.1.2.2 Mirrored Resources

In order to disconnect SUSE CaaS Platform from the external network, we provide ways for the components to retrieve data from alternative sources inside the internal (air gapped) network. You will need to create a mirror server inside the internal network; which acts as a replacement for the default sources.

The three main sources that must be replaced are:

- SUSE Linux Enterprise Server RPM packages
Provided by the SUSE package repositories
- Helm installation charts
Provided by the SUSE helm chart repository (<https://kubernetes-charts.suse.com/>)
- Container images
Provided by the SUSE container registry (<https://registry.suse.com>)

You will provide replacements for these resources on a dedicated server inside your internal (air gapped) network.

The internal mirror must be updated with data retrieved from the original upstream sources; in a trusted and secure fashion. To achieve this, you will need an additional mirroring server outside of the air gapped network which acts as a first stage mirror and allows retrieving data from the internet.

Updating of mirrors happens in three stages.

1. Update the external mirror from upstream.
2. Transfer the updated data onto a trusted storage device.
3. Update the internal mirror from the trusted storage device.

Once the replacement sources are in place, the key components are reconfigured to use the mirrors as their main sources.

3.1.2.3 RPM Package Repository Mirroring

Mirroring of the RPM repositories is handled by the [Repository Mirroring Tool](https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#book-rmt) (<https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#book-rmt>) for SUSE Linux Enterprise Server 15. The tool provides functionality that mirrors the upstream SUSE package repositories on the local network. This is intended to minimize reliance on SUSE infrastructure for updating large volumes of machines. The air gapped deployment uses the same technology to provide the packages locally for the air gapped environment.

SUSE Linux Enterprise Server bundles software packages into so called modules. You must enable the [SUSE CaaS Platform](#), [SUSE Linux Enterprise Server](#) and [Containers Module](#) modules in addition to the modules enabled by default. All enabled modules need to be mirrored inside the air gapped network in order to provide the necessary software for other parts of this scenario. Repository Mirroring Tool (RMT) will provide a repository server that holds the packages and related metadata for SUSE Linux Enterprise Server; to install them like from the upstream repository. Data is synchronized once a day to the external mirror automatically or can be forced via the CLI.

You can copy this data to your trusted storage at any point and update the internal mirror.

3.1.2.4 Helm Chart and Container Image Mirroring

SUSE CaaS Platform uses [Helm](https://www.helm.sh/) (<https://www.helm.sh/>) as one method to install additional software on the cluster. The logic behind this relies on [Charts](#), which are configuration files that tell Kubernetes how to deploy software and its dependencies. The actual software installed using this method is delivered as [container images](#). The download location of the container image is stored inside the Helm chart.

Container images are provided by SUSE and others on so called registries. The SUSE container registry is used to update the SUSE CaaS Platform components.

To mirror container images inside the air gapped environment, you will run two container image registry services that are used to pull and in turn serve these images. The registry service is shipped as a container image itself.

Helm charts are provided independently from container images and can be developed by any number of sources. Please make sure that you trust the origin of container images referenced in the helm charts.

We provide [helm-mirror](https://github.com/openSUSE/helm-mirror) (<https://github.com/openSUSE/helm-mirror>)  to allow downloading all charts present in a chart repository in bulk and moreover to extract all container image URLs from the charts. [skopeo](https://github.com/containers/skopeo) (<https://github.com/containers/skopeo>)  is used to download all the images referred to in the Helm charts from their respective registry.

Helm charts will be provided to the internal network by a webserver and refer to the container images hosted on the internal registry mirror.

Once mirroring is configured, you will not have to modify Dockerfile(s) or Kubernetes manifests to use the mirrors. The requests are passed through the container engine which forwards them to the configured mirrors. For example: All images with a prefix `registry.suse.com/` will be automatically pulled from the configured (internal) mirror instead.

For further information on registry mirror configuration, refer to https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_container_registries_for_cri_o .

3.1.3 Requirements

3.1.3.1 Mirror Servers



Note: Shared Mirror Server

If you have multiple SUSE CaaS Platform clusters or a very large number of nodes accessing the mirrors, you should increase the sizing of CPU/RAM.

Storage sizing depends on your intended update frequency and data retention model. If you want to keep snapshots or images of repository states at various points, you must increase storage size accordingly.

You will need to provide and maintain at least two machines in addition to your SUSE CaaS Platform cluster. These mirror servers will reside on the external part of your network and the internal (air gapped) network respectively.

For more information on the requirements of a SUSE Linux Enterprise 15 server, refer to: [Installation Preparation \(https://documentation.suse.com/sles/15-SP1/single-html/SLES-deployment/#part-prep\)](https://documentation.suse.com/sles/15-SP1/single-html/SLES-deployment/#part-prep).

External

This machine will host the Repository Mirroring Tool (RMT) for RPM packages and the container image registry for container images.

- 1 Host machines for the mirror servers.
 - SLES 15
 - 2 (v)CPU
 - 4 GB RAM
 - 250 GB Storage

Internal (Air gapped)

This machine will host the Repository Mirroring Tool (RMT) for RPM packages, and container image registry for container images as well as the Helm chart repository files.

- 1 Host machines for the mirror servers.
 - SLES 15
 - 2 (v)CPU
 - 8 GB RAM
 - 500 GB Storage



Important: Adjust Number Of Mirror Servers

This scenario description does not contain any fallback contingencies for the mirror servers. Add additional mirror servers (behind a load balancer) if you require additional reliability/availability.

PROCEDURE: PROVISION MIRROR SERVERS

1. Set up two SUSE Linux Enterprise Server 15 machines (<https://documentation.suse.com/sles/15-SP1/single-html/SLES-installquick/#art-sle-installquick>)  one on the internal network and one on the air gapped network.
2. Make sure you have enabled the Containers module (<https://documentation.suse.com/sles/15-SP1/single-html/SLES-dockerquick/#Preparation>)  on both servers.
3. Make sure you have Repository Mirroring Tool installed (<https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#cha-rmt-installation>)  on both server.

3.1.3.2 Networking



Note: Additional Port Configuration

If you choose to add more container image registries to your internal network, these must run on different ports than the standard registry running on 5000. Configure your network to allow for this communication accordingly.

3.1.3.2.1 Ports

The external mirror server must be able to exchange outgoing traffic with upstream sources on ports 80 and 443.

All members of the SUSE CaaS Platform cluster must be able to communicate with the internal mirror server(s) within the air gapped network. You must configure at least these ports in all firewalls between the cluster and the internal mirror:

- 80 HTTP - Repository Mirroring Tool (RMT) Server and Helm chart repository mirror
- 443 HTTPS - Repository Mirroring Tool (RMT) Server and Helm chart repository mirror
- 5000 HTTPS - Container image registry

3.1.3.2.2 Hostnames / FQDN

You need to define fully qualified domain names (FQDN) for both of the mirror servers in their respective network. These hostnames are the basis for the required SSL certificates and are used by the components to access the respective mirror sources.

3.1.3.2.3 SSL Certificates

You will need SSL/TLS certificates to secure services on each server.

On the air gapped network, certificates need to cover the hostname of your server and the subdomains for the registry (registry.) and helm chart repository (charts.). You must add corresponding aliases to the certificate.



Tip

You can use wildcard certificates to cover the entire hostname.

The certificates can be replaced with the self-signed certificate, or you can re-use the certificates created by Repository Mirroring Tool (RMT) during the setup of the mirror servers.

Place the certificate, CA certificate and key file in /etc/rmt/ssl/ as rmt-server.crt, rmt-ca.cert, and rmt-server.key.

These certificates can be re-used by all three mirror services.

Make sure the CA certificate is available to SUSE CaaS Platform system wide; so they can be used by the deployed components.

You can add system wide certificates with following commands on all nodes:

```
sudo cp /etc/rmt/ssl/rmt-ca.crt /etc/pki/trust/anchors/  
sudo update-ca-certificates
```

3.1.3.3 Trusted Storage

Transferring data from the external network mirror to the internal mirror can be performed in many ways. The most common way is portable storage (USB keys or external hard drives).

Sizing of the storage is dependent on the number of data sources that need to be stored. Container images can easily measure several Gigabytes per item; although they are generally smaller for Kubernetes related applications. The overall size of any given RPM repository is at least tens of Gigabytes. For example: At the time of writing, the package repository for SUSE Linux Enterprise Server contains approximately 36 GB of data.

The storage must be formatted to a file system type supporting files larger than 4 GB.

We recommend external storage with at least 128 GB.



Note: Mount Point For Storage In Examples

In the following procedures, we will assume the storage (when connected) is mounted on `/mnt/storage` . Please make sure to adjust the mountpoint in the respective command to where the device is actually available.



Note: Handling Of Trusted Storage

Data integrity checks, duplication, backup, and secure handling procedures of trusted storage are beyond the scope of this document.

3.1.4 RPM Repository Mirror

3.1.4.1 Mirror Configuration



Note: Deploy The Mirror Before SUSE CaaS PlatformCluster Deployment

The mirror on the air gapped network must be running and populated before

PROCEDURE: CONFIGURE THE EXTERNAL MIRROR

1. Connect the external mirror to SUSE Customer Center as described in [these instructions](https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#sec-rmt-mirroring-credentials) (<https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#sec-rmt-mirroring-credentials>) [↗](#).

Important: Mirror Registration

During the installation of Repository Mirroring Tool (RMT) you will be asked for login credentials. On the external mirror, you need to enter your SUSE Customer Center login credentials to register. On the internal mirror, you can skip the SUSE Customer Center login since the registration will not be possible without an internet connection to SUSE Customer Center .

2. You need to disable the automatic repository sync on the internal server. Otherwise it will attempt to download information from SUSE Customer Center which can not be reached from inside the air gapped network.

```
sudo systemctl stop rmt-server-sync.timer
sudo systemctl disable rmt-server-sync.timer
```

Now you need to perform the update procedure to do an initial sync of data between the upstream sources and the external mirror and the external and internal mirrors. Refer to: [Section 3.1.5, "Updating RPM Repository Mirror"](#).

3.1.4.2 Client Configuration

Follow [these instructions \(https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#cha-rmt-client\)](https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#cha-rmt-client) to configure all SUSE CaaS Platform nodes to use the package repository mirror server in the air gapped network.

3.1.5 Updating RPM Repository Mirror

Follow these instructions (<https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#sec-rmt-mirroring-export-import>) to update the external server, transfer the data to a storage device, and use that device to update the air gapped server.

3.1.6 Container Registry Mirror



Note: Mirroring Multiple Image Registries / Chart Repositories

You can mirror images and charts from multiple registries in one shared internal registry. We do not recommend mirroring multiple registries in a shared registry due to the potential conflicts.

We highly recommend running separate helm chart and container registry mirrors for each source registry.

Additional mirror registries must be run on separate mirror servers for technical reasons.

3.1.6.1 Mirror Configuration

The container image registry is provided as a container image itself. You must download the registry container from SUSE and run it on the respective server.



Note: Which images to Mirror

CaaS Platform requires a base set of images to be mirrored, as they contain the core services needed to run the cluster.

This list of base images can be found under the following link: <https://documentation.suse.com/external-tree/en-us/suse-caasp/4/skuba-cluster-images.txt>

Alternatively, the list can be obtained from `skuba` - just run this command on the machine you have `skuba` installed on:

```
skuba cluster images
```

This will print out a list of the images `skuba` is expecting to use on the cluster to be bootstrapped.

Mirror those and setup the `crio`-registries to point to the location they are mirrored at.



Note: Internal Registry Mirror Is Read Only

For security reasons, the internal registry mirror is configured in read-only mode. Therefore, pushing container images to this mirror will not be possible. It can only serve images that were previously pulled and cached by the external mirror and then uploaded to the internal mirror.

You can modify and store your own container images on the external registry and transfer them with the other container images using the same process. If you need to be able to modify and store container images on the internal network, we recommend creating a new registry that will hold these images. The steps needed to run your own full container image registry are not part of this document.

For more information you can refer to: [SLES15 - Docker Open Source Engine Guide: What is Docker Registry? \(https://documentation.suse.com/sles/15-SP1/single-html/SLES-dockerquick/#sec-docker-registry-definition\)](https://documentation.suse.com/sles/15-SP1/single-html/SLES-dockerquick/#sec-docker-registry-definition).

We will re-use the nginx webserver that is running as part of Repository Mirroring Tool (RMT) to act as a reverse proxy for the container image registry service and to serve the chart repository files. This step is not necessary for the external host.

PROCEDURE: SET UP REVERSE PROXY AND VIRTUAL HOST

1. SSH into the internal mirror server.
2. Create a virtual host configuration file `/etc/nginx/vhosts.d/registry-server-https.conf` .

Replace `mymirror.local` with the hostname of your mirror server for which you created the SSL certificates.

```
upstream docker-registry {
    server 127.0.0.1:5000;
}

map $upstream_http_docker_distribution_api_version $docker_distribution_api_version
{
    '' 'registry/2.0';
}

server {
    listen 443    ssl;
    server_name  registry.`mymirror.local`;
```

```

access_log /var/log/nginx/registry_https_access.log;
error_log /var/log/nginx/registry_https_error.log;
root /usr/share/rmt/public;

ssl_certificate /etc/rmt/ssl/rmt-server.crt;
ssl_certificate_key /etc/rmt/ssl/rmt-server.key;
ssl_protocols TLSv1.2 TLSv1.3;

# disable any limits to avoid HTTP 413 for large image uploads
client_max_body_size 0;

location /v2/ {
    # Do not allow connections from docker 1.5 and earlier
    # docker pre-1.6.0 did not properly set the user agent on ping, catch "Go *"
    user agents
    if ($http_user_agent ~ "^(docker\/1\. (3|4|5(?:?!\. [0-9]-dev))|Go )\.*$" ) {
        return 404;
    }

    ## If $docker_distribution_api_version is empty, the header is not added.
    ## See the map directive above where this variable is defined.
    add_header 'Docker-Distribution-API-Version' $docker_distribution_api_version
    always;

    proxy_pass http://docker-registry;
    proxy_set_header Host $http_host; # required for docker
    client's sake
    proxy_set_header X-Real-IP $remote_addr; # pass on real client's IP
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_read_timeout 900;
}
}

```

3. Create a virtual host configuration file [/etc/nginx/vhosts.d/charts-server-https.conf](#) .

Replace [mymirror.local](#) with the hostname of your mirror server for which you created the SSL certificates.

```

server {
    listen 443 ssl;
    server_name charts.`mymirror.local`;

    access_log /var/log/nginx/charts_https_access.log;
    error_log /var/log/nginx/charts_https_error.log;
    root /srv/www/;
}

```

```
ssl_certificate      /etc/rmt/ssl/rmt-server.crt;
ssl_certificate_key  /etc/rmt/ssl/rmt-server.key;
ssl_protocols        TLSv1.2 TLSv1.3;

location /charts {
    autoindex on;
}
}
```

4. Restart nginx for the changes to take effect.

```
sudo systemctl restart nginx
```

PROCEDURE: SET UP THE EXTERNAL MIRROR

1. SSH into the external mirror server.
2. Install `docker` , `helm-mirror` and `skopeo` .

```
sudo zypper in docker helm-mirror skopeo
```

3. Start the docker service and enable it at boot time:

```
sudo systemctl enable --now docker.service
```

4. Pull the registry container image from SUSE .

```
sudo docker pull registry.suse.com/sles12/registry:2.6.2
```

5. Save the pulled image to a `.tar` file.

```
sudo docker save -o /tmp/registry.tar registry.suse.com/sles12/registry:2.6.2
```

6. Connect the trusted storage to the external mirror. Copy the registry image onto the storage.

```
mv /tmp/registry.tar /mnt/storage/registry.tar
```

7. Create basic authentication credentials for the container image registry.

Replace `USERNAME` and `PASSWORD` with proper credentials of your choosing.

```
sudo mkdir -p /etc/docker/registry/{auth,certs}
sudo docker run --entrypoint htpasswd registry.suse.com/sles12/registry:2.6.2 -Bbn
  USERNAME PASSWORD \
> /etc/docker/registry/auth/htpasswd
```

8. Create the `/etc/docker/registry/config.yml` configuration file.



Note

Setting up a required authentication seems to break, when using CRI-O as the client, so the internal registry does not use any authentication.

```
version: 0.1
log:
  fields:
    service: registry
storage:
  cache:
    blobdescriptor: inmemory
  filesystem:
    rootdirectory: /var/lib/registry
http:
  addr: 0.0.0.0:5000
  headers:
    X-Content-Type-Options: [nosniff]
  tls:
    certificate: /etc/rmt/ssl/rmt-server.crt
    key: /etc/rmt/ssl/rmt-server.key
health:
  storagedriver:
    enabled: true
    interval: 10s
  threshold: 3
```

For more details on the configuration, refer to: [Docker Registry: Configuration \(https://docs.docker.com/registry/configuration/\)](https://docs.docker.com/registry/configuration/) ↗

9. Start the registry container.

```
sudo docker run -d -p 5000:5000 -v /etc/rmt/ssl:/etc/rmt/ssl:ro --restart=always --
name registry \
-v /etc/docker/registry:/etc/docker/registry:ro \
-v /var/lib/registry:/var/lib/registry registry.suse.com/sles12/registry:2.6.2
```

PROCEDURE: SET UP INTERNAL MIRROR

1. SSH into the internal mirror server.
2. Install `docker` .

```
sudo zypper in docker
```

3. Start the docker service and enable it at boot time:

```
sudo systemctl enable --now docker.service
```

4. Connect the trusted storage to the internal mirror and load the registry container image to the local file system.

```
sudo docker load -i /mnt/storage/registry.tar
```

5. Create the `/etc/docker/registry/config.yml` configuration file.

```
sudo mkdir -p /etc/docker/registry/
```

```
version: 0.1
log:
  fields:
    service: registry
storage:
  cache:
    blobdescriptor: inmemory
  filesystem:
    rootdirectory: /var/lib/registry
maintenance:
  readonly:
    enabled: true
http:
  addr: 0.0.0.0:5000
  headers:
    X-Content-Type-Options: [nosniff]
  tls:
    certificate: /etc/rmt/ssl/rmt-server.crt
    key: /etc/rmt/ssl/rmt-server.key
health:
  storagedriver:
    enabled: true
    interval: 10s
  threshold: 3
```

For more details on the configuration, refer to: [Docker Registry: Configuration \(https://docs.docker.com/registry/configuration/\)](https://docs.docker.com/registry/configuration/) ↗

6. Start the registry container.

```
sudo docker run -d -p 5000:5000 -v /etc/rmt/ssl:/etc/rmt/ssl:ro --restart=always --
name registry \
-v /etc/docker/registry:/etc/docker/registry:ro \
-v /var/lib/registry:/var/lib/registry registry.suse.com/sles12/registry:2.6.2
```

Now, you should have the registries set up and listening on port 5000 on their respective servers.

3.1.6.2 Client Configuration

Configure /etc/containers/registries.conf to setup the mirroring from registry.suse.com to the internal mirror.

This needs to be done on all cluster nodes:

```
[[registry]]
location = "registry.suse.com"
mirror = [{ location = "internal.mirror"}]
# Optional: if the registry is not secure this can be set
# insecure = true
```

For detailed information about the configuration format see https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_container_registries_for_cri_o.

3.1.7 Helm Chart Repository Mirror

Important

To make use of the helm charts, you must complete *Section 3.1.6, "Container Registry Mirror"*.

The helm charts will require images available from a registry mirror. The charts themselves are served on a simple webserver and do not require any particular configuration apart from basic networking availability and a hostname.

3.1.7.1 Mirror Configuration

Update the Helm chart repository by following the shared update procedure *Section 3.1.8, "Updating Registry Mirror And Helm Charts"*.

3.1.7.2 Client Configuration

Add the webserver as a repo to `helm`.

This step needs to be performed on a machine where Helm is installed and configured to talk to the Tiller server in the SUSE CaaS Platform cluster.

`<SUSE_MIRROR>` will be the user-defined name for this repository listed by Helm. The name of the repository must adhere to [Helm Chart naming conventions \(https://docs.helm.sh/chart_best_practices/#chart-names\)](https://docs.helm.sh/chart_best_practices/#chart-names).

```
helm repo add <SUSE_MIRROR> https://charts.<MYMIRROR.LOCAL>
```

3.1.8 Updating Registry Mirror And Helm Charts



Note: Live Update Of Registry

There is no need to stop the container image registry services while doing the update procedures. All changed images will be re-indexed automatically.

Helm charts and container images must be refreshed in the same procedure, otherwise charts might refer to image versions that are not mirrored or you are mirroring outdated image versions that cause the chart deployment to fail.

PROCEDURE: PULL DATA FROM UPSTREAM SOURCES

1. SSH into the mirror server on the external network.
2. Download all charts from the repository to the file system (e.g. `/tmp/charts`).
This action will download all charts and overwrite the existing Helm chart repository URL. Replace `http://charts.mymirror.local` with the hostname of the webserver providing the Helm chart repository on the internal network.

```
mkdir /tmp/charts
```

```
cd /tmp/charts
```

```
helm-mirror --new-root-url http://charts.mymirror.local https://kubernetes-charts.suse.com /tmp/charts
```

3. Translate the chart information into the `skopeo` format.

```
mkdir /tmp/skopeodata
```

```
helm-mirror inspect-images /tmp/charts -o skopeo=sync.yaml
```



Note: Ignoring Chart Errors

The `helm-mirror` tool will attempt to render and inspect all downloaded charts. Some charts will have values that are filled from environment data on their source repository and produce errors. You can still proceed with this step by using the `--ignore-errors` flag.

4. Download all the referenced images using `skopeo`.

```
skopeo sync --source-yaml sync.yaml dir:/tmp/skopeodata
```

`skopeo` will automatically create a directory named after the hostname of the registry from which you are downloading the images. The final path will be something like `/tmp/skopeodata/registry.suse.com/`.

5. Populate the local registry with the downloaded data.

For `--dest-creds` you must use the credentials you created during [Section 3.1.6.1, "Mirror Configuration"](#).

```
{prompt.user}``skopeo sync --dest-creds USERNAME:PASSWORD \  
dir:/tmp/skopeodata/registry.suse.com/ docker://mymirror.local:5000``
```

6. After the synchronization is done, you can remove the `skopeodata` directory.

```
rm -rf /tmp/skopeodata
```

PROCEDURE: TRANSFER DATA TO SECURE STORAGE

1. Connect the trusted storage to the external mirror.
2. Transfer the container image data to the trusted storage. This will remove all files and directories that are no longer present on the external host from the trusted storage.

```
rsync -aP /var/lib/registry/ /mnt/storage/registry/ --delete
```

3. Transfer the helm chart data to the trusted storage.

```
rsync -aP /tmp/charts/ /mnt/storage/charts --delete
```

4. Connect the trusted storage to the internal mirror.
5. Transfer the container image data to the internal mirror. This will remove all files and directories that are no longer present on the trusted storage from the internal mirror. The target directory is `/var/lib/registry`.

```
rsync -aP /mnt/storage/registry/ /var/lib/registry/ --delete
```

6. Transfer the helm chart data to the internal mirror. This will remove all charts that do not exist on the trusted storage. If you have added any charts to the location manually, please back up these first and restore after the sync from the trusted storage is done.

```
rsync -aP /mnt/storage/charts/ /srv/www/charts/ --delete
```

7. Set the file permissions and ownership to `555` and `nginx:nginx`.

```
sudo chown -R nginx:nginx /srv/www/charts sudo chmod -R 555 /srv/www/charts/
```

PROCEDURE: REFRESH INFORMATION ON THE SUSE CAAS PLATFORMCLUSTER

1. Update the repository information on the machine on which you are using Helm to install software to the cluster.

```
helm repo update
```

You can now deploy additional software on your SUSE CaaS Platform Refer to: <https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#software-installation>.

3.1.9 Deploying SUSE CaaS Platform

Use the SUSE CaaS Platform [Deployment Guide \(https://documentation.suse.com/suse-caasp/4/single-html/caasp-deployment/\)](https://documentation.suse.com/suse-caasp/4/single-html/caasp-deployment/) as usual. Some of the considerations below apply; depending of the chosen installation medium.

Make sure to add the CA certificate of your Repository Mirroring Tool (RMT) server as a systemwide certificate in Velum during the SUSE CaaS Platform deployment.

3.1.9.1 Using the ISO

From YaST register the node against the Repository Mirroring Tool (RMT) server. This will ensure the node zypper repositories are pointed against Repository Mirroring Tool (RMT). Moreover, all the available updates are going to be installed and there is no need to manually install updates right after the installation.

3.1.9.2 Using AutoYaST

Ensure the admin node is registered against Repository Mirroring Tool (RMT), that will ensure the nodes that are provisioned by AutoYaST are registered against Repository Mirroring Tool (RMT) to have all the updates applied.

3.1.10 Troubleshooting

3.1.10.1 Skopeo Fails Because Of Self Signed Certificate

If you are using a self-signed certificate for the registry you can use the `--dest-cert-dir /path/to/the/cert` parameter to provide the certificate.

3.1.10.2 Registering An Existing Node against Repository Mirroring Tool (RMT)

Refer to: *Section 3.1.4.2, "Client Configuration"*.

3.1.10.3 Helm chart connection terminated by HTTPS TO HTTP

When registry mirror is using virtual repository URL. You may need to manually modify the Helm chart index.yaml and point the correct HTTPS base URL.

4 Deployment Instructions

Important

If you are installing over one of the previous milestones, you must remove the RPM repository. SUSE CaaS Platform is now distributed as an extension for SUSE Linux Enterprise and no longer requires the separate repository.

If you do not remove the repository before installation, there might be conflicts with the package dependencies that could render your installation nonfunctional.

4.1 Deployment Preparations

In order to deploy SUSE CaaS Platform you need a workstation running SUSE Linux Enterprise 15 SP1 or similar openSUSE equivalent. This workstation is called the "Management machine". Important files are generated and must be maintained on this machine, but it is not a member of the skuba cluster.

4.1.1 Basic SSH Key Configuration

Note

The use of `ssh-agent` comes with some implications for security that you should take into consideration.

The pitfalls of using `ssh-agent` (<http://rabexc.org/posts/pitfalls-of-ssh-agents>) ↗

To avoid these risks please make sure to either use `ssh-agent -t <TIMEOUT>` and specify a time after which the agent will self-terminate or terminate the agent yourself before logging out by running `ssh-agent -k`.

To log in to the created cluster nodes, you need to configure an SSH key pair and load it into your user's `ssh-agent` program. This is also mandatory in order to be able to use the installation tools `terraform` and `skuba`. In a later deployment step, `skuba` will ensure that the key is distributed across all the nodes and trusted by them. For now, you only need to make sure that an `ssh-agent` is running and that it has the SSH key added:

1. The `ssh-agent` is usually started automatically by graphical desktop environments. If that is not your case, run:

```
eval "$(ssh-agent)"
```

This will start the agent and set environment variables used for agent communication within the current session. This has to be the same terminal session that you run the `skuba` commands in. A new terminal usually requires a new `ssh-agent`. In some desktop environments the `ssh-agent` will also automatically load the SSH keys. To add an SSH key manually, use the `ssh-add` command:

```
ssh-add <PATH_TO_KEY>
```



Tip

If you are adding the SSH key manually, specify the full path. For example: `/home/sles/.ssh/id_rsa`

You can load multiple keys into your agent using the `ssh-add <PATH_TO_KEY>` command. Keys should be password protected as a security measure. The `ssh-add` command will prompt for your password, then the agent caches the private key for a configurable lifetime. The `-t lifetime` option to `ssh-add` specifies a maximum time to cache the key. See `man ssh-add` for more information.



Note: Usage of multiple identities with `ssh-agent`

`Skuba` will try all the identities loaded into the `ssh-agent` until one of them grants access to the node, or until the SSH servers' maximum authentication attempts are exhausted. This could lead to undesired messages in SSH or other security/authentication logs on your local machine.

If you wish to avoid authentication attempts with identities irrelevant to SUSE CaaS Platform, please configure a specific `Host` and `IdentityFile` combination in your local `~/.ssh/config`.

For example:

```
Host caasp.cluster
  HostName cluster.fqdn.domain
  ForwardAgent yes
  User sles
  IdentityFile /home/sles/.ssh/id_rsa
```

4.1.1.1 Forwarding the Authentication Agent Connection

It is also possible to **forward the authentication agent connection** from a host to another, which can be useful if you intend to run skuba on a "jump host" and don't want to copy your private key to this node. This can be achieved using the `ssh -A` command. Please refer to the man page of `ssh` to learn about the security implications of using this feature.

4.1.2 Product Key



Note

The registration code for SUSE CaaS Platform 4 also contains the activation permissions for the underlying SUSE Linux Enterprise operating system. You can use your SUSE CaaS Platform registration code to activate the SUSE Linux Enterprise 15 SP1 subscription during installation.

You need a subscription registration code to use SUSE CaaS Platform. You can retrieve your registration code from SUSE Customer Center.

- Login to <https://scc.suse.com> 
- Navigate to *Products* >]
- Search for "caasp"

- Select *SUSE CaaS Platform > 4.0 > x86_64*
- Copy the *Registration Code*

4.1.3 Installation Tools

For any deployment type you will need skuba and Terraform. These packages are available from the SUSE CaaS Platform package sources. They are provided as an installation "pattern" that will install dependencies and other required packages in one simple step.

Access to the packages requires the SUSE CaaS Platform and Containers extension modules. Enable the modules during the operating system installation or activate them using SUSE Connect.

```
sudo SUSEConnect -r <CAASP_REGISTRATION_CODE> ❶  
sudo SUSEConnect -p sle-module-containers/15.1/x86_64 ❷  
sudo SUSEConnect -p caasp/4.0/x86_64 -r <CAASP_REGISTRATION_CODE> ❸
```

- ❶ Activate SUSE Linux Enterprise
- ❷ Add the free Containers module
- ❸ Add the SUSE CaaS Platform extension with your registration code

Install the required tools:

```
sudo zypper in -t pattern SUSE-CaaS-Management
```

This will install the skuba command line tool and Terraform; as well as various default configurations and examples.



Note: Using a Proxy Server

Sometimes, you need a proxy server to be able to connect to the SUSE Customer Center. To do that:

1. Expose the environmental variable `http_proxy`:

```
export http_proxy=http://PROXY_IP_FQDN:PROXY_PORT
```

2. Replace `<PROXY_IP_FQDN>` by the IP address or a fully qualified domain name (FQDN) of the proxy server and `<PROXY_PORT>` by its port.
3. If you need a proxy server with basic authentication, create the file `/root/.curlrc` with the following content:

```
--proxy-user "<USER>:<PASSWORD>"
```

Replace `<USER>` and `<PASSWORD>` with the credentials of an allowed user for the proxy server.

4.1.4 Load Balancer



Important

Setting up a load balancer is mandatory in any production environment.

SUSE CaaS Platform requires a load balancer to distribute workload between the deployed master nodes of the cluster. A failure-tolerant SUSE CaaS Platform cluster will always use more than one control plane node as well as more than one load balancer, so there isn't a single point of failure.

There are many ways to configure a load balancer. This documentation cannot describe all possible combinations of load balancer configurations and thus does not aim to do so. Please apply your organization's load balancing best practices.

For SUSE OpenStack Cloud, the Terraform configurations shipped with this version will automatically deploy a suitable load balancer for the cluster.

For VMware you must configure a load balancer manually and allow it access to all master nodes created during *Chapter 5, Bootstrapping the Cluster*.

The load balancer should be configured before the actual deployment. It is needed during the cluster bootstrap, and also during upgrades. To simplify configuration, you can reserve the IPs needed for the cluster nodes and pre-configure these in the load balancer.

The load balancer needs access to port 6443 on the apiserver (all master nodes) in the cluster. It also needs access to Gangway port 32001 and Dex port 32000 on all master and worker nodes in the cluster for RBAC authentication.

We recommend performing regular HTTPS health checks on each master node /healthz endpoint to verify that the node is responsive. This is particularly important during upgrades, when a master node restarts the apiserver. During this rather short time window, all requests have to go to another master node's apiserver. The master node that is being upgraded will have to be marked **INACTIVE** on the load balancer pool at least during the restart of the apiserver. We provide reasonable defaults for that on our default openstack load balancer Terraform configuration.

The following contains examples for possible load balancer configurations based on SUSE Linux Enterprise 15 SP1 and nginx or HAProxy.

4.1.4.1 Nginx TCP Load Balancer with Passive Checks

For TCP load balancing, we can use the ngx_stream_module module (available since version 1.9.0). In this mode, nginx will just forward the TCP packets to the master nodes.

The default mechanism is **round-robin** so each request will be distributed to a different server.



Warning

The open source version of Nginx referred to in this guide only allows the use of passive health checks. nginx will mark a node as unresponsive only after a failed request. The original request is lost and not forwarded to an available alternative server.

This load balancer configuration is therefore only suitable for testing and proof-of-concept (POC) environments.

For production environments, we recommend the use of *SUSE Linux Enterprise High Availability Extension 15* (<https://documentation.suse.com/sle-ha/15-SP1/>)

4.1.4.1.1 Configuring the Load Balancer

1. Register SLES and enable the "Server Applications" module:

```
SUSEConnect -r CAASP_REGISTRATION_CODE
SUSEConnect --product sle-module-server-applications/15.1/x86_64
```

2. Install Nginx:

```
zypper in nginx
```

3. Write the configuration in /etc/nginx/nginx.conf:

```
user nginx;
worker_processes auto;

load_module /usr/lib64/nginx/modules/nginx_stream_module.so;

error_log /var/log/nginx/error.log;
error_log /var/log/nginx/error.log notice;
error_log /var/log/nginx/error.log info;

events {
    worker_connections 1024;
    use epoll;
}

stream {
    log_format proxy '$remote_addr [$time_local] '
                    '$protocol $status $bytes_sent $bytes_received '
                    '$session_time "$upstream_addr"';

    error_log /var/log/nginx/k8s-masters-lb-error.log;
    access_log /var/log/nginx/k8s-masters-lb-access.log proxy;

    upstream k8s-masters {
        #hash $remote_addr consistent; ❶
        server master00:6443 weight=1 max_fails=2 fail_timeout=5s; ❷
        server master01:6443 weight=1 max_fails=2 fail_timeout=5s;
        server master02:6443 weight=1 max_fails=2 fail_timeout=5s;
    }
    server {
        listen 6443;
        proxy_connect_timeout 5s;
        proxy_timeout 5s;
        proxy_pass k8s-masters;
    }
}
```

```

upstream dex-backends {
    #hash $remote_addr consistent; ❸
    server master00:32000 weight=1 max_fails=2 fail_timeout=5s; ❹
    server master01:32000 weight=1 max_fails=2 fail_timeout=5s;
    server master02:32000 weight=1 max_fails=2 fail_timeout=5s;
}
server {
    listen 32000;
    proxy_connect_timeout 5s;
    proxy_timeout 5s;
    proxy_pass dex-backends; ❺
}

upstream gangway-backends {
    #hash $remote_addr consistent; ❻
    server master00:32001 weight=1 max_fails=2 fail_timeout=5s; ❼
    server master01:32001 weight=1 max_fails=2 fail_timeout=5s;
    server master02:32001 weight=1 max_fails=2 fail_timeout=5s;
}
server {
    listen 32001;
    proxy_connect_timeout 5s;
    proxy_timeout 5s;
    proxy_pass gangway-backends; ❽
}
}

```

- ❶ ❸ **Note:** To enable session persistence, uncomment the `hash` option so the same client will always be redirected to the same server except if this server is unavailable.
- ❷ ❹ Replace the individual `masterXX` with the IP/FQDN of your actual master nodes (one entry each) in the `upstream k8s-masters` section.
- ❸ ❽ Dex port `32000` and Gangway port `32001` must be accessible through the load balancer for RBAC authentication.

4. Configure `firewalld` to open up port `6443`. As root, run:

```

firewall-cmd --zone=public --permanent --add-port=6443/tcp
firewall-cmd --zone=public --permanent --add-port=32000/tcp
firewall-cmd --zone=public --permanent --add-port=32001/tcp

```

```
firewall-cmd --reload
```

5. Start and enable Nginx. As root, run:

```
systemctl enable --now nginx
```

4.1.4.1.2 Verifying the Load Balancer

! Important

The SUSE CaaS Platform cluster must be up and running for this to produce any useful results. This step can only be performed after *Chapter 5, Bootstrapping the Cluster* is completed successfully.

To verify that the load balancer works, you can run a simple command to repeatedly retrieve cluster information from the master nodes. Each request should be forwarded to a different master node.

From your workstation, run:

```
while true; do skuba cluster status; sleep 1; done;
```

There should be no interruption in the **skuba cluster status** running command.

On the load balancer virtual machine, check the logs to validate that each request is correctly distributed in a round robin way.

```
# tail -f /var/log/nginx/k8s-masters-lb-access.log
10.0.0.47 [17/May/2019:13:49:06 +0000] TCP 200 2553 1613 1.136 "10.0.0.145:6443"
10.0.0.47 [17/May/2019:13:49:08 +0000] TCP 200 2553 1613 0.981 "10.0.0.148:6443"
10.0.0.47 [17/May/2019:13:49:10 +0000] TCP 200 2553 1613 0.891 "10.0.0.7:6443"
10.0.0.47 [17/May/2019:13:49:12 +0000] TCP 200 2553 1613 0.895 "10.0.0.145:6443"
10.0.0.47 [17/May/2019:13:49:15 +0000] TCP 200 2553 1613 1.157 "10.0.0.148:6443"
10.0.0.47 [17/May/2019:13:49:17 +0000] TCP 200 2553 1613 0.897 "10.0.0.7:6443"
```

4.1.4.2 HAProxy TCP Load Balancer with Active Checks

HAProxy is a very powerful load balancer which, unlike nginx, can be used in production. The version used at this date is the 1.8.7.

Important

The configuration of an HA cluster is out of the scope of this document.

Warning: Package Support

HAProxy is available as a supported package with a [SUSE Linux Enterprise High Availability Extension 15](https://documentation.suse.com/sle-ha/15-SP1/) (<https://documentation.suse.com/sle-ha/15-SP1/>)  subscription.

Alternatively, you can install HAProxy from [SUSE Package Hub](https://package-hub.suse.com/packages/haproxy/) (<https://package-hub.suse.com/packages/haproxy/>)  but you will not receive product support for this component.

The default mechanism is **round-robin** so each request will be distributed to a different server. The health-checks are executed every two seconds. If a connection fails, the check will be retried two times with a timeout of five seconds for each request. If no connection succeeds within this interval (2x5s), the node will be marked as DOWN and no traffic will be sent until the checks succeed again.

4.1.4.2.1 Configuring the Load Balancer

1. Register SLES and enable the "Server Applications" module:

```
SUSEConnect -r CAASP_REGISTRATION_CODE
SUSEConnect --product sle-module-server-applications/15.1/x86_64
```

2. Enable the source for the haproxy package:

- If you are using the SUSE Linux Enterprise High Availability Extension

```
SUSEConnect --product sle-ha/15.1/x86_64 -r ADDITIONAL_REGCODE
```

- If you want the free (unsupported) package:

```
SUSEConnect --product PackageHub/15.1/x86_64
```

3. Configure /dev/log for HAProxy chroot (optional)

This step is only required when HAProxy is configured to run in a jail directory (chroot). This is highly recommended since it increases the security of HAProxy.

Since HAProxy is chrooted, it's necessary to make the log socket available inside the jail directory so HAProxy can send logs to the socket.

```
mkdir -p /var/lib/haproxy/dev/ && touch /var/lib/haproxy/dev/log
```

This systemd service will take care of mounting the socket in the jail directory.

```
cat > /etc/systemd/system/bindmount-dev-log-haproxy-chroot.service <<EOF
[Unit]
Description=Mount /dev/log in HAProxy chroot
After=systemd-journald-dev-log.socket
Before=haproxy.service

[Service]
Type=oneshot
ExecStart=/bin/mount --bind /dev/log /var/lib/haproxy/dev/log

[Install]
WantedBy=multi-user.target
EOF
```

Enabling the service will make the changes persistent after a reboot.

```
systemctl enable --now bindmount-dev-log-haproxy-chroot.service
```

4. Install HAProxy:

```
zypper in haproxy
```

5. Write the configuration in /etc/haproxy/haproxy.cfg:



Note

Replace the individual <MASTER_XX_IP_ADDRESS> with the IP of your actual master nodes (one entry each) in the server lines. Feel free to leave the name argument in the server lines (master00 and etc.) as is - it only serves as a label that will show up in the haproxy logs.

```
global
  log /dev/log local0 info ①
  chroot /var/lib/haproxy ②
  user haproxy
  group haproxy
```

```

daemon

defaults
mode      tcp
log       global
option    tcplog
option    redispatch
option    tcpka
retries   2
http-check      expect status 200 ❸
default-server check check-ssl verify none
timeout connect 5s
timeout client 5s
timeout server 5s
timeout tunnel 86400s ❹

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

listen apiserver ❻
bind      *:6443
option    httpchk GET /healthz
server    master00 <MASTER_00_IP_ADDRESS>:6443
server    master01 <MASTER_01_IP_ADDRESS>:6443
server    master02 <MASTER_02_IP_ADDRESS>:6443

listen dex ❼
bind      *:32000
option    httpchk GET /healthz
server    master00 <MASTER_00_IP_ADDRESS>:32000
server    master01 <MASTER_01_IP_ADDRESS>:32000
server    masetr02 <MASTER_02_IP_ADDRESS>:32000

listen gangway ❽
bind      *:32001
option    httpchk GET /
server    master00 <MASTER_00_IP_ADDRESS>:32001
server    master01 <MASTER_01_IP_ADDRESS>:32001
server    master02 <MASTER_02_IP_ADDRESS>:32001

```

- ❶ Forward the logs to systemd journald, the log level can be set to debug to increase verbosity.
- ❷ Define if it will run in a chroot.

- ④ This timeout is set to 24h in order to allow long connections when accessing pod logs or port forwarding.
- ⑤ URL to expose HAProxy stats on port 9000, it is accessible at <http://loadbalancer:9000/stats>
- ③ The performed health checks will expect a 200 return code
- ⑥ Kubernetes apiserver listening on port 6443, the checks are performed against https://MASTER_XX_IP_ADDRESS:6443/healthz
- ⑦ Dex listening on port 32000, it must be accessible through the load balancer for RBAC authentication, the checks are performed against https://MASTER_XX_IP_ADDRESS:32000/healthz
- ⑧ Gangway listening on port 32001, it must be accessible through the load balancer for RBAC authentication, the checks are performed against https://MASTER_XX_IP_ADDRESS:32001/

6. Configure firewalld to open up port 6443. As root, run:

```
firewall-cmd --zone=public --permanent --add-port=6443/tcp
firewall-cmd --zone=public --permanent --add-port=32000/tcp
firewall-cmd --zone=public --permanent --add-port=32001/tcp
firewall-cmd --reload
```

7. Start and enable HAProxy. As root, run:

```
systemctl enable --now haproxy
```

4.1.4.2.2 Verifying the Load Balancer

Important

The SUSE CaaS Platform cluster must be up and running for this to produce any useful results. This step can only be performed after *Chapter 5, Bootstrapping the Cluster* is completed successfully.

To verify that the load balancer works, you can run a simple command to repeatedly retrieve cluster information from the master nodes. Each request should be forwarded to a different master node.

From your workstation, run:

```
while true; do skuba cluster status; sleep 1; done;
```

There should be no interruption in the **skuba cluster status** running command.

On the load balancer virtual machine, check the logs to validate that each request is correctly distributed in a round robin way.

```
# journalctl -flu haproxy
haproxy[2525]: 10.0.0.47:59664 [30/Sep/2019:13:33:20.578] apiserver apiserver/master00
1/0/578 9727 -- 18/18/17/3/0 0/0
haproxy[2525]: 10.0.0.47:59666 [30/Sep/2019:13:33:22.476] apiserver apiserver/master01
1/0/747 9727 -- 18/18/17/7/0 0/0
haproxy[2525]: 10.0.0.47:59668 [30/Sep/2019:13:33:24.522] apiserver apiserver/master02
1/0/575 9727 -- 18/18/17/7/0 0/0
haproxy[2525]: 10.0.0.47:59670 [30/Sep/2019:13:33:26.386] apiserver apiserver/master00
1/0/567 9727 -- 18/18/17/3/0 0/0
haproxy[2525]: 10.0.0.47:59678 [30/Sep/2019:13:33:28.279] apiserver apiserver/master01
1/0/575 9727 -- 18/18/17/7/0 0/0
haproxy[2525]: 10.0.0.47:59682 [30/Sep/2019:13:33:30.174] apiserver apiserver/master02
1/0/571 9727 -- 18/18/17/7/0 0/0
```

4.2 Deployment on SUSE OpenStack Cloud



Note: Preparation Required

You must have completed [Section 4.1, "Deployment Preparations"](#) to proceed.

You will use Terraform to deploy the required master and worker cluster nodes (plus a load balancer) to SUSE OpenStack Cloud and then use the skuba tool to bootstrap the Kubernetes cluster on top of those.

1. Download the SUSE OpenStack Cloud RC file.
 - a. Log in to SUSE OpenStack Cloud.
 - b. Click on your username in the upper right hand corner to reveal the drop-down menu.
 - c. Click on *Download OpenStack RC File v3*.
 - d. Save the file to your workstation.

- e. Load the file into your shell environment using the following command, replacing `DOWNLOADED_RC_FILE` with the name your file:

```
source <DOWNLOADED_RC_FILE>.sh
```

- f. Enter the password for the RC file. This should be same the credentials that you use to log in to SUSE OpenStack Cloud.
2. Get the SLES15-SP1 image.
 - a. Download the pre-built image of SUSE SLES15-SP1 for SUSE OpenStack Cloud from <https://download.suse.com/Download?buildid=OE-3enq3uys~>.
 - b. Upload the image to your SUSE OpenStack Cloud.



Note: The default user is 'sles'

The SUSE SLES15-SP1 images for SUSE OpenStack Cloud come with predefined user `sles`, which you use to log in to the cluster nodes. This user has been configured for password-less 'sudo' and is the one recommended to be used by Terraform and `skuba`.

4.2.1 Deploying the Cluster Nodes

1. Find the Terraform template files for SUSE OpenStack Cloud in `/usr/share/caasp/terraform/openstack` (which was installed as part of the management pattern - `sudo zypper in -t pattern SUSE-CaaS-Management`). Copy this folder to a location of your choice as the files need adjustment.

```
mkdir -p ~/caasp/deployment/  
cp -r /usr/share/caasp/terraform/openstack/ ~/caasp/deployment/  
cd ~/caasp/deployment/openstack/
```

2. Once the files are copied, rename the `terraform.tfvars.example` file to `terraform.tfvars`:

```
mv terraform.tfvars.example terraform.tfvars
```

3. Edit the `terraform.tfvars` file and add/modify the following variables:

```
# Name of the image to use
```

```
image_name = "SLE-15-SP1-JeOS-GMC"

# Identifier to make all your resources unique and avoid clashes with other users of
# this Terraform project
stack_name = "testing" ❶

# Name of the internal network to be created
internal_net = "testing-net" ❷

# Name of the internal subnet to be created
# IMPORTANT: If this variable is not set or empty,
# then it will be generated following a schema like
# internal_subnet = "${var.internal_net}-subnet"
internal_subnet = "testing-subnet"

# Name of the internal router to be created
# IMPORTANT: If this variable is not set or empty,
# then it will be generated following a schema like
# internal_router = "${var.internal_net}-router"
internal_router = "testing-router"

# Name of the external network to be used, the one used to allocate floating IPs
external_net = "floating"

# CIDR of the subnet for the internal network
subnet_cidr = "172.28.0.0/24"

# Number of master nodes
masters = 3 ❸

# Number of worker nodes
workers = 2 ❹

# Size of the master nodes
master_size = "m1.medium"

# Size of the worker nodes
worker_size = "m1.medium"

# Attach persistent volumes to workers
workers_vol_enabled = 0

# Size of the worker volumes in GB
workers_vol_size = 5

# Name of DNS domain
dnsdomain = "testing.example.com"
```

```

# Set DNS Entry (0 is false, 1 is true)
dnsentry = 0

# Optional: Define the repositories to use
# repositories = {
#   repository1 = "http://repo.example.com/repository1/"
#   repository2 = "http://repo.example.com/repository2/"
# }
repositories = {} ⑤

# Define required packages to be installed/removed. Do not change those.
packages = [ ⑥
  "kernel-default",
  "-kernel-default-base",
  "new-package-to-install"
]

# ssh keys to inject into all the nodes
authorized_keys = [ ⑦
  ""
]

# IMPORTANT: Replace these ntp servers with ones from your infrastructure
ntp_servers = ["0.novell.pool.ntp.org", "1.novell.pool.ntp.org",
  "2.novell.pool.ntp.org", "3.novell.pool.ntp.org"] ⑧

```

- ① stack_name: Prefix for all machines of the cluster spawned by terraform.
- ② internal_net: the internal network name that will be created/used for the cluster in SUSE OpenStack Cloud. **Note:** This string will be used to generate the human readable IDs in SUSE OpenStack Cloud. If you use a generic term, deployment is very likely to fail because the term is already in use by someone else. It's a good idea to use your username or some other unique identifier.
- ③ masters: Number of master nodes to be deployed.
- ④ workers: Number of worker nodes to be deployed.
- ⑤ repositories: A list of additional repositories to be added on each machines. Leave empty if no additional packages need to be installed.
- ⑥ packages: Additional packages to be installed on the node. **Note:** Do not remove any of the pre-filled values in the packages section. This can render your cluster unusable. You can add more packages but do not remove any of the default packages listed.

- 7 authorized_keys: List of ssh public keys that will be injected into the cluster nodes, allowing you to be able to log in into them via SSH as sles user. Copy and paste the text from the *keyname.pub* file here, **not** the private key. At least one of the keys must match a key loaded into your ssh-agent.
- 8 ntp_servers: A list of ntp servers you would like to use with chrony.



Tip

You can set the timezone before deploying the nodes by modifying the following file:

- ~/caasp/deployment/openstack/cloud-init/common.tpl

4. (Optional) If you absolutely need to be able to SSH into your cluster nodes using password instead of key-based authentication, this is the best time to set it globally for all of your nodes. If you do this later, you will have to do it manually. To set this, modify the cloud-init configuration and comment out the related SSH configuration: ~/caasp/deployment/openstack/cloud-init/common.tpl

```
# Workaround for bsc#1138557 . Disable root and password SSH login
# - sed -i -e '/^PermitRootLogin/s/^.*$/PermitRootLogin no/' /etc/ssh/sshd_config
# - sed -i -e '/^#ChallengeResponseAuthentication/s/^.*$/
ChallengeResponseAuthentication no/' /etc/ssh/sshd_config
# - sed -i -e '/^#PasswordAuthentication/s/^.*$/PasswordAuthentication no/' /etc/
ssh/sshd_config
# - systemctl restart sshd
```

5. Register your nodes by using the SUSE CaaSP Product Key or by registering nodes against local SUSE Repository Mirroring Server in ~/caasp/deployment/openstack/registration.auto.tfvars:

Substitute <CAASP_REGISTRATION_CODE> for the code from *Section 4.1.2, "Product Key"*.

```
## To register CaaSP product please use one of the following method
# - register against SUSE Customer Service, with SUSE CaaSP Product Key
# - register against local SUSE Repository Mirroring Server

# SUSE CaaSP Product Key
caasp_registry_code = "<CAASP_REGISTRATION_CODE>"

# SUSE Repository Mirroring Server Name (FQDN)
#rmt_server_name = "rmt.example.com"
```

This is required so all the deployed nodes can automatically register with SUSE Customer Center and retrieve packages.

6. You can also enable Cloud Provider Integration with OpenStack in `~/caasp/deployment/openstack/cpi.auto.tfvars`:

```
# Enable CPI integration with OpenStack
cpi_enable = true

# Used to specify the name of to your custom CA file located in /etc/pki/trust/
anchors/.
# Upload CUSTOM_CA_FILE to this path on nodes before joining them to your cluster.
#ca_file = "/etc/pki/trust/anchors/<CUSTOM_CA_FILE>"
```

7. Now you can deploy the nodes by running:

```
terraform init
terraform plan
terraform apply
```

Check the output for the actions to be taken. Type "yes" and confirm with Enter when ready. Terraform will now provision all the machines and network infrastructure for the cluster.

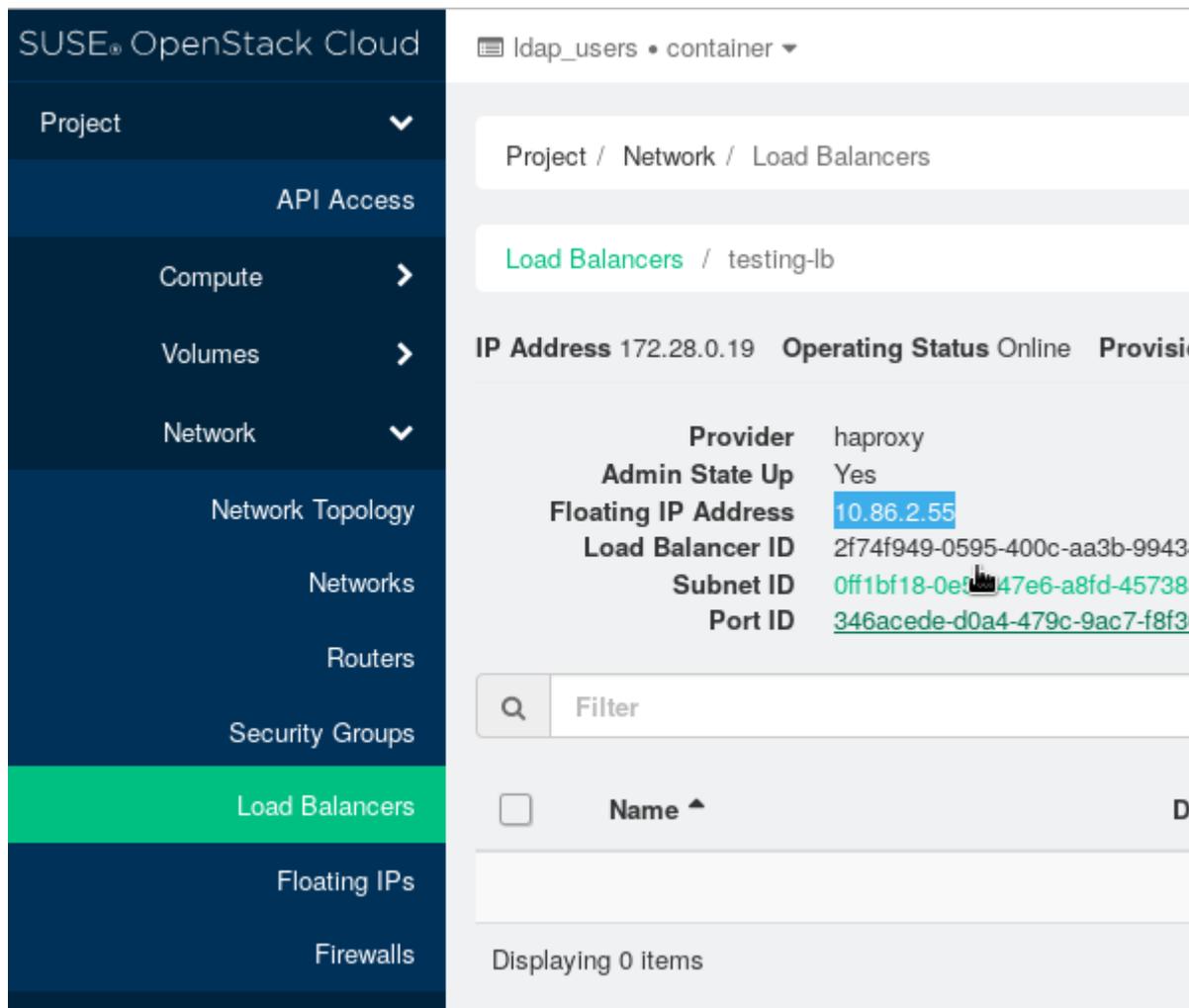


Important: Note down IP/FQDN for nodes

The IP addresses of the generated machines will be displayed in the Terraform output during the cluster node deployment. You need these IP addresses to deploy SUSE CaaS Platform to the cluster.

If you need to find an IP address later on, you can run `terraform output` within the directory you performed the deployment from the `~/caasp/deployment/openstack` directory or perform the following steps:

1. Log in to SUSE OpenStack Cloud and click on *Network > Load Balancers*. Find the one with the string you entered in the Terraform configuration above, for example "testing-lb".
2. Note down the "Floating IP". If you have configured an FQDN for this IP, use the host name instead.



3. Now click on *Compute > Instances*.
4. Switch the filter dropdown box to Instance Name and enter the string you specified for stack_name in the terraform.tfvars file.
5. Find the floating IPs on each of the nodes of your cluster.

4.2.2 Logging in to the Cluster Nodes

1. Connecting to the cluster nodes can be accomplished only via SSH key-based authentication thanks to the ssh-public key injection done earlier via Terraform. You can use the predefined sles user to log in.

If the ssh-agent is running in the background, run:

```
ssh sles@<NODE_IP_ADDRESS>
```

Without the ssh-agent running, run:

```
ssh sles@<NODE_IP_ADDRESS> -i <PATH_TO_YOUR_SSH_PRIVATE_KEY>
```

2. Once connected, you can execute commands using password-less `sudo`. In addition to that, you can also set a password if you prefer to.

To set the **root password**, run:

```
sudo passwd
```

To set the **sles user's password**, run:

```
sudo passwd sles
```



Important: Password authentication has been disabled

Under the default settings you always need your SSH key to access the machines. Even after setting a password for either `root` or `sles` user, you will be unable to log in via SSH using their respective passwords. You will most likely receive a `Permission denied (publickey)` error. This mechanism has been deliberately disabled because of security best practices. However, if this setup does not fit your workflows, you can change it at your own risk by modifying the SSH configuration: under `/etc/ssh/sshd_config`

To allow password SSH authentication, set:

```
+ PasswordAuthentication yes
```

To allow login as root via SSH, set:

```
+ PermitRootLogin yes
```

For the changes to take effect, you need to restart the SSH service by running:

```
sudo systemctl restart sshd.service
```

4.2.3 Container Runtime Proxy

Important

CRI-O proxy settings must be adjusted manually on all nodes before joining the cluster!

In some environments you must configure the container runtime to access the container registries through a proxy. In this case, please refer to: [SUSE CaaS Platform Admin Guide: Configuring HTTP/HTTPS Proxy for CRI-O \(https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_httphttps_proxy_for_cri_o\)](https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_httphttps_proxy_for_cri_o)

4.3 Deployment on VMware

Note: Preparation Required

You must have completed [Section 4.1, “Deployment Preparations”](#) to proceed.

4.3.1 Environment Description

Note

These instructions are based on [VMware ESXi 6.7.0.20000](#).

Important

These instructions currently do not describe how to set up a load balancer.

This will be added in future versions. You must provide your own load balancing solution that directs access to the master nodes.

Important

VMware vSphere doesn't offer a load-balancer solution. Please expose port [6443](#) for the Kubernetes api-servers on the master nodes on a local load balancer using round-robin 1:1 port forwarding.

4.3.2 Setup with AutoYaST

For each VM deployment, follow the AutoYaST installation method used for deployment on bare metal machines as described in *Section 4.4, "Deployment on Bare Metal"*.

4.3.3 Setup using VMWare Template

4.3.3.1 VM Preparation for Creating a Template

1. Upload the ISO image SLE-15-SP1-Installer-DVD-x86_64-GM-DVD1.iso to the desired VMware datastore.

Now you can create a new base VM for SUSE CaaS Platform within the designated resource pool through the vSphere WebUI:

1. Create a "New Virtual Machine".
2. Define a name for the virtual machine (VM).

New Virtual Machine

✓ 1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Select a name and folder

Specify a unique name and target location

Virtual machine name:

Select a location for the virtual machine.

File browser showing the location selection process:

- ▼ jazz.qa.prvsuse.net
 - > PROVO
 - > PROVO-DO-NOT-USE

CANCEL

BACK

NEXT

3. Select the folder where the VM will be stored.
4. Select a Compute Resource that will run the VM.

New Virtual Machine

- ✓ 1 Select a creation type
- ✓ 2 Select a name and folder
- 3 Select a compute resource**
- 4 Select storage
- 5 Select compatibility
- 6 Select a guest OS
- 7 Customize hardware
- 8 Ready to complete

Select a compute resource

Select the destination compute resource for this operation

- ▼ PROVO
 - ▼ Cluster-JAZZ
 - 📄 jazz14.qa.prv.suse.net
 - 📄 jazz15.qa.prv.suse.net
 - 📄 jazz16.qa.prv.suse.net
 - 📄 jazz19.qa.prv.suse.net
 - 📄 jazz20.qa.prv.suse.net
 - 📄 jazz22.qa.prv.suse.net
 - > CaaSP_RP
 - > VCS
 - > VMGuests

Compatibility

✓ Compatibility checks succeeded.

CANCEL

BACK

NEXT

5. Select the storage to be used by the VM.

New Virtual Machine

- ✓ 1 Select a creation type
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- 4 Select storage**
- 5 Select compatibility
- 6 Select a guest OS
- 7 Customize hardware
- 8 Ready to complete

Select storage

Select the storage for the configuration and disk files

VM Storage Policy: Datastore Default ▾

Name	Capacity	Provisioned	Free	Type
 3PAR	5.37 TB	1.16 TB	4.73 TB	VM
 jazz14	178.75 GB	973 MB	177.8 GB	VM
 jazz15	365 GB	974 MB	364.05 GB	VM
 jazz16	551.25 GB	975 MB	550.3 GB	VM
 jazz19	499.75 GB	1.29 GB	498.46 GB	VM
 jazz20	499.75 GB	3.49 GB	496.26 GB	VM
 jazz22	499.75 GB	41.29 GB	458.46 GB	VM
 VMWARE	243.98 GB	62.65 GB	181.33 GB	NF

Compatibility

✓ Compatibility checks succeeded.

CANCEL

BACK

NEXT

6. Select ESXi 6.7 and later from compatibility.

New Virtual Machine

- ✓ 1 Select a creation type
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- ✓ 4 Select storage
- ✓ 5 Select compatibility
- 6 Select a guest OS
- 7 Customize hardware
- 8 Ready to complete

Select compatibility

Select compatibility for this virtual machine depending on the hosts in your environment

The host or cluster supports more than one VMware virtual machine version. Select a compatibility for the virtual machine.

Compatible with: ⓘ

This virtual machine uses hardware version 14, which provides the best performance and latest features available in ESXi 6.7.

CANCEL

BACK

NEXT

7. Select *Guest OS Family* > *Linux* and *Guest OS Version* > *SUSE Linux Enterprise 15 (64-bit)*.

Note: You will manually select the correct installation medium in the next step.

New Virtual Machine

- ✓ 1 Select a creation type
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- ✓ 4 Select storage
- ✓ 5 Select compatibility
- 6 Select a guest OS**
- 7 Customize hardware
- 8 Ready to complete

Select a guest OS

Choose the guest OS that will be installed on the virtual machine

Identifying the guest operating system here allows the wizard to provide the appropriate defaults for the operating system installation.

Guest OS Family:

Guest OS Version:

Compatibility: ESXi 6.7 and later (VM version 14)

CANCEL

BACK

NEXT

8. Now customize the hardware settings.

New Virtual Machine

- ✓ 1 Select a creation type
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- ✓ 4 Select storage
- ✓ 5 Select compatibility
- ✓ 6 Select a guest OS
- 7 Customize hardware**
- 8 Ready to complete

Customize hardware

Configure the virtual machine hardware

Virtual Hardware VM Options

Shares	Normal	1000
Limit - IOPs	Unlimited	
Virtual flash read cache	0	MB
Disk Mode	Dependent	
Virtual Device Node	New SCSI controller	SCSI(0:0) New Hard disk
> New SCSI controller *	VMware Paravirtual	
▼ New Network *	VM Network	
Status	<input checked="" type="checkbox"/> Connect At Power On	
Adapter Type	VMXNET 3	
DirectPath I/O	<input type="checkbox"/> Enable	
MAC Address	Automatic	
▼ New CD/DVD Drive *	Datastore ISO File	
Status	<input checked="" type="checkbox"/> Connect At Power On	
CD/DVD Media	[3PAR] caasp-team/SLE-15-SF	BROWSE...
Device Mode	Passthrough CD-ROM	
Virtual Device Node	New SATA Controller	SATA(0:0) New CD/DVD Drive

Compatibility: ESXi 6.7 and later

CANCEL

- a. Select *CPU* > 2.
- b. Select *Memory* > 4096 MB.
- c. Select *New Hard disk* > 40 GB, *New Hard disk* > *Disk Provisioning* > *Thin Provision*.
- d. Select *New SCSI Controller* > *LSI Logic Parallel SCSI controller (default)* and change it to "VMware Paravirtualized".
- e. Select *New Network* > *VM Network*, *New Network* > *Adapter Type* > *VMXNET3*.
("VM Network" sets up a bridged network which provides a public IP address reachable within a company.)
- f. Select *New CD/DVD* > *Datastore ISO File*.
- g. Check the box *New CD/DVD* > *Connect At Power On* to be able boot from ISO/DVD.
- h. Then click on "Browse" next to the CD/DVD Media field to select the downloaded ISO image on the desired datastore.
- i. Go to the VM Options tab.

New Virtual Machine

- ✓ 1 Select a creation type
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- ✓ 4 Select storage
- ✓ 5 Select compatibility
- ✓ 6 Select a guest OS
- 7 Customize hardware**
- 8 Ready to complete

Customize hardware

Configure the virtual machine hardware

Virtual Hardware **VM Options**

> General Options	VM Name: fb-sles15-sp1-caasp
> VMware Remote Console Options	<input type="checkbox"/> Lock the guest operating system when the last remote user disconnects
> Encryption	Expand for encryption settings
> Power management	Expand for power management settings
> VMware Tools	Expand for VMware Tools settings
▼ Boot Options	
Firmware	BIOS ▼
	<div style="border: 1px solid orange; padding: 5px;"> Changing firmware might cause the installed guest operating system to become unbootable. ✕</div>
Boot Delay	When powering on or resetting, delay boot order by <input type="text" value="0"/> ▲ ▼ milliseconds
Force EFI boot	<input type="checkbox"/>

CANCEL

BACK

- j. Select *Boot Options*.
- k. Select *Firmware* > *BIOS*.
- l. Confirm the process with *Next*.

4.3.3.2 SUSE Linux Enterprise Server Installation

Power on the newly created VM and install the system over graphical remote console:

1. Enter registration code for SUSE Linux Enterprise in YaST.
2. Confirm the update repositories prompt with "Yes".
3. Remove the check mark in the "Hide Development Versions" box.

4. Make sure the following modules are selected on the "Extension and Module Selection" screen:

- SUSE CaaS Platform 4.0 x86_64 (BETA)
 - Basesystem Module
 - Containers Module (this will automatically be checked when you select SUSE CaaS Platform)
 - Public Cloud Module
5. Enter the registration code to unlock the SUSE CaaS Platform extension.
 6. Select *System Role* › *Minimal* on the "System Role" screen.
 7. Click on "Expert Partitioner" to redesign the default partition layout.
 8. Select "Start with current proposal".

Expert Partitioner

System View

vm154062

- Hard Disks
 - sda
 - sda1
 - sda2
 - sda3
 - sda4
 - RAID
 - Volume Management
 - Bcache
 - NFS
 - Btrfs
 - Device Graphs
 - Installation Summary
 - Settings

Available Storage on vm154062

Device	Size	F	Enc	Type	FS Type	Label	Mount
/dev/sda	40.00 GiB			VMware-VMware Virtual disk			
/dev/sda1	8.00 MiB			BIOS Boot			
/dev/sda2	23.69 GiB	F		Linux Native	Btrfs	/	
/dev/sda3	12.46 GiB	F		Linux Native	XFS	/home	
/dev/sda4	3.84 GiB	F		Linux Swap	Swap	swap	

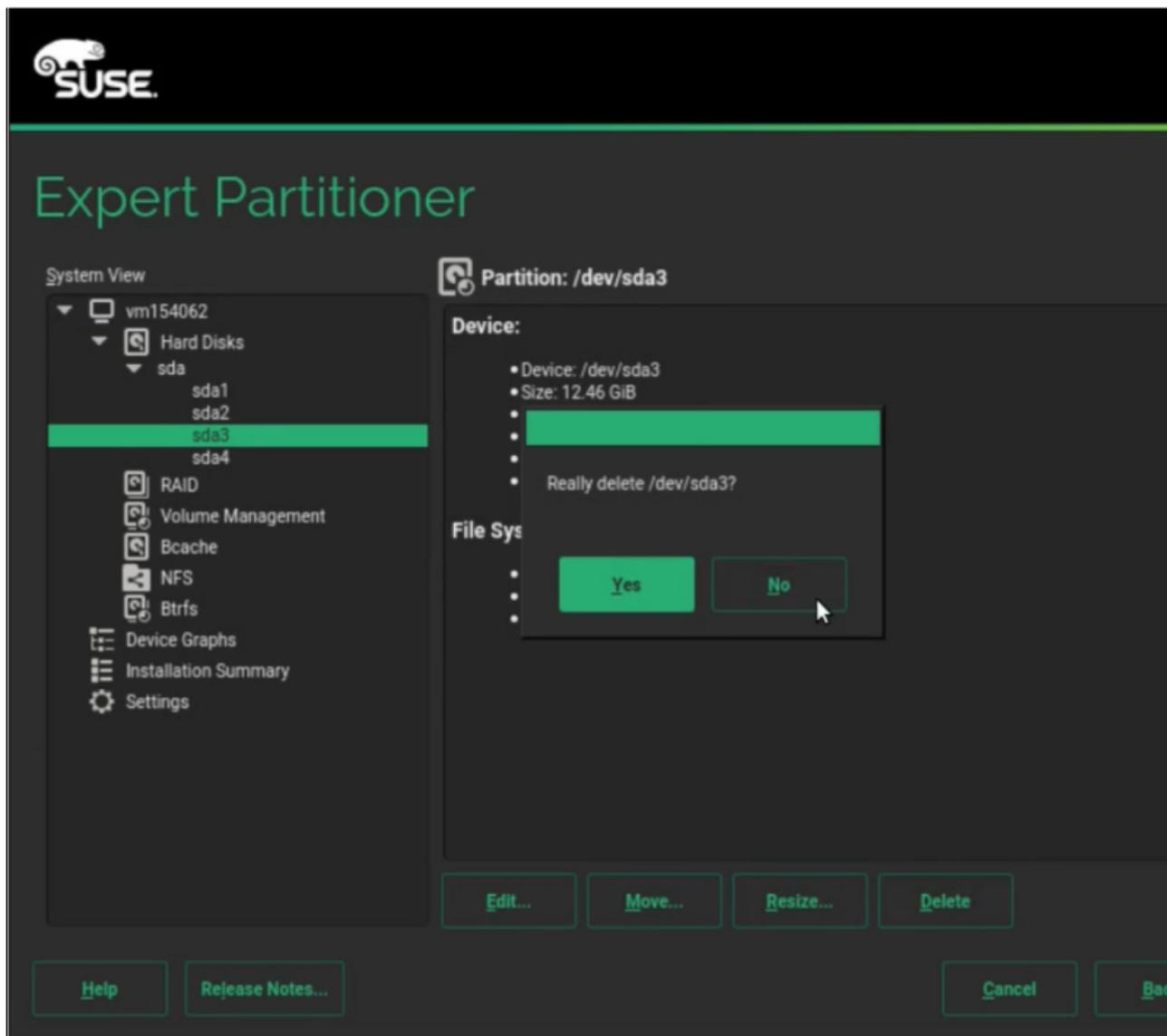
Rescan Devices Import Mount Points...

Help Release Notes... Cancel Back

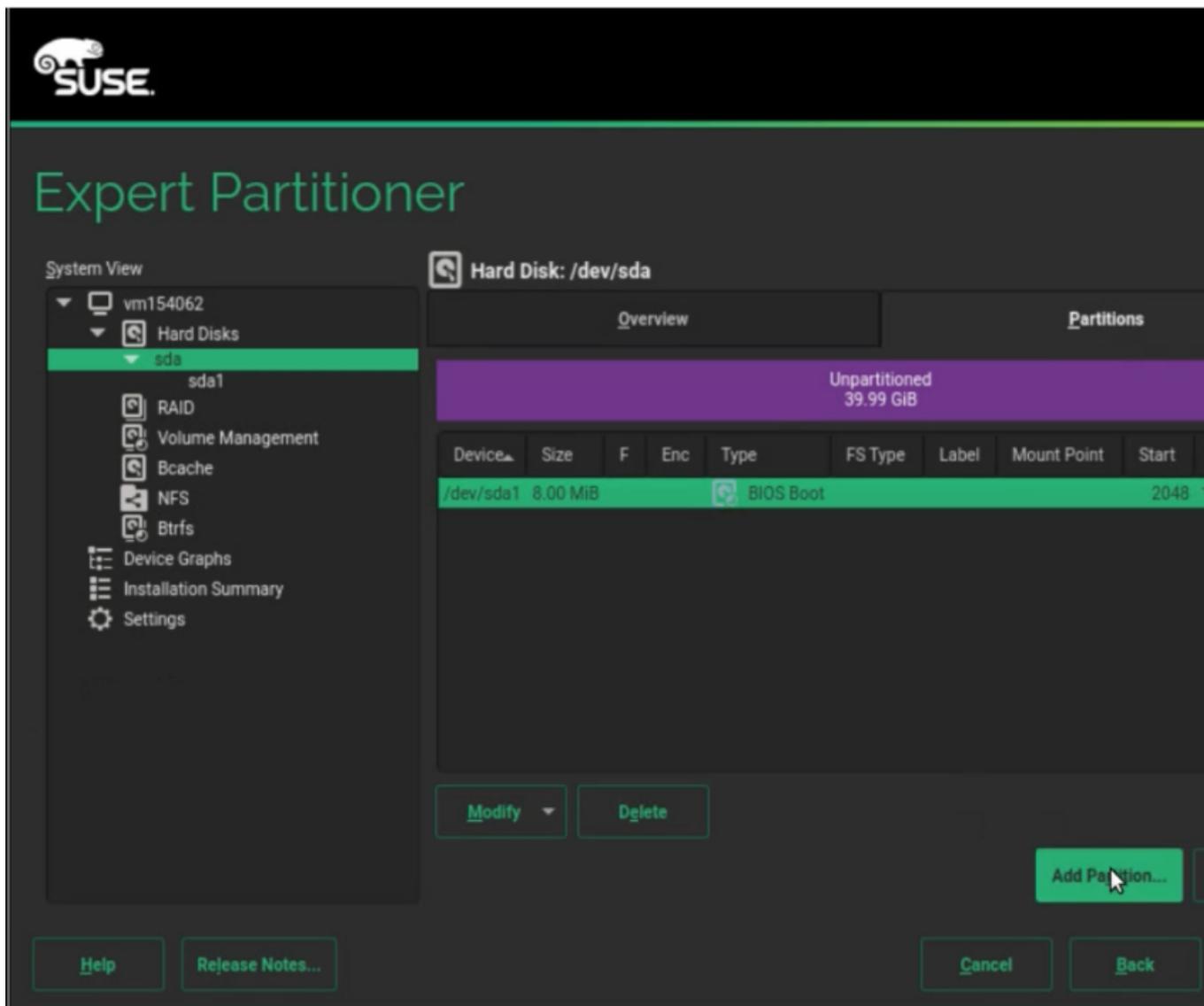
a. Keep sda1 as BIOS partition.

b. Remove the root / partition.

Select the device in "System View" on the left (default: /dev/sda2) and click "Delete".
Confirm with "Yes".



- c. Remove the /home partition.
 - d. Remove the swap partition.
9. Select the /dev/sda/ device in "System View" and then click *Partitions* > *Add Partition*.



10. Accept the default maximum size (remaining size of the hard disk defined earlier without the boot partition).

Add Partition on /dev/sda

New Partition Size

Maximum Size (39.99 GiB)

Custom Size

Size

39.99 GiB

Custom Region

Start Block

18432

End Block

83886046

Help

Release Notes...

Cancel

Back

- a. Confirm with "Next".
- b. Select *Role > Operating System*



Add Partition on /dev/sda

Role

- Operating System
- Data and ISV Applications
- Swap
- EFI Boot Partition
- Raw Volume (unformatted)

Help

Release Notes...

Cancel

Back

- c. Confirm with "Next".
- d. Accept the default settings.

Add Partition on /dev/sda

Formatting Options

Format device

Filesystem

Btrfs

Enable Snapshots

Do not format device

Partition ID:

Linux Native

Encrypt Device

Mounting Options

Mount device

Mount Point

/

Fstab Options...

Do not mount device

Subvolume Handling

Help Release Notes... Cancel Back

- Filesystem: Btrfs
- Enable Snapshots
- Mount Device
- Mount Point /

11. You should be left with two partitions. Now click "Accept".

Expert Partitioner

System View

- vm154062
 - Hard Disks
 - sda**
 - sda1
 - sda2
 - RAID
 - Volume Management
 - Bcache
 - NFS
 - Btrfs
 - Device Graphs
 - Installation Summary
 - Settings

Hard Disk: /dev/sda

Overview | Partitions

sda2 39.99 GiB

Device	Size	F	Enc	Type	FS Type	Label	Mount Point	Start
/dev/sda1	8.00 MiB			BIOS Boot				2048
/dev/sda2	39.99 GiB	F		Linux Native	Btrfs		/	18432

Modify | Delete | Add Partition... | Help | Release Notes... | Cancel | Back

12. Confirm the partitioning changes.

Suggested Partitioning

Layout configured manually using the Expert Partitioner.

Changes to partitioning:

- Create GPT on `/dev/sda`
- Create partition `/dev/sda1` (8.00 MiB) as BIOS Boot Partition
- Create partition `/dev/sda2` (39.99 GiB) for `/` with `btrfs`
- 10 subvolume actions ([see details](#))

Guided Setup

Expert Partitioner ▾

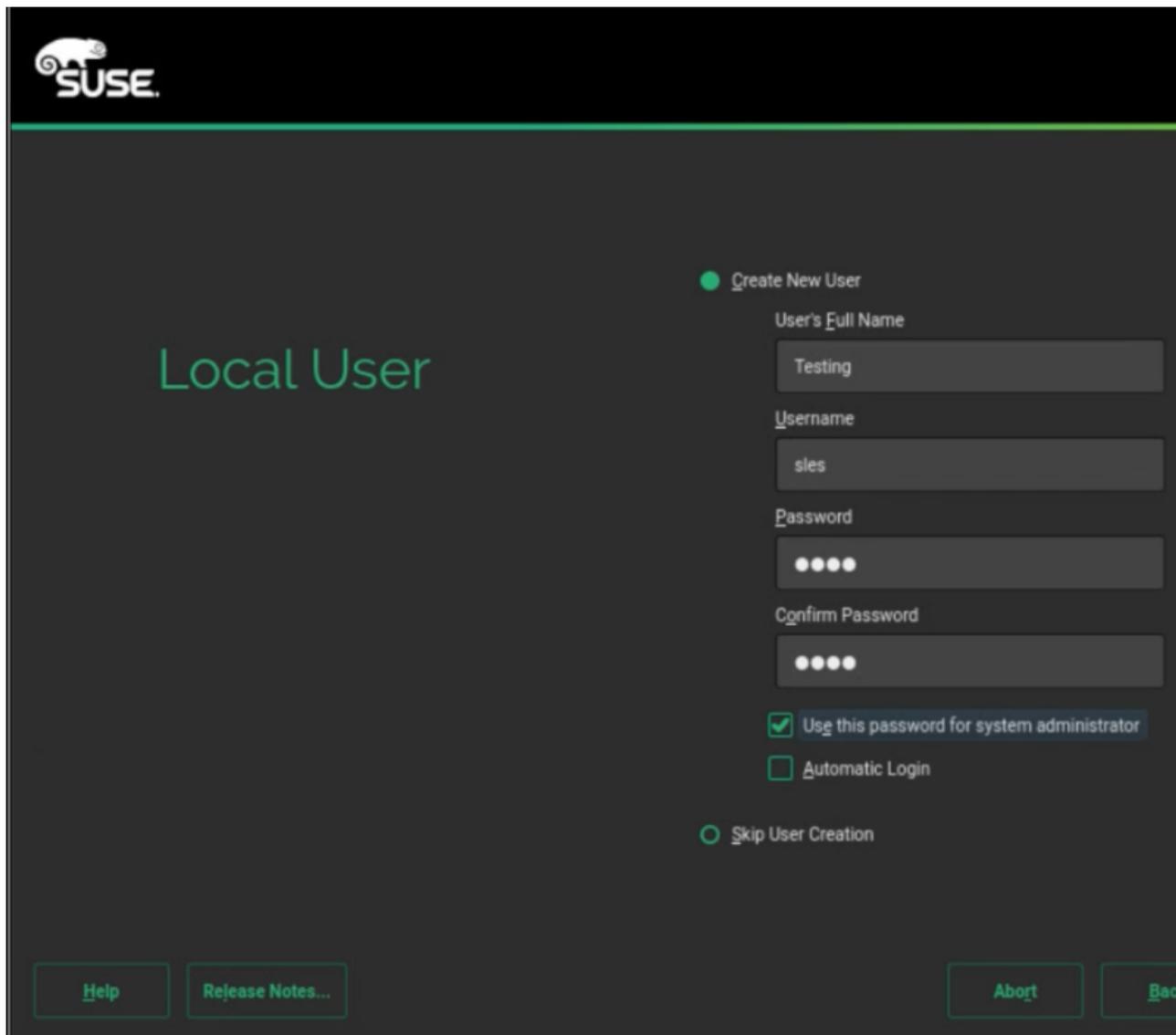
Help

Release Notes...

Abort

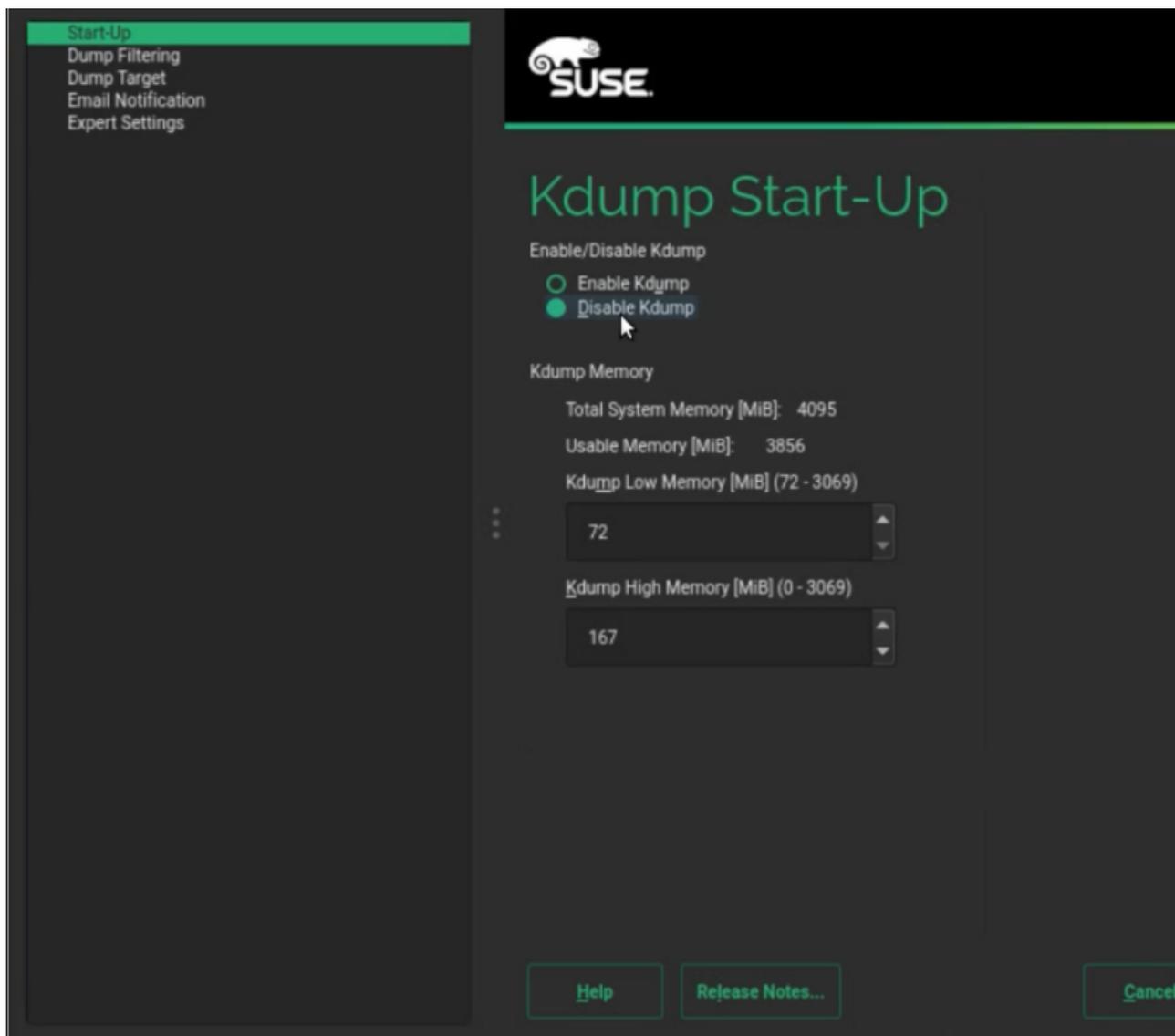
Back

13. Click "Next".
14. Configure your timezone and click "Next".
15. Create a user with the username `sles` and specify a password.
 - a. Check the box *Local User > Use this password for system administrator*.

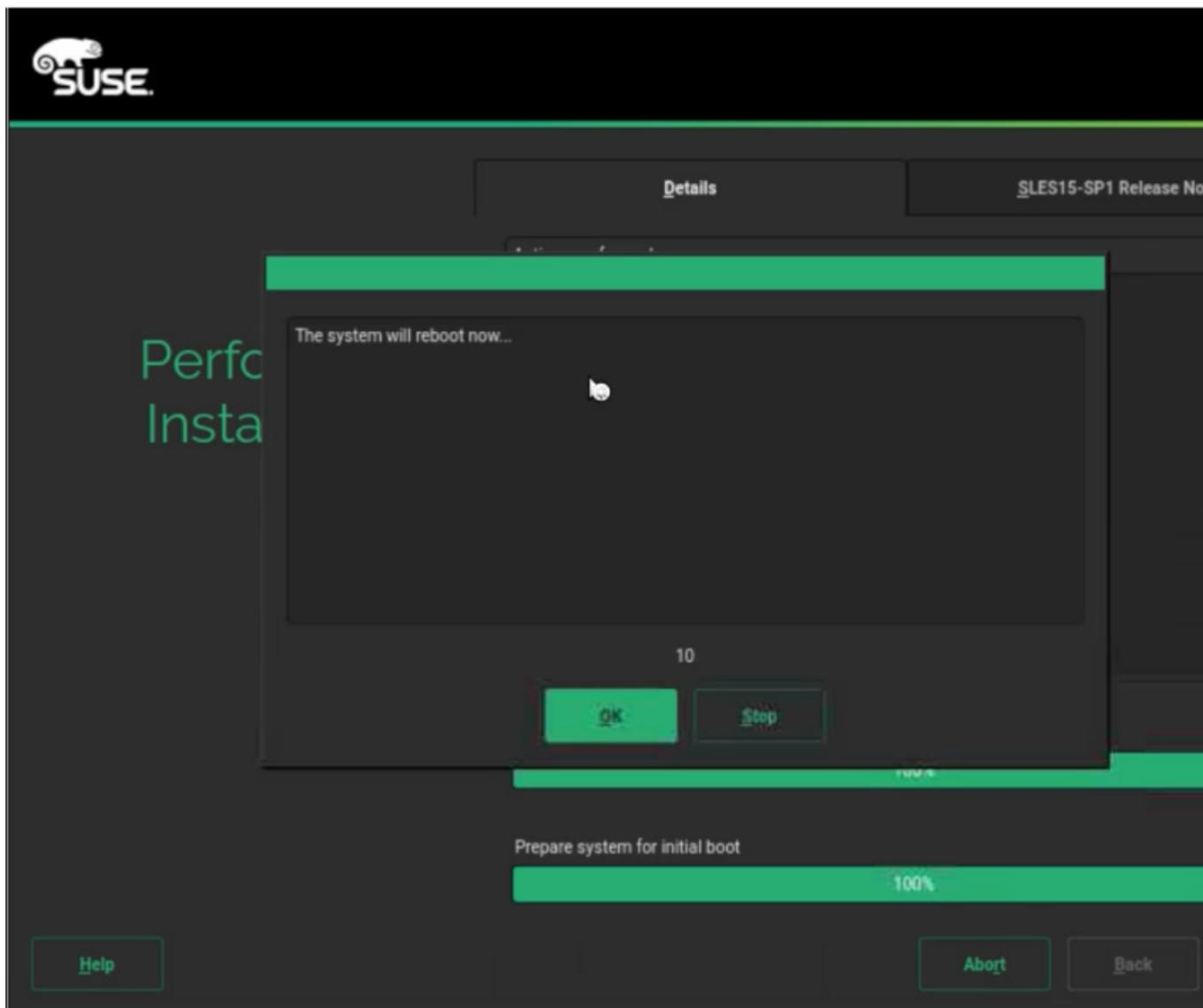


16. Click "Next".
17. On the "Installation Settings" screen:
 - a. In the "Security" section:
 - i. Disable the Firewall (click on (disable)).
 - ii. Enable the SSH service (click on (enable)).
 - b. Scroll to the kdump section of the software description and click on the title.
18. In the "Kdump Start-Up" screen, select *Enable/Disable Kdump > Disable Kdump*.

- a. Confirm with "OK".



19. Click "Install". Confirm the installation by clicking "Install" in the pop-up dialog.
20. Finish the installation and confirm system reboot with "OK".



4.3.3.3 Preparation of the VM as a Template

In order to run SUSE CaaS Platform on the created VMs, you must configure and install some additional packages like `sudo`, `cloud-init` and `open-vm-tools`.



Tip: Activate extensions during SUSE Linux Enterprise installation with YaST

Steps 1-4 may be skipped, if they were already performed in YaST during the SUSE Linux Enterprise installation.

1. Register the SLES15-SP1 system. Substitute `<CAASP_REGISTRATION_CODE>` for the code from *Section 4.1.2, "Product Key"*.

```
SUSEConnect -r CAASP_REGISTRATION_CODE
```

2. Register the Containers module (free of charge):

```
SUSEConnect -p sle-module-containers/15.1/x86_64
```

3. Register the Public Cloud module for basic cloud-init package (free of charge):

```
SUSEConnect -p sle-module-public-cloud/15.1/x86_64
```

4. Register the SUSE CaaS Platform module. Substitute `<CAASP_REGISTRATION_CODE>` for the code from *Section 4.1.2, "Product Key"*.

```
SUSEConnect -p caasp/4.0/x86_64 -r CAASP_REGISTRATION_CODE
```

5. Install required packages. As root, run:

```
zypper in sudo cloud-init cloud-init-vmware-guestinfo open-vm-tools
```

6. Enable the installed cloud-init services. As root, run:

```
systemctl enable cloud-init cloud-init-local cloud-config cloud-final
```

7. Deregister from scc:

```
SUSEConnect -d; SUSEConnect --cleanup
```

8. Do a cleanup of the SLE image for converting into a VMware template. As root, run:

```
rm /etc/machine-id /var/lib/zypp/AnonymousUniqueId \  
/var/lib/systemd/random-seed /var/lib/dbus/machine-id \  
/var/lib/wicked/*
```

9. Clean up btrfs snapshots and create one with initial state:

```
snapper list  
snapper delete <list_of_nums_of_unneeded_snapshots>
```

```
snapper create -d "Initial snapshot for caasp template" -t single
```

10. Power down the VM. As root, run:

```
shutdown -h now
```

4.3.3.4 Creating the VMware Template

Now you can convert the VM into a template in VMware (or repeat this action block for each VM).

1. In the vSphere WebUI, right-click on the VM and select *Template > Convert to Template*. Name it reasonably so you can later identify the template. The template will be created.

4.3.4 Deploying VMs from the Template

4.3.4.1 Using Terraform

1. Find the Terraform template files for VMware in `/usr/share/caasp/terraform/vmware` which was installed as part of the management pattern (`sudo zypper in patterns-caasp-Management`). Copy this folder to a location of your choice; as the files need to be adjusted.

```
mkdir -p ~/caasp/deployment/  
cp -r /usr/share/caasp/terraform/vmware/ ~/caasp/deployment/  
cd ~/caasp/deployment/vmware/
```

2. Once the files are copied, rename the `terraform.tfvars.example` file to `terraform.tfvars`:

```
mv terraform.tfvars.example terraform.tfvars
```

3. Edit the `terraform.tfvars` file and add/modify the following variables:

```
# datastore to use in vSphere  
vsphere_datastore = "STORAGE-0" ①  
  
# datacenter to use in vSphere  
vsphere_datacenter = "DATACENTER" ②
```

```

# network to use in vSphere
vsphere_network = "VM Network" ③

# resource pool the machines will be running in
vsphere_resource_pool = "My_RP" ④

# template name the machines will be copied from
template_name = "sles15-sp1-caasp" ⑤

# IMPORTANT: Replace by "efi" string in case your template was created by using EFI
firmware
firmware = "bios"

# prefix that all of the booted machines will use
# IMPORTANT: please enter unique identifier below as value of
# stack_name variable to not interfere with other deployments
stack_name = "caasp-v4" ⑥

# Number of master nodes
masters = 1 ⑦

# Optional: Size of the root disk in GB on master node
master_disk_size = 50 ⑧

# Number of worker nodes
workers = 2 ⑨

# Optional: Size of the root disk in GB on worker node
worker_disk_size = 40 ⑩

# Username for the cluster nodes. Must exist on base OS.
username = "sles" ⑪

# Optional: Define the repositories to use
# repositories = {
#   repository1 = "http://repo.example.com/repository1/"
#   repository2 = "http://repo.example.com/repository2/"
# }
repositories = {} ⑫

# Minimum required packages. Do not remove them.
# Feel free to add more packages
packages = [ ⑬
]

# ssh keys to inject into all the nodes
authorized_keys = [ ⑭

```

```
"ssh-rsa <example_key> example@example.com"
]

# IMPORTANT: Replace these ntp servers with ones from your infrastructure
ntp_servers = ["0.novell.pool.ntp.org", "1.novell.pool.ntp.org",
"2.novell.pool.ntp.org", "3.novell.pool.ntp.org"] 15
```

- 1 vsphere_datastore: The datastore to use.
- 2 vsphere_datacenter: The datacenter to use.
- 3 vsphere_network: The network to use.
- 4 vsphere_resource_pool: The resource pool to use.
- 5 template_name: The name of the template created according to instructions.
- 6 stack_name: Prefix for all machines of the cluster spawned by terraform. **Note:** This string will be used to generate the human readable IDs in SUSE OpenStack Cloud. If you use a generic term, deployment very likely to fail because the term is already in use by someone else. It's a good idea to use your username or some other unique identifier.
- 7 masters: Number of master nodes to be deployed.
- 8 master_disk_size: Size of the root disk in GB. **Note:** The value must be at least the same size as the source template. It is only possible to increase the size of a disk.
- 9 workers: Number of worker nodes to be deployed.
- 10 worker_disk_size: Size of the root disk in GB. **Note:** The value must be at least the same size as the source template. It is only possible to increase the size of a disk.
- 11 username: Login username for the nodes. **Note:** Leave this as the default `sles`. The username must exist on the used base operating system. It will not be created.
- 12 repositories: A list of additional repositories to be added on each machines. Leave empty if no additional packages need to be installed.
- 13 packages: Additional packages to be installed on the node. **Note:** Do not remove any of the pre-filled values in the `packages` section. This can render your cluster unusable. You can add more packages but do not remove any of the default packages listed.
- 14 authorized_keys: List of ssh-public-keys that will be able to log in to the deployed machines.

15 ntp_servers: A list of ntp servers you would like to use with chrony.

4. Enter the registration code for your nodes in ~/caasp/deployment/vmware/registration.auto.tfvars:

Substitute <CAASP_REGISTRATION_CODE> for the code from *Section 4.1.2, "Product Key"*.

```
# SUSE CaaSP Product Product Key
caasp_registry_code = "CAASP_REGISTRATION_CODE"
```

This is required so all the deployed nodes can automatically register with SUSE Customer Center and retrieve packages.

Once the files are adjusted, terraform needs to know about the vSphere server and the login details for it; these can be exported as environment variables or entered every time terraform is invoked.

Additionally, the ssh-key that is specified in the tfvars file must be added to the keyring, so the machine running skuba can ssh into the machines:

```
export VSPHERE_SERVER="<server_address>"
export VSPHERE_USER="<username>"
export VSPHERE_PASSWORD="<password>"
export VSPHERE_ALLOW_UNVERIFIED_SSL=true # In case you are using custom certificate for
accessing vsphere API

ssh-add <path_to_private_ssh_key_from_tfvars>
```

Run Terraform to create the required machines for use with skuba:

```
terraform init
terraform plan
terraform apply
```

4.3.4.2 Setup by Hand



Note

Full instructions for the manual setup and configuration are currently not in scope of this document.

Deploy the template to your created VMs. After that, boot into the node and configure the OS as needed.

1. Power on the newly created VMs
2. Generate new machine IDs on each node
3. You need to know the FQDN/IP for each of the created VMs during the bootstrap process
4. Continue with bootstrapping/joining of nodes

Important: Regenerating Machine ID

In case you are not using Terraform or AutoYaST you must regenerate machine IDs manually.

During the template preparation you will have removed the machine ID from the template image. This ID is required for proper functionality in the cluster and must be (re-)generated on each machine.

Log in to each virtual machine created from the template and run:

```
dbus-uuidgen --ensure
systemd-machine-id-setup
systemctl restart systemd-journald
```

This will regenerate the `machine_id` values for `DBUS (/var/lib/dbus/machine-id)` and `systemd (/etc/machine-id)` and restart the logging service to make use of the new IDs.

4.3.5 Container Runtime Proxy

Important

CRI-O proxy settings must be adjusted manually on all nodes before joining the cluster!

In some environments you must configure the container runtime to access the container registries through a proxy. In this case, please refer to: [SUSE CaaS Platform Admin Guide: Configuring HTTP/HTTPS Proxy for CRI-O](https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_httphttps_proxy_for_cri_o) (https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_httphttps_proxy_for_cri_o)

4.4 Deployment on Bare Metal



Note: Preparation Required

You must have completed [Section 4.1, “Deployment Preparations”](#) to proceed.

4.4.1 Environment Description



Important

These instructions currently do not describe how to set up a load balancer. This will be added in future versions. You must provide your own load balancing solution that directs access to the master nodes.



Note

The AutoYaST file found in `skuba` is a template. It has the base requirements. This AutoYaST file should act as a guide and should be updated with your company’s standards.



Note

To account for hardware/platform-specific setup criteria (legacy BIOS vs. (U)EFI, drive partitioning, networking, etc.), you must adjust the AutoYaST file to your needs according to the requirements.

Refer to the official AutoYaST documentation for more information: [AutoYaST Guide \(https://documentation.suse.com/sles/15-SP1/single-html/SLES-autoyast/\)](https://documentation.suse.com/sles/15-SP1/single-html/SLES-autoyast/).

4.4.1.1 Hardware Prerequisites

Deployment with AutoYaST will require a minimum **disk size of 40 GB**. 10 GB out of that total space will be reserved for container images without any workloads, for the root partition (30 GB) and the EFI system partition (200 MB).

4.4.2 AutoYaST Preparation

1. On the management machine, get an example AutoYaST file from `/usr/share/caasp/autoyast/bare-metal/autoyast.xml`, (which was installed earlier on as part of the management pattern (`sudo zypper in -t pattern SUSE-CaaS-Management`)).
2. Copy the file to a suitable location to modify it. Name the file `autoyast.xml`.
3. Modify the following places in the AutoYaST file (and any additional places as required by your specific configuration/environment):

a. `<ntp-client>`

Change the pre-filled value to your organization's NTP server. Provide multiple servers if possible by adding new `<ntp_server>` subentries.

b. `<username>sles</username>`

Insert your authorized key in the placeholder field.

c. `<users>`

You can add additional users by creating new blocks in the configuration containing their data.



Note

If the users are configured to not have a password like in the example, ensure the system's `sudoers` file is updated. Without updating the `sudoers` file the user will only be able to perform basic operations that will prohibit many administrative tasks.

The default AutoYaST file provides examples for a disabled `root` user and a `sles` user with authorized key SSH access.

The password for root can be enabled by using the `passwd` command.

d. `<suse_register>`

Insert the email address and SUSE CaaS Platform registration code in the placeholder fields. This activates SUSE Linux Enterprise 15 SP1.

e. `<addon>`

Insert the SUSE CaaS Platform registration code in the placeholder field. This enables the SUSE CaaS Platform extension module. Update the AutoYaST file with your registration keys and your company's best practices and hardware configurations.



Note

Your SUSE CaaS Platform registration key can be used to both activate SUSE Linux Enterprise 15 SP1 and enable the extension.

Refer to the official AutoYaST documentation for more information: [AutoYaST Guide \(https://documentation.suse.com/sles/15-SP1/single-html/SLES-autoyast/\)](https://documentation.suse.com/sles/15-SP1/single-html/SLES-autoyast/).

4. Host the AutoYaST files on a Web server reachable inside the network you are installing the cluster in.

4.4.2.1 Deploying with local Repository Mirroring Tool (RMT) server

In order to use a local Repository Mirroring Tool (RMT) server for deployment of packages, you need to specify the server configuration in your AutoYaST file. To do so add the following section:

```
<suse_register>
<do_registration config:type="boolean">true</do_registration>
<install_updates config:type="boolean">true</install_updates>

<reg_server>https://rmt.example.org</reg_server> ①
<reg_server_cert>https://rmt.example.org/rmt.crt</reg_server_cert> ②
<reg_server_cert_fingerprint_type>SHA1</reg_server_cert_fingerprint_type>
<reg_server_cert_fingerprint>0C:A4:A1:06:AD:E2:A2:AA:D0:08:28:95:05:91:4C:07:AD:13:78:FE</
reg_server_cert_fingerprint> ③
<slp_discovery config:type="boolean">>false</slp_discovery>
<addons config:type="list">
  <addon>
    <name>sle-module-containers</name>
    <version>15.1</version>
    <arch>x86_64</arch>
  </addon>
  <addon>
    <name>caasp</name>
    <version>4.0</version>
    <arch>x86_64</arch>
  </addon>
</addons>
```

```
</suse_register>
```

- 1 Provide FQDN of the Repository Mirroring Tool (RMT) server
- 2 Provide the location on the server where the certificate can be found
- 3 Provide the certificate fingerprint for the Repository Mirroring Tool (RMT) server

4.4.3 Provisioning the Cluster Nodes

Once the AutoYaST file is available in the network that the machines will be configured in, you can start deploying machines.

The default production scenario consists of 8 nodes:

- 2 load balancers
- 3 masters
- 3 workers

Depending on the type of load balancer you wish to use, you need to deploy at least 6 machines to serve as cluster nodes and provide 2 load balancers from the environment.

The load balancer must point at the machines that are assigned to be used as master nodes in the future cluster.



Tip

If you do not wish to use infrastructure load balancers, please deploy additional machines and refer to *Section 4.1.4, "Load Balancer"*.

Install SUSE Linux Enterprise 15 SP1 from your preferred medium and follow the steps for *Invoking the Auto-Installation Process* (<https://documentation.suse.com/sles/15-SP1/single-html/SLES-autoyast/#invoking-autoinst>)

Provide autoyast=https://[webserver/path/to/autoyast.xml] during the SUSE Linux Enterprise 15 SP1 installation.

4.4.3.1 SUSE Linux Enterprise Server Installation



Note

Use AutoYaST and make sure to use a staged frozen patchlevel via RMT/SUSE Manager to ensure a 100% reproducible setup. [RMT Guide \(https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#cha-rmt-client\)](https://documentation.suse.com/sles/15-SP1/single-html/SLES-rmt/#cha-rmt-client)

Once the machines have been installed using the AutoYaST file, you are now ready proceed with [Chapter 5, Bootstrapping the Cluster](#).

4.4.4 Container Runtime Proxy



Important

CRI-O proxy settings must be adjusted manually on all nodes before joining the cluster!

In some environments you must configure the container runtime to access the container registries through a proxy. In this case, please refer to: [SUSE CaaS Platform Admin Guide: Configuring HTTP/HTTPS Proxy for CRI-O \(https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_httphttps_proxy_for_cri_o\)](https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_httphttps_proxy_for_cri_o)

4.5 Deployment on Existing SLES Installation

If you already have a running SUSE Linux Enterprise 15 SP1 installation, you can add SUSE CaaS Platform to this installation using SUSE Connect. You also need to enable the "Containers" module because it contains some dependencies required by SUSE CaaS Platform.

4.5.1 Requirements



Note: Preparation Required

You must have completed [Section 4.1, "Deployment Preparations"](#) to proceed.

4.5.1.1 Dedicated Cluster Nodes

Important

Adding a machine with an existing use case (e.g. web server) as a cluster node is not supported!

SUSE CaaS Platform requires dedicated machines as cluster nodes.

The instructions in this document are meant to add SUSE CaaS Platform to an existing SUSE Linux Enterprise installation that has no other active use case.

For example: You have installed a machine with SUSE Linux Enterprise but it has not yet been commissioned to run a specific application and you decide now to make it a SUSE CaaS Platform cluster node.

4.5.1.2 Disabling Swap

When using a pre-existing SUSE Linux Enterprise installation, swap will be enabled. You must disable swap for all cluster nodes before performing the cluster bootstrap.

On all nodes that are meant to join the cluster; run:

```
sudo swapoff -a
```

Then modify /etc/fstab on each node to remove the swap entries.

Important

It is recommended to reboot the machine to finalize these changes and prevent accidental reactivation of swap during an automated reboot of the machine later on.

4.5.2 Adding SUSE CaaS Platform repositories

Retrieve your SUSE CaaS Platform registration code and run the following. Substitute <CAASP_REGISTRATION_CODE> for the code from *Section 4.1.2, "Product Key"*.

```
SUSEConnect -p sle-module-containers/15.1/x86_64
```

```
SUSEConnect -p caasp/4.0/x86_64 -r <CAASP_REGISTRATION_CODE>
```

Repeat all preparation steps for any cluster nodes you wish to join. You can then proceed with *Chapter 5, Bootstrapping the Cluster*.

5 Bootstrapping the Cluster

Bootstrapping the cluster is the initial process of starting up the cluster and defining which of the nodes are masters and which are workers. For maximum automation of this process, SUSE CaaS Platform uses the `skuba` package.

5.1 Preparation

5.1.1 Install skuba

First you need to install `skuba` on a management machine, like your local workstation:

1. Add the SLE15 SP1 extension containing `skuba`. This also requires the "containers" module.

```
SUSEConnect -p sle-module-containers/15.1/x86_64
SUSEConnect -p caasp/4.0/x86_64 -r <PRODUCT_KEY>
```

2. Install the management pattern with:

```
zypper in -t pattern SUSE-CaaS-Management
```



Tip

Example deployment configuration files for each deployment scenario are installed under `/usr/share/caasp/terraform/`, or in case of the bare metal deployment: `/usr/share/caasp/autoyast/`.

5.1.2 Container Runtime Proxy



Important

CRI-O proxy settings must be adjusted manually on all nodes before joining the cluster!

In some environments you must configure the container runtime to access the container registries through a proxy. In this case, please refer to: [SUSE CaaS Platform Admin Guide: Configuring HTTP/HTTPS Proxy for CRI-O](https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_httphttps_proxy_for_cri_o) (https://documentation.suse.com/suse-caasp/4/single-html/caasp-admin/#_configuring_httphttps_proxy_for_cri_o)

5.2 Cluster Deployment

Make sure you have added the SSH identity (corresponding to the public SSH key distributed above) to the `ssh-agent` on your workstation. For instructions on how to add the SSH identity, refer to [Section 4.1.1, "Basic SSH Key Configuration"](#).

This is a requirement for `skuba` (<https://github.com/SUSE/skuba#prerequisites>).

By default `skuba` connects to the nodes as `root` user. A different user can be specified by the following flags:

```
--sudo --user <USERNAME>
```

Important

You must configure `sudo` for the user to be able to authenticate without password. Replace `<USERNAME>` with the user you created during installation. As root, run:

```
echo "<USERNAME> ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers
```

5.2.1 Initializing the Cluster

Now you can initialize the cluster on the deployed machines. As `--control-plane` enter the IP/FQDN of your load balancer. If you do not use a load balancer use your first master node.

```
skuba cluster init --control-plane <LB_IP/FQDN> my-cluster
```

`cluster init` generates the folder named `my-cluster` and initializes the directory that will hold the configuration (`kubeconfig`) for the cluster.

Important

The IP/FQDN must be reachable by every node of the cluster and therefore 127.0.0.1/localhost cannot be used.

5.2.1.1 Transitioning from Docker to CRI-O

SUSE CaaS Platform 4.0.3 **default configuration** uses the CRI-O Container Engine in conjunction with Docker Linux capabilities. This means SUSE CaaS Platform 4.0.3 containers run on top of CRI-O with the following additional Linux capabilities: audit_write, setfcap and mknod. This measure ensures a transparent transition and seamless compatibility with workloads running on the previous SUSE CaaS Platform versions and out-of-the-box Docker compatibility.

In case you wish to use **unmodified CRI-O**, use the --strict-capability-defaults option during the initial setup when you run skuba cluster init, which will create the vanilla CRI-O configuration:

```
skuba cluster init --strict-capability-defaults
```

Please be aware that this might result in incompatibility with your previously running workloads, unless you explicitly define the additional Linux capabilities required on top of CRI-O defaults.

Important

After the bootstrap of the Kubernetes cluster there will be no easy way to revert this modification. Please choose wisely.

5.2.2 Configuring Kubernetes Services

Inspect the kubeadm-init.conf file inside your cluster definition and set extra configuration settings supported by kubeadm. The latest supported version is v1beta1. Later, when you later run skuba node bootstrap, kubeadm will read kubeadm-init.conf and will forcefully set certain settings to the ones required by SUSE CaaS Platform.

5.2.2.1 Network Settings

The default network settings inside `kubeadm-init.conf` are viable for production clusters and adjusting them is optional. If you however wish to change the pod and service subnets, it is important that you do so before the bootstrap. The subnet ranges must be planned carefully, because the settings cannot be adjusted after deployment is complete. The default settings are the following:

```
networking:
  podSubnet: 10.244.0.0/16
  serviceSubnet: 10.96.0.0/12
```

The `podSubnet` IP range must be big enough to contain all IP addresses for all PODs planned for the cluster. The subnet also mustn't conflict with services from outside of the cluster - external databases, file services, etc. This also holds for `serviceSubnet` - the IP range must not conflict with external services and needs to be broad enough for all services planned for the cluster.

5.2.3 Cluster Configuration

Before bootstrapping the cluster, it is advisable to perform some additional configuration.

5.2.3.1 Enabling Cloud Provider Integration

Enable cloud provider integration to take advantage of the underlying cloud platforms and automatically manage resources like the Load Balancer, Nodes (Instances), Network Routes and Storage services.

If you want to enable cloud provider integration with different cloud platforms, initialize the cluster with the flag `--cloud-provider <CLOUD_PROVIDER>`. The only currently available option is `openstack`, but more options are planned:

```
skuba cluster init --control-plane <LB_IP/FQDN> --cloud-provider openstack my-cluster
```

Running the above command will create a directory `my-cluster/cloud/openstack` with a `README.md` and an `openstack.conf.template` in it. Copy `openstack.conf.template` or create an `openstack.conf` file inside `my-cluster/cloud/openstack`, according to the supported format. The supported format and content can be found in the official Kubernetes documentation:

<https://kubernetes.io/docs/concepts/cluster-administration/cloud-providers/#openstack> ↗



Warning

The file `my-cluster/cloud/openstack/openstack.conf` must not be freely accessible. Please remember to set proper file permissions for it, for example `600`.

5.2.3.2 Example OpenStack Cloud Provider Configuration

You can find the required parameters in OpenStack RC File v3.

```
[Global]
auth-url=<OS_AUTH_URL> ①
username=<OS_USERNAME> ②
password=<OS_PASSWORD> ③
tenant-id=<OS_PROJECT_ID> ④
domain-name=<OS_USER_DOMAIN_NAME> ⑤
region=<OS_REGION_NAME> ⑥
ca-file="/etc/pki/trust/anchors/SUSE_Trust_Root.pem" ⑦
[LoadBalancer]
lb-version=v2 ⑧
subnet-id=<PRIVATE_SUBNET_ID> ⑨
floating-network-id=<PUBLIC_NET_ID> ⑩
create-monitor=yes ⑪
monitor-delay=1m ⑫
monitor-timeout=30s ⑬
monitor-max-retries=3 ⑭
[BlockStorage]
bs-version=v2 ⑮
ignore-volume-az=true ⑯
```

- ① (required) Specifies the URL of the Keystone API used to authenticate the user. This value can be found in Horizon (the OpenStack control panel), under Project > Access and Security > API Access > Credentials.
- ② (required) Refers to the username of a valid user set in Keystone.
- ③ (required) Refers to the password of a valid user set in Keystone.
- ④ (required) Used to specify the ID of the project where you want to create your resources.
- ⑤ (optional) Used to specify the name of the domain your user belongs to.
- ⑥ (optional) Used to specify the identifier of the region to use when running on a multi-region OpenStack cloud. A region is a general division of an OpenStack deployment.
- ⑦ (optional) Used to specify the path to your custom CA file.

- 8 (optional) Used to override automatic version detection. Valid values are v1 or v2. Where no value is provided, automatic detection will select the highest supported version exposed by the underlying OpenStack cloud.
- 9 (optional) Used to specify the ID of the subnet you want to create your load balancer on. Can be found at Network > Networks. Click on the respective network to get its subnets.
- 10 (optional) If specified, will create a floating IP for the load balancer.
- 11 (optional) Indicates whether or not to create a health monitor for the Neutron load balancer. Valid values are true and false. The default is false. When true is specified then monitor-delay, monitor-timeout, and monitor-max-retries must also be set.
- 12 (optional) The time between sending probes to members of the load balancer. Ensure that you specify a valid time unit.
- 13 (optional) Maximum time for a monitor to wait for a ping reply before it times out. The value must be less than the delay value. Ensure that you specify a valid time unit.
- 14 (optional) Number of permissible ping failures before changing the load balancer member's status to INACTIVE. Must be a number between 1 and 10.
- 15 (optional) Used to override automatic version detection. Valid values are v1, v2, v3 and auto. When auto is specified, automatic detection will select the highest supported version exposed by the underlying OpenStack cloud.
- 16 (optional) Influences availability zone, use when attaching Cinder volumes. When Nova and Cinder have different availability zones, this should be set to true.

After setting options in the `openstack.conf` file, please proceed with [Section 5.2.5, "Cluster Bootstrap"](#).

Important

When cloud provider integration is enabled, it's very important to bootstrap and join nodes with the same node names that they have inside `Openstack`, as these names will be used by the `Openstack` cloud controller manager to reconcile node metadata.

5.2.3.3 Integrate External LDAP TLS

1. Open the `Dex ConfigMap` in `my-cluster/addons/dex/dex.yaml`
2. Adapt the `ConfigMap` by adding LDAP configuration to the connector section of the `config.yaml` file. For detailed configurations for the LDAP connector, refer to <https://github.com/dexidp/dex/blob/v2.16.0/Documentation/connectors/ldap.md>.

Example LDAP connector

```
connectors:  
- type: ldap  
  id: 389ds  
  name: 389ds  
  config:  
    host: ldap.example.org:636 ① ②  
    rootCAData: <BASE64_ENCODED_PEM_FILE> ③  
    bindDN: cn=user-admin,ou=Users,dc=example,dc=org ④  
    bindPW: <BIND_DN_PASSWORD> ⑤  
    usernamePrompt: Email Address ⑥  
    userSearch:  
      baseDN: ou=Users,dc=example,dc=org ⑦  
      filter: "(objectClass=person)" ⑧  
      username: mail ⑨  
      idAttr: DN ⑩  
      emailAttr: mail ⑪  
      nameAttr: cn ⑫
```

- ① Host name of LDAP server reachable from the cluster.
- ② The port on which to connect to the host (for example StartTLS: 389, TLS: 636).
- ③ LDAP server base64 encoded root CA certificate file (for example `cat <root-ca-pem-file> | base64 | awk '{print}' ORS='' && echo`)
- ④ Bind DN of user that can do user searches.
- ⑤ Password of the user.
- ⑥ Label of LDAP attribute users will enter to identify themselves (for example username).
- ⑦ BaseDN where users are located (for example ou=Users,dc=example,dc=org).
- ⑧ Filter to specify type of user objects (for example "(objectClass=person)").
- ⑨ Attribute users will enter to identify themselves (for example mail).
- ⑩ Attribute used to identify user within the system (for example DN).
- ⑪ Attribute containing the user's email.

12 Attribute used as username within OIDC tokens.

Besides the LDAP connector you can also set up other connectors. For additional connectors, refer to the available connector configurations in the Dex repository: <https://github.com/dex-idp/dex/tree/v2.16.0/Documentation/connectors>.

5.2.3.4 Prevent Nodes Running Special Workloads from Being Rebooted

Some nodes might run specially treated workloads (pods).

To prevent downtime of those workloads and the respective node, it is possible to flag the pod with `--blocking-pod-selector=<POD_NAME>`. Any node running this workload will not be rebooted via `kured` and needs to be rebooted manually.

1. Open the `kured` deployment in `my-cluster/addons/kured/kured.yaml`
2. Adapt the `DaemonSet` by adding one of the following flags to the `command` section of the `kured` container:

```
---
apiVersion: apps/v1
kind: DaemonSet
...
spec:
  ...
  ...
  containers:
    ...
    command:
      - /usr/bin/kured
      - --blocking-pod-selector=name=<POD_NAME>
```

You can add any key/value labels to this selector:

```
--blocking-pod-selector=<LABEL_KEY_1>=<LABEL_VALUE_1>,<LABEL_KEY_2>=<LABEL_VALUE_2>
```

Alternatively you can adapt the `kured` `DaemonSet` also later during runtime (after bootstrap) by editing `my-cluster/addons/kured/kured.yaml` and executing:

```
kubectl apply -f my-cluster/addons/kured/kured.yaml
```

This will restart all `kured` pods with the additional configuration flags.

5.2.4 Prevent Nodes with Any Prometheus Alerts from Being Rebooted



Note

By default, **any** prometheus alert blocks a node from reboot. However you can filter specific alerts to be ignored via the `--alert-filter-regexp` flag.

1. Open the `kured` deployment in `my-cluster/addons/kured/kured.yaml`
2. Adapt the `DaemonSet` by adding one of the following flags to the `command` section of the `kured` container:

```
---
apiVersion: apps/v1
kind: DaemonSet
...
spec:
  ...
  ...
  containers:
    ...
    command:
      - /usr/bin/kured
      - --prometheus-url=<PROMETHEUS_SERVER_URL>
      - --alert-filter-regexp=^(RebootRequired|AnotherBenignAlert|...$
```



Important

The `<PROMETHEUS_SERVER_URL>` needs to contain the protocol (`http://` or `https://`)

Alternatively you can adapt the `kured` `DaemonSet` also later during runtime (after bootstrap) by editing `my-cluster/addons/kured/kured.yaml` and executing:

```
kubectl apply -f my-cluster/addons/kured/kured.yaml
```

This will restart all `kured` pods with the additional configuration flags.

5.2.5 Cluster Bootstrap

1. Switch to the new directory.
2. Now bootstrap a master node. For `--target` enter the FQDN of your first master node. Replace `<NODE_NAME>` with a unique identifier, for example, "master-one".



Warning: Secure configuration files access

The directory created during this step contains configuration files that allow full administrator access to your cluster. Apply best practices for access control to this folder.



Tip: Custom root CA certificate

During cluster bootstrap, `skuba` automatically generates a root CA certificate. You can however also deploy the Kubernetes cluster with your own custom root CA certificate.

Please refer to the *SUSE CaaS Platform Admin Guide* for more information on custom certificates.



Warning

Please plan carefully when deploying with a custom root CA certificate. This certificate can not be reconfigured once deployed and requires a full re-installation of the cluster to replace.

```
cd my-cluster
skuba node bootstrap --user sles --sudo --target <IP/FQDN> <NODE_NAME>
```

This will bootstrap the specified node as the first master in the cluster. The process will generate authentication certificates and the `admin.conf` file that is used for authentication against the cluster. The files will be stored in the `my-cluster` directory specified in step one.

3. Add additional master nodes to the cluster.

Replace the `<IP/FQDN>` with the IP for the machine. Replace `<NODE_NAME>` with a unique identifier, for example, "master-two".

```
skuba node join --role master --user sles --sudo --target <IP/FQDN> <NODE_NAME>
```

4. Add a worker to the cluster:

Replace the `<IP/FQDN>` with the IP for the machine. Replace `<NODE_NAME>` with a unique identifier, for example, "worker-one".

```
skuba node join --role worker --user sles --sudo --target <IP/FQDN> <NODE_NAME>
```

5. Verify that the nodes have been added:

```
skuba cluster status
```

The output should look like this:

NAME	OS-IMAGE	KERNEL-VERSION	CONTAINER-
master-one	SUSE Linux Enterprise Server 15 SP1	4.12.14-110-default	cri-
o://1.13.3	<none>	<none>	
worker-one	SUSE Linux Enterprise Server 15 SP1	4.12.14-110-default	cri-
o://1.13.3	<none>	<none>	

Important

The IP/FQDN must be reachable by every node of the cluster and therefore 127.0.0.1/localhost cannot be used.

5.3 Using kubectl

You can install and use `kubectl` by installing the `kubernetes-client` package from the SUSE CaaS Platform extension.

```
sudo zypper in kubernetes-client
```

Tip

Alternatively you can install from upstream: <https://kubernetes.io/docs/tasks/tools/install-kubectl/>.

To talk to your cluster, you must be in the `my-cluster` directory when running commands so it can find the `admin.conf` file.



Tip: Setting up kubeconfig

To make usage of Kubernetes tools easier, you can store a copy of the `admin.conf` file as `kubeconfig` (<https://kubernetes.io/docs/concepts/configuration/organize-cluster-access-kubeconfig/>)⁷.

```
mkdir -p ~/.kube
cp admin.conf ~/.kube/config
```



Warning

The configuration file contains sensitive information and must be handled in a secure fashion. Copying it to a shared user directory might grant access to unwanted users.

You can run commands against your cluster like usual. For example:

- `kubectl get nodes -o wide`
- or
- `kubectl get pods --all-namespaces`

```
# kubectl get pods --all-namespaces
NAMESPACE      NAME                                     READY   STATUS    RESTARTS
AGE
kube-system    coredns-86c58d9df4-5zftb              1/1     Running   0
2m
kube-system    coredns-86c58d9df4-fct4m              1/1     Running   0
2m
kube-system    etcd-my-master                         1/1     Running   0
1m
kube-system    kube-apiserver-my-master               1/1     Running   0
1m
kube-system    kube-controller-manager-my-master      1/1     Running   0
1m
kube-system    cilium-operator-7d6dddbf5-dmbhv       1/1     Running   0
51s
```

kube-system	cilium-qjt9h	1/1	Running	0
53s				
kube-system	cilium-szkqc	1/1	Running	0
2m				
kube-system	kube-proxy-5qxnt	1/1	Running	0
2m				
kube-system	kube-proxy-746ws	1/1	Running	0
53s				
kube-system	kube-scheduler-my-master	1/1	Running	0
1m				
kube-system	kured-ztnfj	1/1	Running	0
2m				
kube-system	kured-zv696	1/1	Running	0
2m				
kube-system	oidc-dex-55fc689dc-b9bxw	1/1	Running	0
2m				
kube-system	oidc-gangway-7b7fbdbdf-11618	1/1	Running	0
2m				

6 Network Security

6.1 Cilium

Cilium is open source software for transparently securing the network connectivity between application services deployed using Linux container management platforms. The main characteristic of Cilium is that it uses the extended Berkeley Packet Filter as a mechanism for filtering incoming and outgoing packets in containers. The Berkeley Packet Filter (BPF) is a bytecode interpreter inside the Linux kernel which allows to write programs to analyze and filter network packets and to monitor system function calls. Cilium translates network/security policies into BPF programs, which are loaded into the kernel. This means that security policies can be applied and updated without any changes to the application code or container configuration.

In SUSE CaaS Platform 4.0.3, Cilium is deployed automatically with the Platform installation. Please refer to the official Cilium documentation for instructions on how to secure various parts of your infrastructure. The following chapters are specifically recommended for SUSE CaaS Platform users.

General introduction to Cilium

<https://cilium.readthedocs.io/en/v1.5/intro/> 

Securing traffic for HTTP servers and APIs

<https://cilium.readthedocs.io/en/v1.5/gettingstarted/http/> 

Restricting the network traffic to specific DNS queries or domains

<https://cilium.readthedocs.io/en/v1.5/gettingstarted/dns/> 

Securing traffic for the Kafka protocol

<https://cilium.readthedocs.io/en/v1.5/gettingstarted/kafka/> 

Securing applications using gRPC

<https://cilium.readthedocs.io/en/v1.5/gettingstarted/grpc/> 

Securing Cassandra databases

<https://cilium.readthedocs.io/en/v1.5/gettingstarted/cassandra/> 

Securing Memcached

<https://cilium.readthedocs.io/en/v1.5/gettingstarted/memcached/> 

Making security policies on clusters on AWS

<https://cilium.readthedocs.io/en/v1.5/gettingstarted/aws/> 

A GNU Licenses

This appendix contains the GNU Free Documentation License version 1.2.

A.1 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 non-commercially. 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 to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material

on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <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.