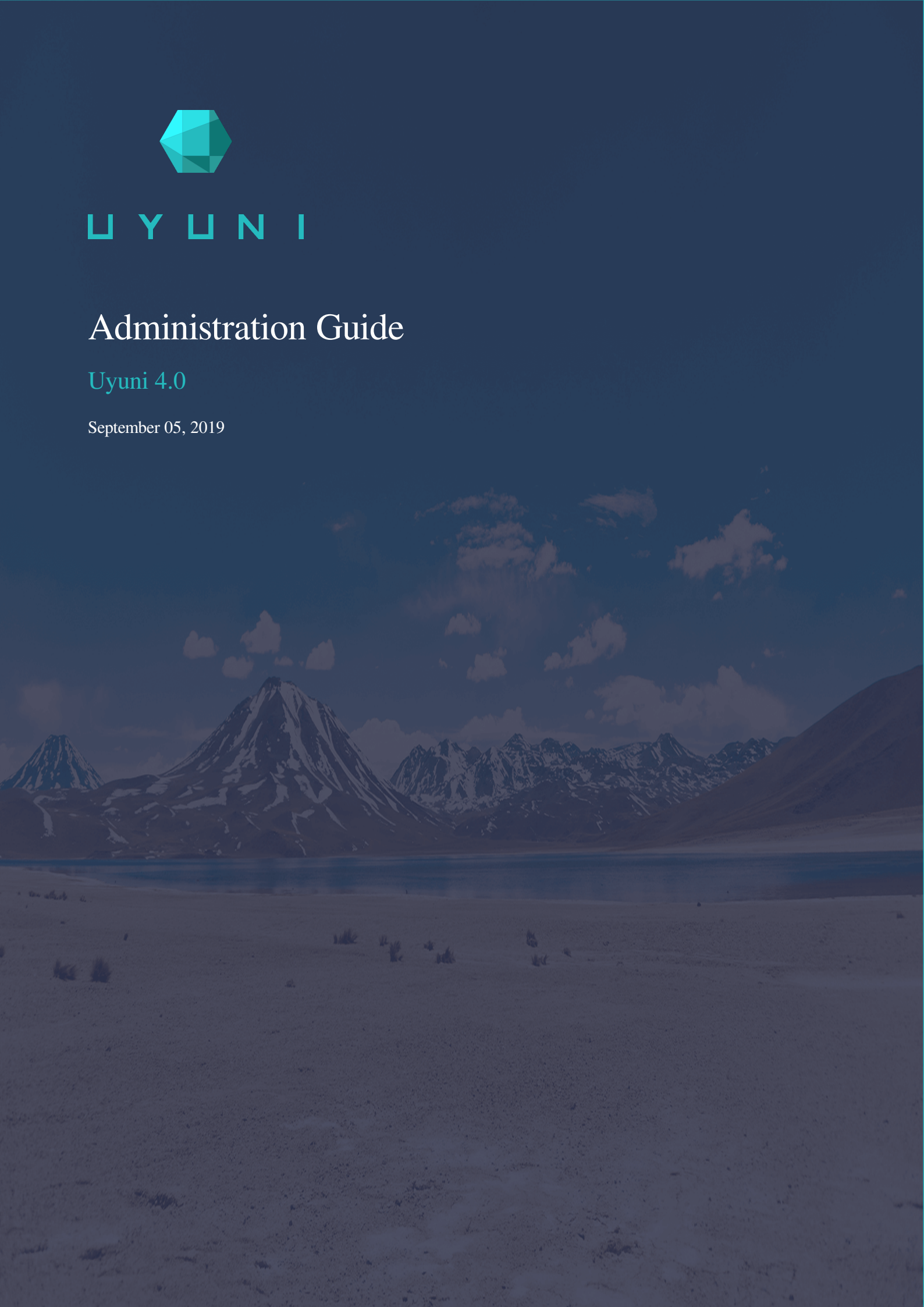


Administration Guide

Uyuni 4.0

September 05, 2019

Table of Contents

GNU Free Documentation License 1
Introduction 8
Image Building and Management 9
- Image Building Overview . . . 9
- Container Images . . . 9
- OS Images . . . 17
- Listing Image Profiles Available for Building . . . 25

Live Patching with SUSE Manager 26
- Live Patching on SLES 15 . . . 26
- Live Patching on SLES 12 . . . 29

Monitoring with Prometheus 32
- Prometheus Metrics . . . 32
- PromQL . . . 33
- Exporters . . . 33
- Install and Configure Prometheus . . . 34
- Monitoring Salt Clients . . . 35
- Enable and Configure Monitoring . . . 35
- Visualization with Grafana . . . 37

Kubernetes 39
- Prerequisites . . . 39
- Requirements . . . 39
- Register Kubernetes as a Virtual Host Manager . . . 39
- View the List of Nodes in a Cluster . . . 39
- Obtain Runtime Data about Images . . . 40
- Build an image for deployment in Kubernetes . . . 40
- Import a Previously Deployed Image in Kubernetes . . . 40
- Obtain Additional Runtime Data . . . 41
- Rebuild a Previously Deployed Image in Kubernetes . . . 41
- Role Based Access Control Permissions and Certificate Data . . . 41

Public Cloud 43
- Instance Requirements . . . 43
- Network Setup . . . 43
- Registration of Cloned Systems . . . 48

Inter-Server Synchronization 50
Set up a Client to Master Validation Fingerprint 51
Signing Repository Metadata 52
Mirror Source Packages 54
Authentication Methods 55
- Authentication Via PAM . . . 55
- Authentication Via Single Sign-On (SSO) . . . 56

Using a Custom SSL Certificate 59
- Prerequisites . . . 59
- Setup . . . 59
- Using a Custom Certificate with SUSE Manager Proxy . . . 60

Backup and Restore 61
- Backing up Uyuni . . . 61

Administering the Database with smdba . . . 63
Database Backup with smdba . . . 64
Restoring from Backup . . . 66
Archive Log Settings . . . 66
Retrieving an Overview of Occupied Database Space . . . 67
Moving the Database . . . 67
Recovering from a Crashed Root Partition . . . 69
Database Connection Information . . . 69
Content Lifecycle Management 70
Create a Content Lifecycle Project . . . 70
Advanced: Filter Types . . . 71
Build a Content Lifecycle Project . . . 71
Promote Environments . . . 72
Assign Systems to Environments . . . 72
Tuning Changelogs 73
Maintenance Window Checklist . . . 73
Troubleshooting 75
Troubleshooting 76
Producing Reports . . . 76
Troubleshooting Corrupt Repositories . . . 78
Troubleshooting Disk Space . . . 78
Troubleshooting Local Issuer Certificates . . . 78
Troubleshooting OSAD and jabberd . . . 79
Troubleshooting Package Inconsistencies . . . 81
Troubleshooting Registering Cloned Clients . . . 82
Troubleshooting RPC Connection Timeouts . . . 84
Troubleshooting the Saltboot Formula . . . 85
AutoYast Example File 86
Minimalist AutoYaST Profile for Automated Installations and Useful Enhancements . . . 86

GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

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

1. APPLICABILITY AND DEFINITIONS

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

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

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

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections

then there are none.

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

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

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

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

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

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

2. VERBATIM COPYING

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

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

3. COPYING IN QUANTITY

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

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

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

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

4. MODIFICATIONS

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

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

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

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

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

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

5. COMBINING DOCUMENTS

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

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

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

6. COLLECTIONS OF DOCUMENTS

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

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

7. AGGREGATION WITH INDEPENDENT WORKS

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

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

8. TRANSLATION

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

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

9. TERMINATION

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

10. FUTURE REVISIONS OF THIS LICENSE

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

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
   Permission is granted to copy, distribute and/or modify this document
   under the terms of the GNU Free Documentation License, Version 1.2
   or any later version published by the Free Software Foundation;
   with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
   A copy of the license is included in the section entitled{ldquo}GNU
   Free Documentation License{rdquo}.
```

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.

Introduction

This book provides guidance on performing common administrative tasks on Uyuni.

Image Building and Management

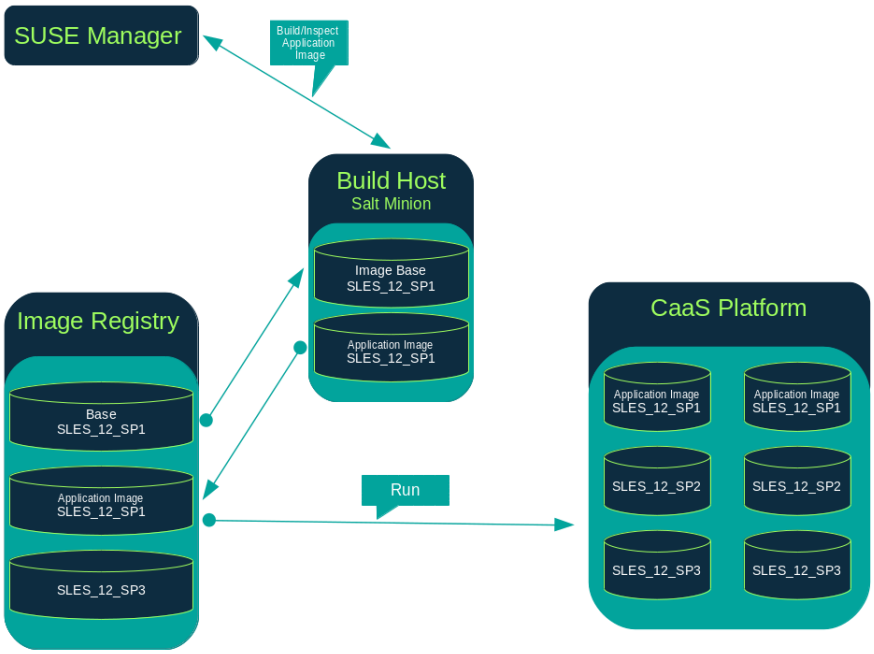
Image Building Overview

Uyuni enables system administrators to build containers, systems, and virtual images. Uyuni helps with creating image stores and managing image profiles.

Uyuni supports two distinct build types:

- Dockerfile-for more information, see Container Images
- Kiwi image system-for more information, see OS Images

Container Images



Requirements

The containers feature is available for Salt clients running SUSE Linux Enterprise Server 12 or later. Before you begin, ensure your environment meets these requirements:

- An existing external GitHub or internal GitLab repository containing a dockerfile and configuration scripts (example scripts are provided in this chapter).
- A properly configured image registry.

Registry Provider Solutions

If you require a private image registry you can use an open source solution such as `Portus`. For additional information on setting up Portus as a registry provider, see the Portus Documentation.

For more information on Containers or CaaS Platform, see:

- SUSE Linux Enterprise Server Docker Guide
- SUSE CaaS Platform 3 Documentation

Creating a Build Host

To build images with Uyuni, you will need to create and configure a build host. Container build hosts are Salt clients running SUSE Linux Enterprise 12 or later. This section guides you through the initial configuration for a build host.

From the Uyuni Web UI, perform these steps to configure a build host:

1. Select a Salt client to be designated as a build host from the **Systems › Overview** page.
2. From the `System Details` page of the selected client assign the containers modules. Go to **Software › Software Channels** and enable the containers module (for example, `SLE-Module-Containers15-Pool` and `SLE-Module-Containers15-Updates`). Confirm by clicking **`[Change Subscriptions]`**.
3. From the **System Details › Properties** page, enable `Container Build Host` from the `Add-on System Types` list and confirm by clicking **`[Update Properties]`**.
4. Install all required packages by applying `Highstate`. From the system details page select **States › Highstate** and click `Apply Highstate`. Alternatively, apply Highstate from the Uyuni Server command line:

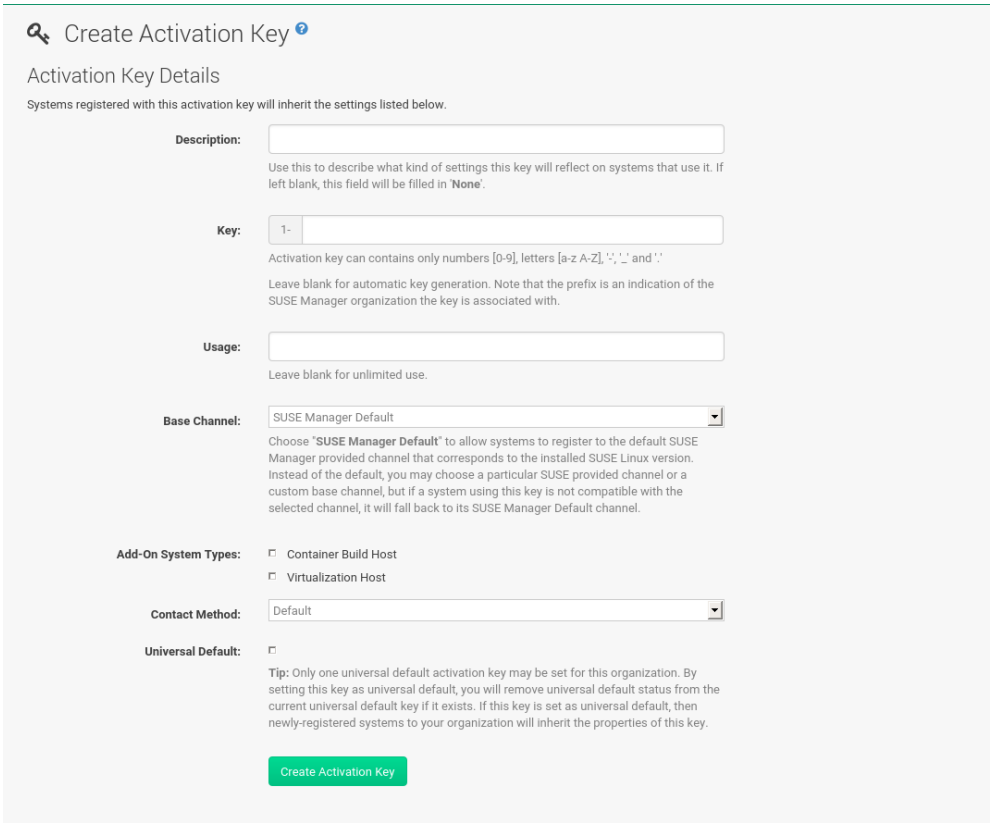
```
salt '$your_client' state.highstate
```

Define Container Build Channels with an Activation Key

Create an activation key associated with the channel that your images will use.

Relationship Between Activation Keys and Image Profiles

To build containers, you will need an activation key that is associated with a channel other than `SUSE Manager Default`.

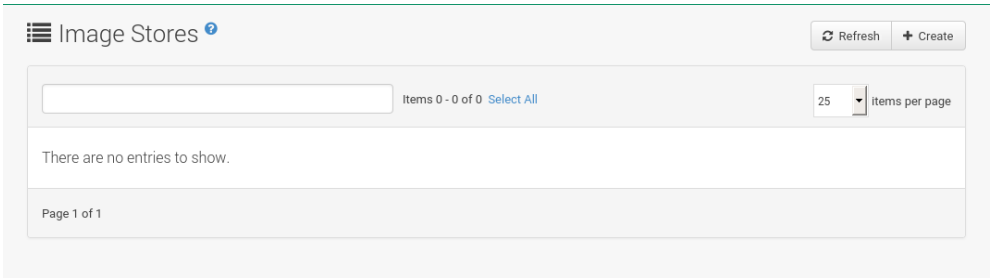


1. Select **Systems › Activation Keys**.
2. Click **[Create Key]**.
3. Enter a `Description` and a `Key` name. Use the drop-down menu to select the `Base Channel` to associate with this key.
4. Confirm with **[Create Activation Key]**.

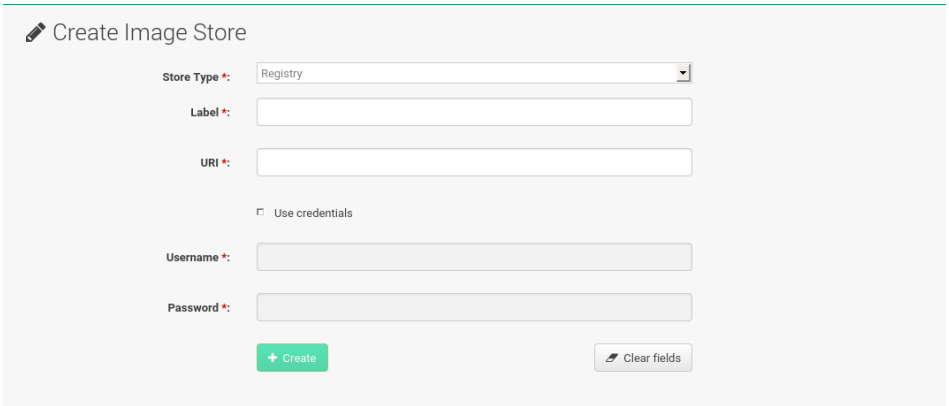
For more information, see [bp.key.managment].

Creating an Image Store

Define a location to store all of your images by creating an image store.



1. Select **Images › Stores**.
2. Click `Create` to create a new store.



3. Uyuni currently provides support only for the `Registry` store type. Define a name for the image store in the `Label` field.
4. Provide the path to your image registry by filling in the `URI` field, as a fully qualified domain name (FQDN) for the container registry host (whether internal or external).

```
registry.example.com
```

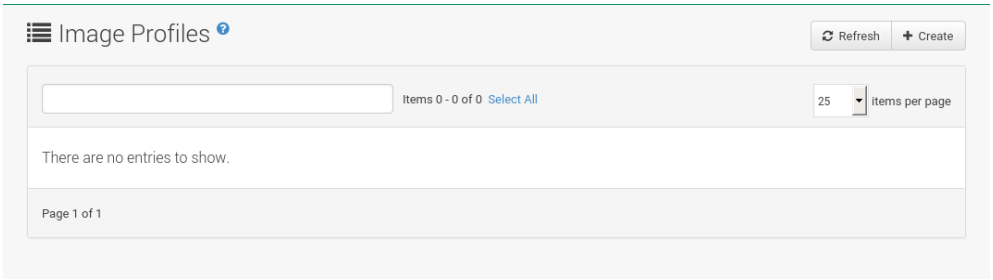
The Registry URI can also be used to specify an image store on a registry that is already in use.

```
registry.example.com:5000/myregistry/myproject
```

5. Click **`[Create]`** to add the new image store.

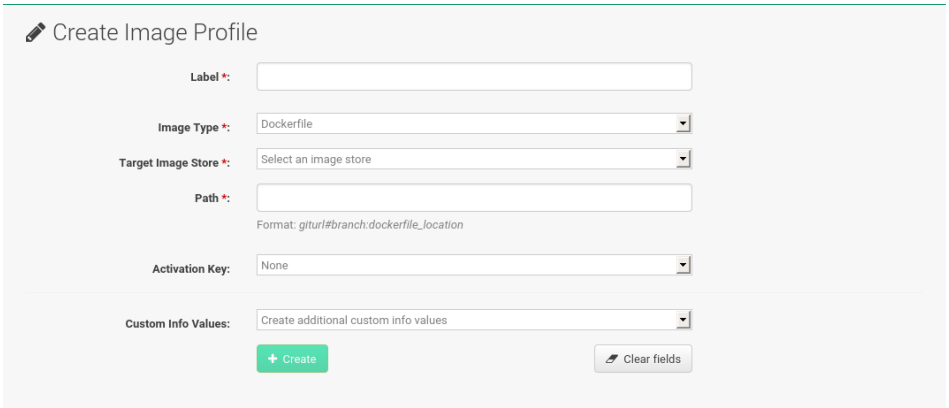
Creating an Image Profile

Manage image profiles from the `Image Profile` page.



Procedure: Create an Image Profile

1. To create an image profile select **Images › Profiles** and click **`[Create]`**.



2. Provide a name for the image profile by filling in the `Label` field.

Only lowercase characters are permitted in container labels. If your container image tag is in a format such as `myproject/myimage`, make sure your image store registry URI contains the `/myproject` suffix.

3. Use a dockerfile as the `Image Type`.
4. Use the drop-down menu to select your registry from the `Target Image Store` field.
5. Enter a Github or Gitlab repository URL (http, https, or token authentication) in the `Path` field using one of the following formats:

Github Path Options

- Github single user project repository

```
https://github.com/USER/project.git#branchname:folder
```

- Github organization project repository

```
https://github.com/ORG/project.git#branchname:folder
```

- Github token authentication

  If your git repository is private and not publicly accessible, you need to modify the profile's git URL to include authentication. Use this URL format to authenticate with a Github token:

```
https://USER:<AUTHENTICATION_TOKEN>@github.com/USER/project.git#master:/container/
```

Gitlab Path Options

- Gitlab single user project repository

```
https://gitlab.example.com/USER/project.git#master:/container/
```

- Gitlab groups project repository

```
https://gitlab.example.com/GROUP/project.git#master:/container/
```

- Gitlab token authentication

  If your git repository is private and not publicly accessible, you need to modify the profile's git URL to include authentication. Use this URL format to authenticate with a Gitlab token:

```
https://gitlab-ci-
token:<AUTHENTICATION_TOKEN>@gitlab.example.com/USER/project.git#master:/container/
```

Specifying a Github or Gitlab Branch

If a branch is not specified, the `master` branch will be used by default. If a `folder` is not specified the image sources (dockerfile sources) are expected to be in the root directory of the Github or Gitlab checkout.

1. Select an `Activation Key`. Activation Keys ensure that images using a profile are assigned to the correct channel and packages.

Relationship Between Activation Keys and Image Profiles

When you associate an activation key with an image profile you are ensuring any image using the profile will use the correct software channel and any packages in the channel.

2. Click the **[Create]** button.

Example Dockerfile and add_packages Script

This section contains an example dockerfile. You specify a dockerfile that will be used during image building when creating an image profile. A dockerfile and any associated scripts should be stored within an external or internal Github or Gitlab repository.

The dockerfile provides access to a specific repository version served by Uyuni. The following example dockerfile is used by Uyuni to trigger a build job on a build host. The `ARG` parameters ensure that the built image is associated with the desired repository version served by Uyuni. The `ARG` parameters also allow you to build image versions of SUSE Linux Enterprise Server which may differ from the version of SUSE Linux Enterprise Server used by the build host itself.

For example: The `ARG repo` parameter and the `echo` command pointing to the repository file, creates and then injects the correct path into the repository file for the desired channel version.

The repository version is determined by the activation key that you assigned to your image profile.

```
FROM registry.example.com/sles12sp2
MAINTAINER Tux Administrator "tux@example.com"

### Begin: These lines Required for use with {productname}

ARG repo
ARG cert

# Add the correct certificate
RUN echo "$cert" > /etc/pki/trust/anchors/RHN-ORG-TRUSTED-SSL-CERT.pem

# Update certificate trust store
RUN update-ca-certificates

# Add the repository path to the image
RUN echo "$repo" > /etc/zypp/repos.d/susemanager:dockerbuild.repo

### End: These lines required for use with {productname}

# Add the package script
ADD add_packages.sh /root/add_packages.sh

# Run the package script
RUN /root/add_packages.sh

# After building remove the repository path from image
RUN rm -f /etc/zypp/repos.d/susemanager:dockerbuild.repo
```

This is an example `add_packages.sh` script for use with your dockerfile:

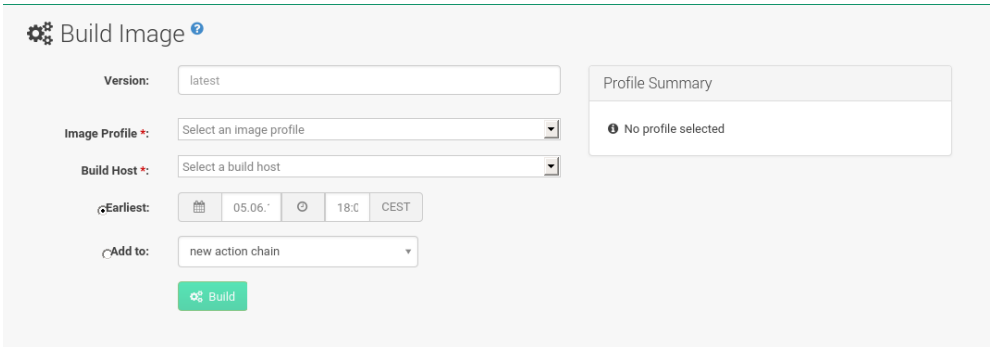
```
#!/bin/bash
set -e

zypper --non-interactive --gpg-auto-import-keys ref

zypper --non-interactive in python python-xml aaa_base aaa_base-extras net-tools timezone vim
less sudo tar
```

Packages Required for Inspecting Your Images

To inspect images and provide the package and product list of a container to the Uyuni Web UI you must install python and python-xml within the container. Without these packages your images will still build, but the package and product list will be unavailable from the Web UI.

Building an Image

There are two ways to build an image. You can select **Images › Build** from the left navigation bar, or click the build icon in the **Images › Profiles** list.



Procedure: Building an Image

1. Select **Images › Build**.
2. Add a different tag name if you want a version other than the default `latest` (only relevant to containers).
3. Select `Build Profile` and `Build Host`.

Profile Summary

Notice the `Profile Summary` to the right of the build fields. When you have selected a build profile, detailed information about the selected profile will be displayed in this area.

4. To schedule a build click the **`[Build]`** button.

Importing an Image

You can import and inspect arbitrary images. Select **Images › Image List** from the left navigation bar. Complete the text boxes of the `Import` dialog. When it has processed, the imported image will be listed on the `Image List` page.

Procedure: Importing an Image

1. From **Images › Image list** click **`[Import]`** to open the `Import Image` dialog.
2. In the `Import Image` dialog complete these fields:

Image store

The registry from where the image will be pulled for inspection.

Image name

The name of the image in the registry.

Image version

The version of the image in the registry.

Build host

The build host that will pull and inspect the image.

Activation key

The activation key that provides the path to the software channel that the image will be inspected with.

3. For confirmation, click **[Import]**.

The entry for the image is created in the database, and an `Inspect Image` action on Uyuni is scheduled.

When it has been processed, you can find the imported image in the `Image List`. It has a different icon in the `Build` column, to indicate that the image is imported. The status icon for the imported image can also be seen on the `Overview` tab for the image.

Troubleshooting

These are some known problems when working with images:

- HTTPS certificates to access the registry or the git repositories should be deployed to the client by a custom state file.
- SSH git access using Docker is currently unsupported. You may test it, but SUSE will not provide support.
- If the python and python-xml packages are not installed in your images during the build process, Salt cannot run within the container and reporting of installed packages or products will fail. This will result in an `unknown` update status.

OS Images

OS images are built by the Kiwi image system. They can be of various types: PXE, QCOW2, LiveCD images, and others.

For more information about the Kiwi build system, see the Kiwi documentation.

Requirements

The Kiwi image building feature is available for Salt clients running SUSE Linux Enterprise Server 12. It is currently not supported to build SUSE Linux Enterprise 15 images. Building SUSE Linux Enterprise 15 images requires a SUSE Linux Enterprise 15 based build host.

Kiwi image configuration files and configuration scripts must be accessible in one of these locations:

- Git repository
- HTTP hosted tarball
- Local build host directory

Example scripts are provided in Example of Kiwi Sources.

Hardware Requirements for Hosts Running OS Images

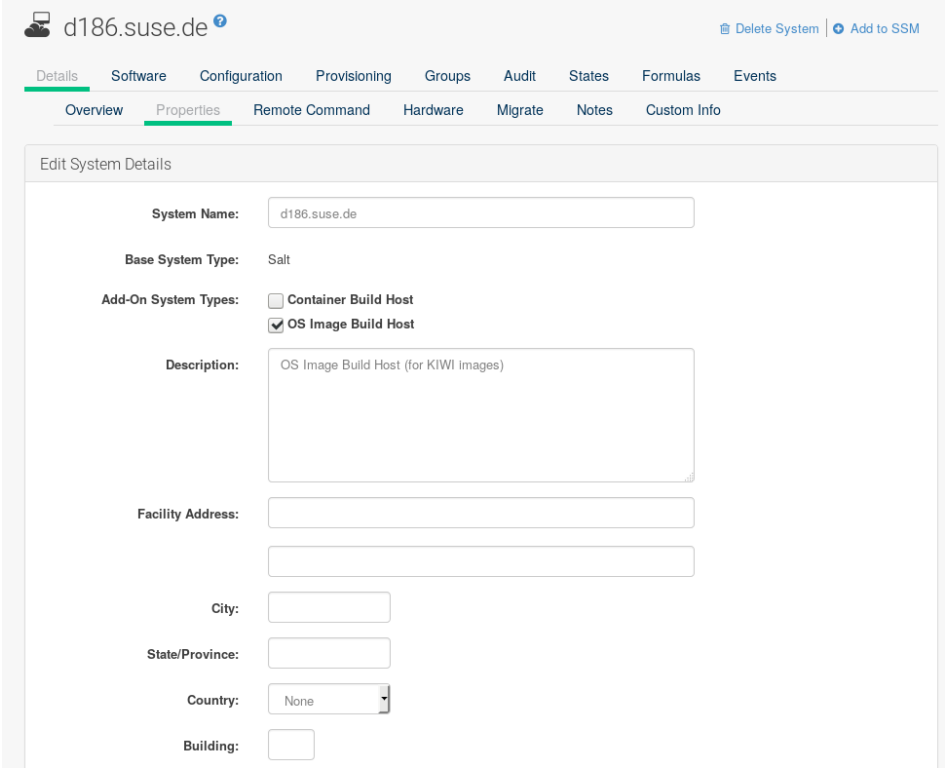
Hosts running OS images built with Kiwi need at least 1 GB of RAM. Disk space depends on the actual size of the image. For more information, see the documentation of the underlying system.

Creating a Build Host

To build all kinds of images with Uyuni, create and configure a build host. OS image build hosts are Salt clients running SUSE Linux Enterprise Server 12 (SP3 or later) or 15. This procedure will guide you through the initial configuration for a build host.

From the Uyuni Web UI, perform these steps to configure a build host:

1. Select a client that will be designated as a build host from the **Systems** › **Overview** page.
2. From the **System Details** › **Properties** page, enable the `Add-on System Type: OS Image Build Host` and confirm with **`[Update Properties]`**.



3. From the **System Details** › **Software** › **Software Channels** page, enable `SLE-Manager-Tools12-Pool` and `SLE-Manager-Tools12-Updates` (or a later version). Schedule and click **`[Confirm]`**.
4. Install Kiwi and all required packages by applying `Highstate`. From the system details page select **States** › **Highstate** and click **`[Apply Highstate]`**. Alternatively, apply Highstate from the Uyuni Server command line:

```
salt '$your_client' state.highstate
```

Uyuni Web Server Public Certificate RPM

Build host provisioning copies the Uyuni certificate RPM to the build host. This certificate is used for accessing repositories provided by Uyuni.

The certificate is packaged in RPM by the `mgr-package-rpm-certificate-osimage` package script. The package script is called automatically during a new Uyuni installation.

When you upgrade the `spacewalk-certs-tools` package, the upgrade scenario will call the package script using the default values. However if the certificate path was changed or unavailable, you will need to call the package script manually using `--ca-cert-full-path <path_to_certificate>` after the upgrade procedure has finished.

Listing 1. Package script call example

```
/usr/sbin/mgr-package-rpm-certificate-osimage --ca-cert-full-path /root/ssl-build/RHN-ORG-
TRUSTED-SSL-CERT
```

The RPM package with the certificate is stored in a salt-accessible directory such as `/usr/share/susemanager/salt/images/rhn-org-trusted-ssl-cert-osimage-1.0-1.noarch.rpm`.

The RPM package with the certificate is provided in the local build host repository `/var/lib/Kiwi/repo`.

The RPM Package with the Uyuni Certificate Must Be Specified in the Build Source

Make sure your build source Kiwi configuration contains `rhn-org-trusted-ssl-cert-osimage` as a required package in the `bootstrap` section.

Listing 2. config.xml

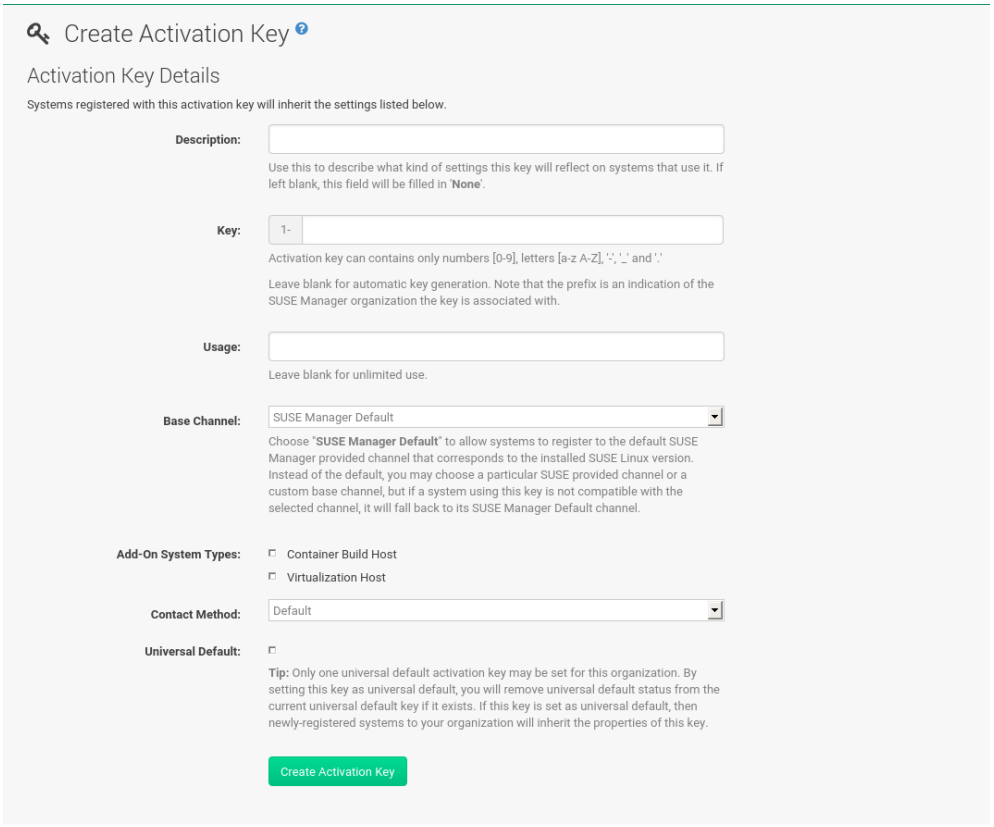
```
...
  <packages type="bootstrap">
    ...
    <package name="rhn-org-trusted-ssl-cert-osimage"
bootinclude="true"/>
  </packages>
...
```

Define Kiwi Build Channels with an Activation Key

Create an activation key associated with the channel that your images will use. Activation keys are mandatory for OS image building.

Relationship Between Activation Keys and Image Profiles

To build OS images, you will need an activation key that is associated with a channel other than `SUSE Manager Default`.



1. In the Web UI, select **Systems › Activation Keys**.
2. Click `Create Key`.
3. Enter a `Description`, a `Key` name, and use the drop-down box to select a `Base Channel` to associate with the key.
4. Confirm with **`[Create Activation Key]`**.

For more information, see [bp.key.managment].

Image Store

OS images can require a significant amount of storage space. Therefore, we recommended that the OS image store is located on a partition of its own or on a Btrfs subvolume, separate from the root partition. By default, the image store will be located at `/srv/www/os-images`.

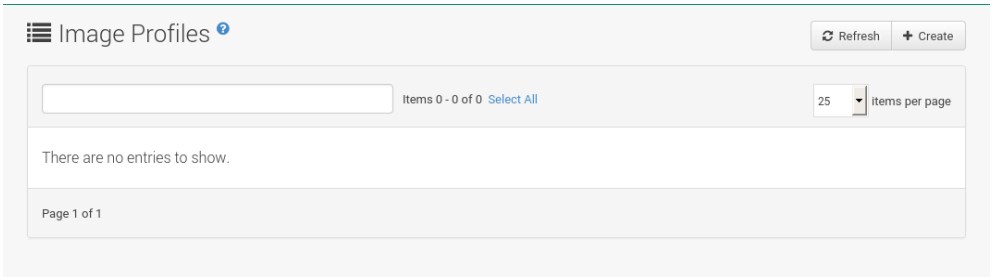
Image Stores for Kiwi Build Type

Image stores for Kiwi build type, used to build system, virtual, and other images, are not supported yet.

Images are always stored in `/srv/www/os-images/<organization id>` and are accessible via HTTP/HTTPS `https://<susemanager_host>/os-images/<organization id>`.

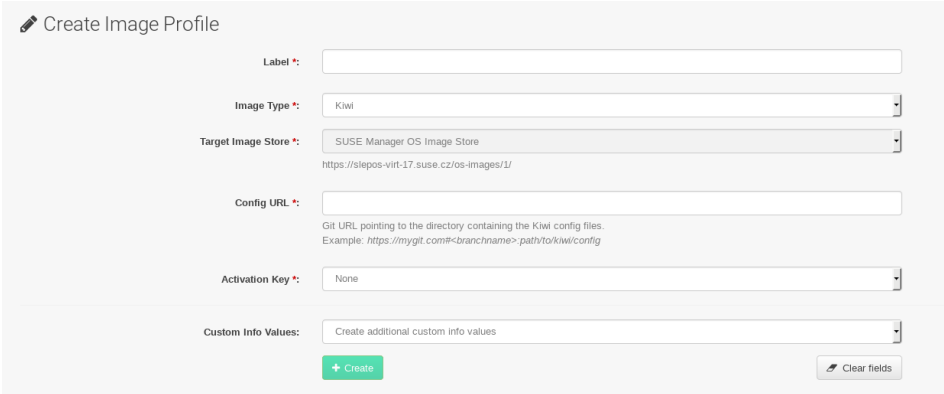
Creating an Image Profile

Manage image profiles using the Web UI.



Procedure: Create an Image Profile

1. To create an image profile select from **Images › Profiles** and click **`[Create]`**.



2. In the `Label` field, provide a name for the `Image Profile`.
3. Use `Kiwi` as the `Image Type`.
4. Image store is automatically selected.
5. Enter a `Config URL` to the directory containing the Kiwi configuration files:
    a. Git URI
    b. HTTPS tarball
    c. Path to build host local directory
6. Select an `Activation Key`. Activation keys ensure that images using a profile are assigned to the correct channel and packages.

Relationship Between Activation Keys and Image Profiles

When you associate an activation key with an image profile you are ensuring any image using the profile will use the correct software channel and any packages in the channel.

7. Confirm with the **[Create]** button.

Source format options

- Git/HTTP(S) URL to the repository

  URL to the Git repository containing the sources of the image to be built. Depending on the layout of the repository the URL can be:

  ```
  https://github.com/SUSE/manager-build-profiles
  ```

  You can specify a branch after the `#` character in the URL. In this example, we use the `master` branch:

  ```
  https://github.com/SUSE/manager-build-profiles#master
  ```

  You can specify a directory that contains the image sources after the `:` character. In this example, we use `OSImage/POS_Image-JeOS6`:

  ```
  https://github.com/SUSE/manager-build-profiles#master:OSImage/POS_Image-JeOS6
  ```

- HTTP(S) URL to the tarball

  URL to the tar archive, compressed or uncompressed, hosted on the webserver.

  ```
  https://myimagesourceserver.example.org/MyKiwiImage.tar.gz
  ```

- Path to the directory on the build host

  Enter the path to the directory with the Kiwi build system sources. This directory must be present on the selected build host.

  ```
  /var/lib/Kiwi/MyKiwiImage
  ```

Example of Kiwi Sources

Kiwi sources consist at least of `config.xml`. Usually, `config.sh` and `images.sh` are present as well.

Sources can also contain files to be installed in the final image under the `root` subdirectory.

For information about the Kiwi build system, see the Kiwi documentation.

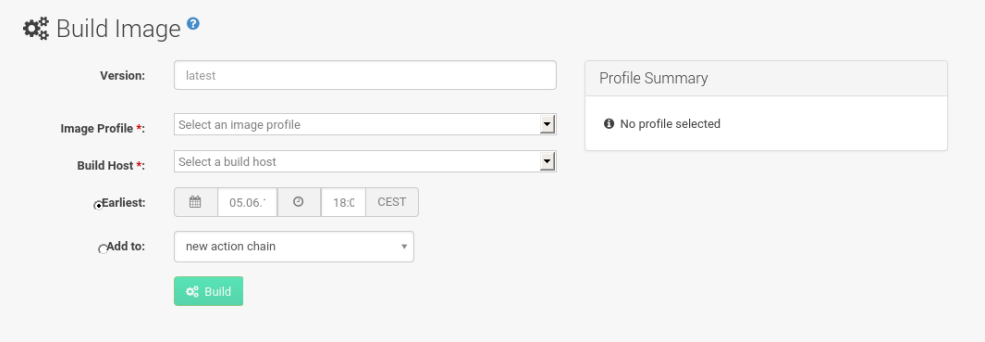
SUSE provides examples of fully functional image sources at the SUSE/manager-build-profiles public GitHub repository.

Listing 3. Example of JeOS config.xml

```
<?xml version="1.0" encoding="utf-8"?>

<image schemaversion="6.1" name="POS_Image_JeOS6">
    <description type="system">
        <author>Admin User</author>
        <contact>noemail@example.com</contact>
        <specification>SUSE Linux Enterprise 12 SP3 JeOS</specification>
    </description>
    <preferences>
        <version>6.0.0</version>
        <packagemanager>zypper</packagemanager>
        <bootsplash-theme>SLE</bootsplash-theme>
        <bootloader-theme>SLE</bootloader-theme>

        <locale>en_US</locale>
        <keytable>us.map.gz</keytable>
        <timezone>Europe/Berlin</timezone>
        <hwclock>utc</hwclock>

        <rpm-excludedocs>true</rpm-excludedocs>
        <type boot="saltboot/suse-SLES12" bootloader="grub2" checkprebuilt="true"
compressed="false" filesystem="ext3" fsmountoptions="acl" fsnocheck="true" image="pxe"
kernelcmdline="quiet"></type>
    </preferences>
    <!--    CUSTOM REPOSITORY
    <repository type="rpm-dir">
      <source path="this://repo"/>
    </repository>
    -->
    <packages type="image">
        <package name="patterns-sles-Minimal"/>
        <package name="aaa_base-extras"/> <!-- wouldn't be SUSE without that ;-) -->
        <package name="kernel-default"/>
        <package name="salt-minion"/>
        ...
    </packages>
    <packages type="bootstrap">
        ...
        <package name="sles-release"/>
        <!-- this certificate package is required to access {productname} repositories
             and is provided by {productname} automatically -->
        <package name="rhn-org-trusted-ssl-cert-osimage" bootinclude="true"/>

    </packages>
    <packages type="delete">
        <package name="mtools"/>
        <package name="initviocons"/>
        ...
    </packages>
</image>
```

Building an Image

There are two ways to build an image using the Web UI. Either select **Images › Build**, or click the build icon in the **Images › Profiles** list.



Procedure: Building an Image

1. Select **Images › Build**.
2. Add a different tag name if you want a version other than the default `latest` (applies only to containers).
3. Select the `Image Profile` and a `Build Host`.

Profile Summary

A `Profile Summary` is displayed to the right of the build fields. When you have selected a build profile, detailed information about the selected profile will show up in this area.

4. To schedule a build, click the **`[Build]`** button.

Image Inspection and Salt Integration

After the image is successfully built, the inspection phase begins. During the inspection phase SUSE Manager collects information about the image:

- List of packages installed in the image
- Checksum of the image
- Image type and other image details

If the built image type is `PXE`, a Salt pillar will also be generated. Image pillars are stored in the `/srv/susemanager/pillar_data/images/` directory and the Salt subsystem can access details about the generated image. Details include where the pillar is located and provided, image checksums, information needed for network boot, and more.

The generated pillar is available to all connected clients.

Troubleshooting

Building an image requires of several dependent steps. When the build fails, investigation of Salt states results can help you to identify the source of the failure. Usual checks when the build fails:

- The build host can access the build sources
- There is enough disk space for the image on both the build host and the Uyuni server
- The activation key has the correct channels associated with it
- The build sources used are valid
- The RPM package with the Uyuni public certificate is up to date and available at `/usr/share/susemanager/salt/images/rhn-org-trusted-ssl-cert-osimage-1.0-1.noarch.rpm`. For more on how to refresh a public certificate RPM, see Creating a Build Host.

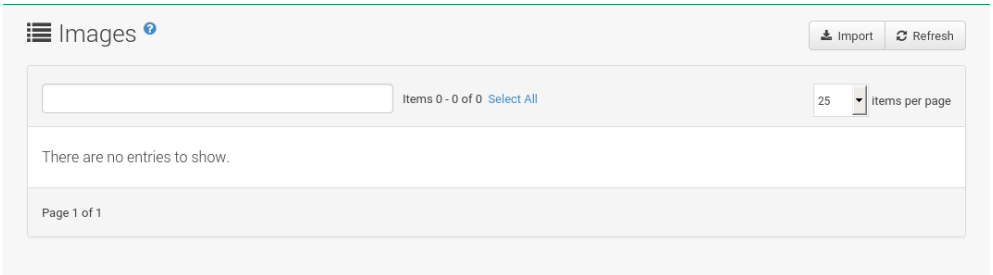
Limitations

The section contains some known issues when working with images.

- HTTPS certificates used to access the HTTP sources or Git repositories should be deployed to the client by a custom state file, or configured manually.
- Importing Kiwi-based images is not supported.

Listing Image Profiles Available for Building

To list images available for building select **Images › Image List**. A list of all images will be displayed.



Displayed data about images includes an image `Name`, its `Version` and the build `Status`. You will also see the image update status with a listing of possible patch and package updates that are available for the image.

Clicking the **`[Details]`** button on an image will provide a detailed view. The detailed view includes an exact list of relevant patches and a list of all packages installed within the image.

The patch and the package list is only available if the inspect state after a build was successful.

Live Patching with SUSE Manager

Performing a kernel update usually requires a system reboot. Common vulnerability and exposure (CVE) patches should be applied as soon as possible, but if you cannot afford the downtime, you can use Live Patching to inject these important updates and skip the need to reboot.

The procedure for setting up Live Patching is slightly different for SLES 12 and SLES 15. Both procedures are documented in this section.

Live Patching on SLES 15

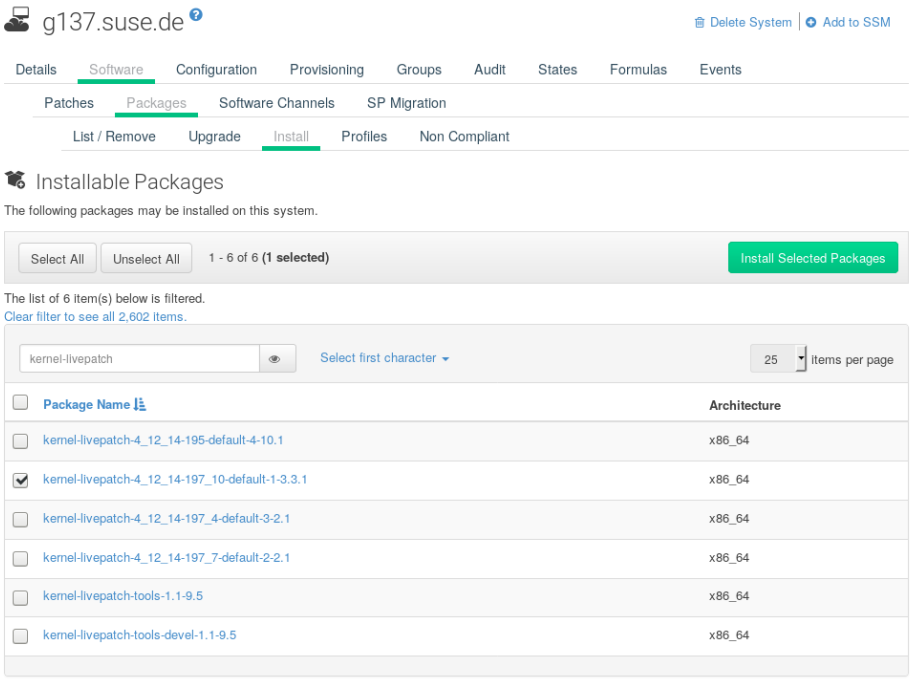
On SLES 15 systems and newer, live patching is managed by the `klp livepatch` tool.

Before you begin, ensure:

- Uyuni is fully updated
- You have one or more Salt clients running SLES 15 (SP1 or later)
- Your SLES 15 Salt clients are registered with Uyuni
- You have access to the SLES 15 channels appropriate for your architecture, including the Live Patching child channel (or channels)
- The clients are fully synchronized

Procedure: Setting up for Live Patching

1. Select the client you want to manage with Live Patching from **Systems** › **Overview**, and navigate to the **Software** › **Packages** › **Install** tab. Search for the `kernel-livepatch` package, and install it.



2. Apply the highstate to enable Live Patching, and reboot the client.
3. Repeat for each client that you want to manage with Live Patching.
4. To check that Live Patching has been enabled correctly, select the client from **Systems › System List**, and ensure that `Live Patch` appears in the `Kernel` field.

When you have the Live Patching channel installed on the client, you can clone the default vendor channel. This cloned channel will be used to manage Live Patching on your clients.

Cloned vendor channels should be prefixed by `dev` for development, `testing`, or `prod` for production. In this procedure, you will create a `dev` cloned channel, and later, you will need to promote the channel to `testing`.

Procedure: Cloning Live Patching Channels

1. At the command prompt on the client, as `root`, obtain the current package channel tree:

```
# spacewalk-manage-channel-lifecycle --list-channels
Spacewalk Username: admin
Spacewalk Password:
Channel tree:

 1. sles15-{sp-vert}-pool-x86_64
      \__ sle-live-patching15-pool-x86_64-{sp-vert}
      \__ sle-live-patching15-updates-x86_64-{sp-vert}
      \__ sle-manager-tools15-pool-x86_64-{sp-vert}
      \__ sle-manager-tools15-updates-x86_64-{sp-vert}
      \__ sles15-{sp-vert}-updates-x86_64
```

2. Use the `spacewalk-manage-channel` command with the `init` argument to automatically create a new development clone of the original vendor channel:

```
spacewalk-manage-channel-lifecycle --init -c sles15-{sp-vert}-pool-x86_64
```

3. Check that `dev-sles15-{sp-vert}-updates-x86_64` is available in your channel list.

Check the `dev` cloned channel you created, and remove any kernel updates that require a reboot.

Procedure: Removing Non-Live Kernel Patches from Cloned Channels

1. Check the current kernel version by selecting the client from **Systems › System List**, and taking note of the version displayed in the `Kernel` field.
2. In the Uyuni Web UI, select the client from **Systems › Overview**, navigate to the **Software › Manage › Channels** tab, and select `dev-sles15-sp{sp-vert}-updates-x86_64`. Navigate to the `Patches` tab, and click **`[List/Remove Patches]`**.
3. In the search bar, type `kernel` and identify the kernel version that matches the kernel currently used by your client.
4. Remove all kernel versions that are newer than the currently installed kernel.

Your channel is now set up for Live Patching, and can be promoted to `testing`. In this procedure, you will also add the Live Patching child channels to your client, ready to be applied.

Procedure: Promoting Live Patching Channels

1. At the command prompt on the client, as `root`, promote and clone the `dev-sles15-{sp-vert}-pool-x86_64` channel to a new testing channel:

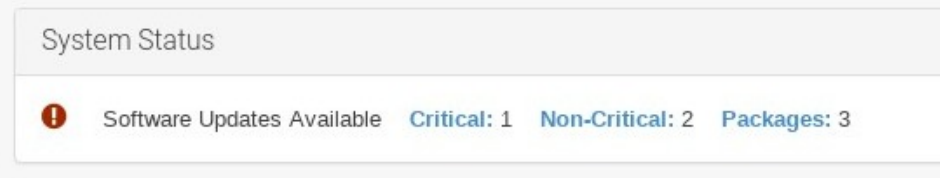
```
# spacewalk-manage-channel-lifecycle --promote -c dev-sles15-{sp-vert}-pool-x86_64
```

2. In the Uyuni Web UI, select the client from **Systems › Overview**, and navigate to the **Software › Software Channels** tab.
3. Check the new `test-sles15-sp3-pool-x86_64` custom channel to change the base channel, and check both corresponding Live Patching child channels.
4. Click **[Next]**, confirm that the details are correct, and click **[Confirm]** to save the changes.

You can now select and view available CVE patches, and apply these important kernel updates with Live Patching.

Procedure: Applying Live Patches to a Kernel

1. In the Uyuni Web UI, select the client from **Systems › Overview**. You will see a banner at the top of the screen showing the number of critical and non-critical packages available for the client:



2. Click **[Critical]** to see a list of the available critical patches.
3. Select any patch with a synopsis reading `Important: Security update for the Linux kernel`. Security bugs will also include their CVE number, where applicable.
4. OPTIONAL: If you know the CVE number of a patch you want to apply, you can search for it in **Audit › CVE Audit**, and apply the patch to any clients that require it.

Not all kernel patches are Live Patches! Non-Live kernel patches are represented by a `Reboot Required` icon located next to the `Security` shield icon. These patches will always require a reboot.

Not all security issues can be fixed by applying a live patch. Some security issues can only be fixed by applying a full kernel update and will require a reboot. The assigned CVE numbers for these issues are not included in live patches. A CVE audit will display this requirement.

Live Patching on SLES 12

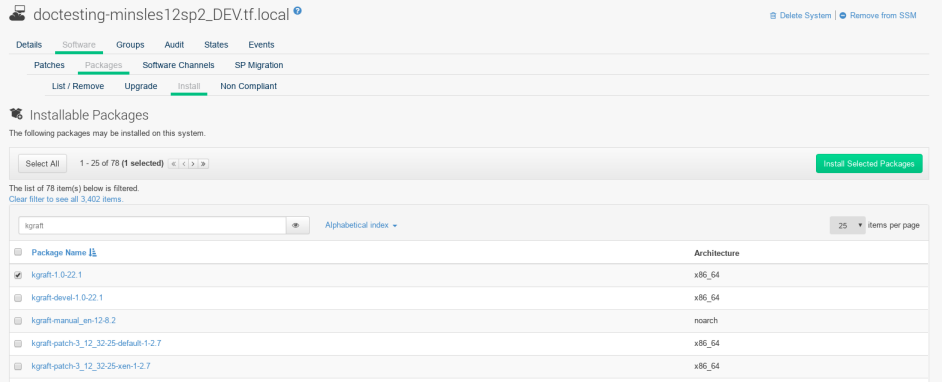
On SLES 12 systems, live patching is managed by kGraft. For in depth information covering kGraft use, see https://www.suse.com/documentation/sles-12/singlehtml/book_sle_admin/book_sle_admin.html#cha.kgraft.

Before you begin, ensure:

- Uyuni is fully updated
- You have one or more Salt clients running SLES 12 (SP1 or later)
- Your SLES 12 Salt clients are registered with Uyuni
- You have access to the SLES 12 channels appropriate for your architecture, including the Live Patching child channel (or channels)
- The clients are fully synchronized

Procedure: Setting up for Live Patching

1. Select the client you want to manage with Live Patching from **Systems** › **Overview**, and on the system details page navigate to the **Software** › **Packages** › **Install** tab. Search for the `kgraft` package, and install it.



2. Apply the highstate to enable Live Patching, and reboot the client.
3. Repeat for each client that you want to manage with Live Patching.
4. To check that Live Patching has been enabled correctly, select the client from **Systems** › **System List**, and ensure that `Live Patching` appears in the `Kernel` field.

When you have the Live Patching channel installed on the client, you can clone the default vendor channel. This cloned channel will be used to manage Live Patching on your clients.

Cloned vendor channels should be prefixed by `dev` for development, `testing`, or `prod` for production. In this procedure, you will create a `dev` cloned channel, and later, you will need to promote the channel to `testing`.

Procedure: Cloning Live Patching Channels

1. At the command prompt on the client, as `root`, obtain the current package channel tree:

```
# spacewalk-manage-channel-lifecycle --list-channels
Spacewalk Username: admin
Spacewalk Password:
Channel tree:

 1. sles12-sp4-pool-x86_64
      \__ sle-live-patching12-pool-x86_64-sp4
      \__ sle-live-patching12-updates-x86_64-sp4
      \__ sle-manager-tools12-pool-x86_64-sp4
      \__ sle-manager-tools12-updates-x86_64-sp4
      \__ sles12-sp4-updates-x86_64
```

2. Use the `spacewalk-manage-channel` command with the `init` argument to automatically create a new development clone of the original vendor channel:

```
spacewalk-manage-channel-lifecycle --init -c sles12-sp4-pool-x86_64
```

3. Check that `dev-sles12-sp4-updates-x86_64` is available in your channel list.

Check the `dev` cloned channel you created, and remove any kernel updates that require a reboot.

Procedure: Removing Non-Live Kernel Patches from Cloned Channels

1. Check the current kernel version by selecting the client from **Systems › System List**, and taking note of the version displayed in the `Kernel` field.
2. In the Uyuni Web UI, select the client from **Systems › Overview**, navigate to the **Software › Manage › Channels** tab, and select `dev-sles12-sp4-updates-x86_64`. Navigate to the `Patches` tab, and click **`[List/Remove Patches]`**.
3. In the search bar, type `kernel` and identify the kernel version that matches the kernel currently used by your client.
4. Remove all kernel versions that are newer than the currently installed kernel.

Your channel is now set up for Live Patching, and can be promoted to `testing`. In this procedure, you will also add the Live Patching child channels to your client, ready to be applied.

Procedure: Promoting Live Patching Channels

1. At the command prompt on the client, as `root`, promote and clone the `dev-sles12-sp4-pool-x86_64` channel to a new testing channel:

```
# spacewalk-manage-channel-lifecycle --promote -c dev-sles12-sp4-pool-x86_64
```

2. In the Uyuni Web UI, select the client from **Systems › Overview**, and navigate to the **Software › Software Channels** tab.
3. Check the new `test-sles12-sp4-pool-x86_64` custom channel to change the base channel,

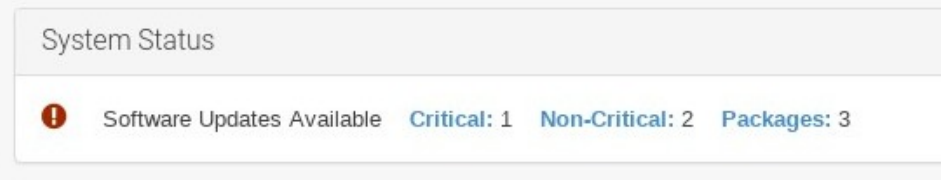
and check both corresponding Live Patching child channels.

4. Click **[Next]**, confirm that the details are correct, and click **[Confirm]** to save the changes.

You can now select and view available CVE patches, and apply these important kernel updates with Live Patching.

Procedure: Applying Live Patches to a Kernel

1. In the Uyuni Web UI, select the client from **Systems › Overview**. You will see a banner at the top of the screen showing the number of critical and non-critical packages available for the client:



2. Click **[Critical]** to see a list of the available critical patches.
3. Select any patch with a synopsis reading `Important: Security update for the Linux kernel`. Security bugs will also include their CVE number, where applicable.
4. OPTIONAL: If you know the CVE number of a patch you want to apply, you can search for it in **Audit › CVE Audit**, and apply the patch to any clients that require it.

Not all kernel patches are Live Patches! Non-Live kernel patches are represented by a `Reboot Required` icon located next to the `Security` shield icon. These patches will always require a reboot.

Not all security issues can be fixed by applying a live patch. Some security issues can only be fixed by applying a full kernel update and will require a reboot. The assigned CVE numbers for these issues are not included in live patches. A CVE audit will display this requirement.

Monitoring with Prometheus

You can monitor your Uyuni environment using Prometheus and Grafana. Uyuni Server and Proxy are able to provide self-health metrics, or install and manage a limited number of Prometheus exporters on Salt clients.

Prometheus is a monitoring tool that is used to record real-time metrics in a time-series database. It is an open-source software project, written in Go. Metrics are collected using HTTP pulls, allowing for higher performance and scalability. For more information about Prometheus, see https://prometheus.io/docs/.

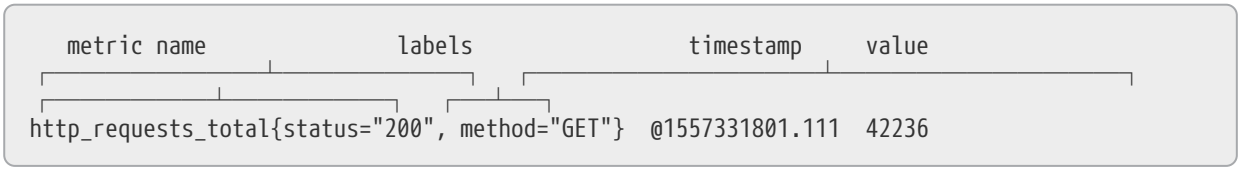
Grafana is a tool for data visualization, monitoring, and analysis. It is used to create dashboards with panels representing specific metrics over a set period of time. Grafana is commonly used together with Prometheus, but also supports other data sources such as ElasticSearch, MySQL, PostgreSQL, and Influx DB. For more information about Grafana, see https://grafana.com/docs/.

You need to install Prometheus and Grafana on a machine separate from the Uyuni Server. We recommend you use a managed Salt client as your monitoring server.

Prometheus and Grafana packages are included in the SUSE Manager Client Tools for SUSE Linux Enterprise 12 and SUSE Linux Enterprise 15.

Prometheus Metrics

Prometheus metrics are time series data, or timestamped values belonging to the same group or dimension. A metric is uniquely identified by its name and set of labels.



Each application or system being monitored must expose metrics in the format above, either through code instrumentation or Prometheus exporters.

The different metric types are:

- Counter - cumulative values. For example, number of errors
- Gauge - can go up or down. For example, temperature
- Histogram - count observations in buckets
- Summary - similar to histogram, but provides totals (sum and count)

For more information about metric types, see https://prometheus.io/docs/concepts/metric_types/.

PromQL

Prometheus has its own query language called PromQL, which is a functional expression language. PromQL allows you to filter multi-dimensional time series data. It is used in all Prometheus interactions.

In PromQL, an expression can evaluate to one of three types:

- Instant vector: a set of time series containing a single sample for each time series, all sharing the same timestamp
- Range vector: a set of time series containing a range of data points over time for each time series
- Scalar: a numeric floating point value

The core part of any PromQL query is the metric name, for example, **http_requests_total**. Labels can be used as optional selectors. This example returns the total number of HTTP requests that have status **200** and method **GET**:

```
http_requests_total{status="200", method="GET"}
```

For more information about PromQL, see the official Prometheus documentation at https://prometheus.io/docs/prometheus/latest/querying/basics/.

Exporters

Exporters are libraries that help with exporting metrics from third-party systems as Prometheus metrics. Exporters are useful whenever it is not feasible to instrument a given application or system with Prometheus metrics directly. Multiple exporters can run on a monitored host to export local metrics.

The Prometheus community provides a list of official exporters, and more can be found as community contributions. For detailed information and an extensive list of exporters, see https://prometheus.io/docs/instrumenting/exporters/.

With Uyuni 4, you can set up the Server and Proxy to expose Prometheus metrics to provide insights about self-health of Uyuni. Metrics are available for these services:

- Hardware and Operating System
- Java Virtual Machines
- Apache
- Squid
- PostgreSQL
- Uyuni internals

The self-health metrics are made available by Uyuni Java application combined with Prometheus standalone exporters, running as systemd daemons.

Uyuni requires these packages to be installed on the Server and the Proxy. The packages are shipped with Uyuni Server and Proxy, but their respective systemd daemons are disabled by default.

These exporter packages are shipped with Uyuni Server:

- Node exporter: `golang-github-prometheus-node_exporter`. See https://github.com/prometheus/node_exporter.
- PostgreSQL exporter: `golang-github-wrouesnel-postgres_exporter`. See https://github.com/wrouesnel/postgres_exporter.
- JMX exporter: `prometheus-jmx_exporter`. See https://github.com/prometheus/jmx_exporter.
- Apache exporter: `golang-github-lusitaniae-apache_exporter`. See https://github.com/Lusitaniae/apache_exporter.

These exporter packages are shipped with Uyuni Proxy:

- Node exporter: `golang-github-prometheus-node_exporter`. See https://github.com/prometheus/node_exporter.
- Squid exporter: `golang-github-boynux-squid_exporter`. See https://github.com/boynux/squid-exporter.

Install and Configure Prometheus

Prometheus is installed on your monitoring server from a package, and needs configuration before you can use it to gather metrics. We recommend you use a managed Salt client as your monitoring server.

Installing Prometheus

If your monitoring server is a Uyuni client, you can install the Prometheus package using the Uyuni Web UI. Otherwise you can download and install the package on your monitoring server manually.

Procedure: Installing Prometheus

1. On the monitoring server, install the `golang-github-prometheus-prometheus` package:

```
zypper in golang-github-prometheus-prometheus
```

2. Enable the Prometheus service:

```
systemctl enable --now prometheus
```

3. Check that the Prometheus interface is loading correctly. In your browser, navigate to the URL of the server where Prometheus is installed, and listen on port 9090 (for example, `http://example.com:9090`).

Configuring Prometheus

Prometheus requires some configuration to collect metrics and set up alarms, or to display metrics graphically in Grafana. You can configure Prometheus in the static configuration file at `/etc/prometheus/prometheus.yml`. It is important to understand how this file is structured. For example:

```
yaml
- job_name: 'suse-manager-server'
  static_configs:
    - targets:
      - 'suse-manager.local:9100'  # Node exporter
      - 'suse-manager.local:9187'  # PostgreSQL exporter
      - 'suse-manager.local:5556'  # JMX exporter (Tomcat)
      - 'suse-manager.local:5557'  # JMX exporter (Taskomatic)
      - 'suse-manager.local:9800'  # Taskomatic
    - targets:
      - 'suse-manager.local:80'    # Message queue
      labels:
        __metrics_path__: /rhn/metrics
```

For more information about configuring Prometheus, see the official Prometheus documentation at https://prometheus.io/docs/prometheus/latest/configuration/configuration/

Monitoring Salt Clients

Prometheus metrics exporters can also be used on Salt clients. The packages are available from the Uyuni client tools channels, and can be enabled and configured directly in the Uyuni Web UI. Currently, two exporters are supported:

- Node exporter: `golang-github-prometheus-node_exporter`. See https://github.com/prometheus/node_exporter.
- PostgreSQL exporter: `golang-github-wrouesnel-postgres_exporter`. See https://github.com/wrouesnel/postgres_exporter.

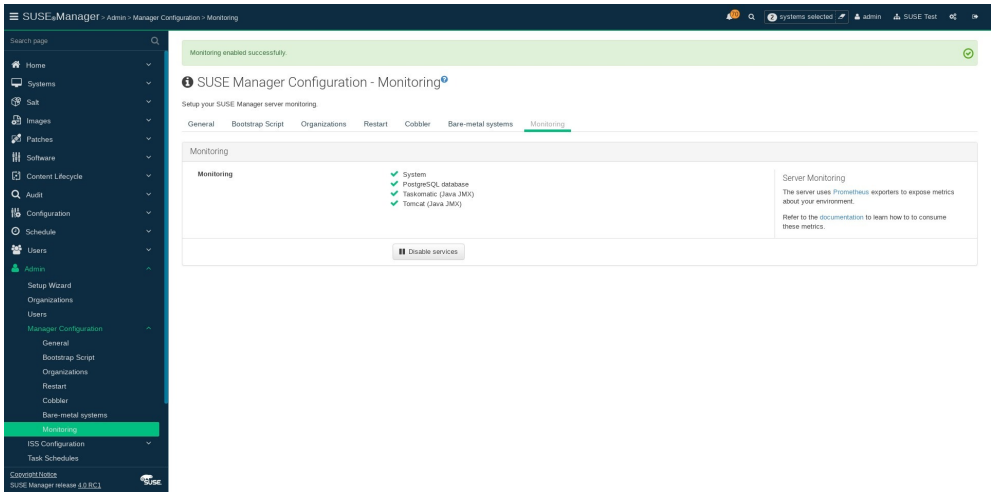
Installing and configuring exporters is done using a Salt formula.

When you have the exporters installed and configured, you can begin using Prometheus to scrape metrics from monitored systems. You can do this directly through the Uyuni Web UI, or set up service discovery. Service discovery instructs Prometheus to automatically scrape metrics from systems as they are enabled.

Enable and Configure Monitoring

Procedure: Enabling Self Monitoring for Uyuni

1. In the Uyuni Web UI, navigate to **Admin › Manager Configuration › Monitoring**.
2. Click **`[Enable services]`**.

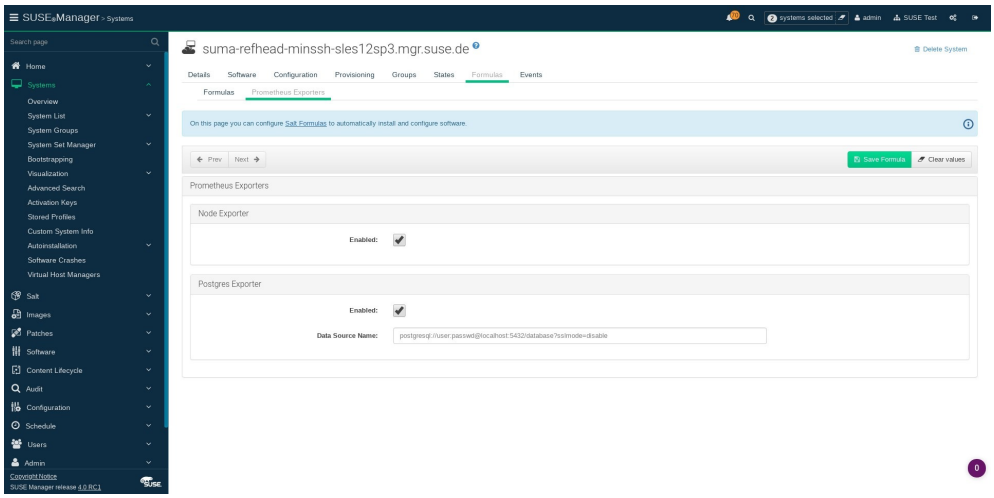


Procedure: Configuring Monitoring Formulas

1. In the SUSE Manager Web UI, open the details page of the system to be monitored, and navigate to the `Formulas` tab.
2. Check the `Monitoring` checkbox to select all monitoring formulas, and click **`[Save]`**.
3. Apply the highstate.

Procedure: Configuring the Exporters

1. In the SUSE Manager Web UI, open the details page of the system to be monitored, and navigate to the **Formulas › Prometheus Exporters** tab.
2. Check the `Enabled` checkbox for both the Node and the Postgres Exporter.
3. In the `Postgres Exporter` section, in the `Data Source Namer` field, enter the path to your data source (for example, `postgresql://user:passwd@localhost:5432/database?sslmode=disable`).
4. Click **`[Save Formula]`**.
5. Apply the highstate.



Procedure: Enable Service Discovery

This feature is a technical preview available in Uyuni 4.0.2 and later. It should not be used in production systems.

1. On the monitoring server, open the Prometheus static configuration file `/etc/prometheus/prometheus.yml`.
2. Add or update the scrape configurations section:

```
job_name: 'suma'
uyuni_sd_configs:
host: "http://your-suse-manager-server-url"
username: "apiuser"
password: "password"
```

3. Save the configuration file and restart the Prometheus service:

```
systemctl restart prometheus
```

Visualization with Grafana

The Grafana website contains dozens of dashboards uploaded by the community. For an example of the Uyuni dashboard, see https://grafana.com/dashboards/10277. For more information about dashboards, see https://grafana.com/dashboards

To use Grafana with Uyuni, you must enable metrics in the Uyuni Web UI and configure your Prometheus instance to collect those metrics.

If your monitoring server is a Uyuni client, you can install the Grafana package using the Uyuni Web UI. Otherwise you can download and install the package on your monitoring server manually.

Procedure: Setting up Grafana

1. Install the `grafana` package:

```
zypper in grafana
```

2. Enable the Grafana service:

```
systemctl enable --now grafana-server
```

3. Navigate to port 3000 in your browser.

Grafana settings are configured in `/etc/grafana/grafana.ini`.

Kubernetes

Prerequisites

The prerequisites listed below should be met before proceeding.

- At least one *Kubernetes* or _SUSE CaaS Platform _ cluster available on your network
- Uyuni configured for container management

Required channels are present, a registered container build host available, etc.

- virtual-host-gatherer-Kubernetes package installed on your Uyuni server

Requirements

- Kubernetes version 1.5.0 or higher. Alternatively use SUSE CaaS Platform *(SUSE CaaS Platform includes Kubernetes 1.5.0 by default)*
- Docker version 1.12 or higher on the container build host

To enable all Kubernetes related features within the Web UI, the virtual-host-gatherer-Kubernetes package must be installed.

Register Kubernetes as a Virtual Host Manager

Kubernetes clusters are registered with SUSE Manager as `virtual host managers`. Registration and authorization begins with importing a `kubeconfig` file using Kubernetes official command line tool `kubectl`.

Procedure: Registering a Kubernetes Cluster with Uyuni

1. In the Uyuni Web UI, navigate to **Systems › Virtual Host Managers**.
2. In the `Create` menu, select `Kubernetes Cluster`.
3. Enter a label for the new virtual host manager.
4. Select the `kubeconfig` file that contains the required data for the Kubernetes cluster.
5. Select the correct *context* for the cluster, as specified in the kubeconfig file.
6. Click **`[Create]`**.

View the List of Nodes in a Cluster

1. Select **Systems › Virtual Host Managers** from the navigation menu.

2. Select the Kubernetes cluster to view it.
3. Node data is not refreshed during registration. To refresh node data, click **`[Schedule refresh data]`**.
4. Refresh the browser. If the node data is not available wait a few moments, and try again.

Obtain Runtime Data about Images

See the following steps to find runtime data for images.

1. In the Uyuni Web UI, navigate to **Images › Image List**.
2. In the image list table, the runtime columns are labeled `Revision`, `Runtime`, and `Instances`. In these columns find the following information:
    - `Revision`: An artificial sequence number that increments on every rebuild for manager-built images or on every re-import for externally built images.
    - `Runtime`: Overall status of the running instances of the image throughout the registered clusters. The status can be one of the following:
        - *All instances are consistent with SUSE Manager*: All the running instances are running the same build of the image as tracked by Uyuni.
        - *Outdated instances found*: Some of the instances are running an older build of the image. A redeploy of the image into the pod may be required.
        - *No information*: The checksum of the instance image does not match the image data contained in Uyuni. A redeploy of the image into the pod may be required.
    - `Instances`: Number of instances running this image across all the clusters registered in Uyuni. A breakdown of numbers can be seen by clicking the pop-up icon next to the number.

Build an image for deployment in Kubernetes

The following steps will help you build an image for deployment in Kubernetes.

1. Under **Images › Stores**, create an image store.
2. In **Images › Profiles**, create an image profile (with a dockerfile that is suitable to deploy to Kubernetes).
3. Under **Images › Build**, build an image with the new profile and wait for the build to finish.
4. Deploy the image into one of the registered Kubernetes clusters using `kubectl`.

In the `Runtime` and `Instances` columns in the respective image row you can now see the updated data.

Import a Previously Deployed Image in Kubernetes

The following steps will guide you through importing a previously deployed image in Kubernetes.

1. Select an image that has already been deployed to any of your registered Kubernetes clusters.
2. Add the registry owning the image to SUSE Manager as an image store.
3. Navigate to **Images › Image List**, click `Import` from the top-right corner, fill in the form fields and click `Import`.

In the `Runtime` and `Instances` columns in the respective image row you can now see the updated data.

Obtain Additional Runtime Data

The following steps will help you find additional runtime data.

1. Navigate to **Images › Image List**, locate the row that contains the running instance, and click **`[Details]`** on the right end. Under the `Overview` tab, notice the data in `Runtime` and `Instances` fields under the `Image Info` section.
2. Select the `Runtime` tab.
3. In the `Runtime` tab is a breakdown of the Kubernetes pods running this image in all the registered clusters including the following data:
    - Pod name
    - Namespace which the pod resides in
    - The runtime status of the container in the specific pod. For more about status icons, see the next section.

Rebuild a Previously Deployed Image in Kubernetes

These steps will guide you through rebuilding an image that has been deployed to a Kubernetes cluster.

1. Go to **Images › Image List**. Click the `Details` button on the right end of a row that has running instances. The image must be manager-built.
2. Click the `Rebuild` button located under the `Build Status` section and wait for the build to finish.
3. Notice the change in the `Runtime` icon and title, reflecting the fact that now the instances are running a previous build of the image.

Role Based Access Control Permissions and Certificate Data

Currently, only kubeconfig files containing all embedded certificate data can be used with Uyuni.

The API calls from Uyuni are:

- `GET /api/v1/pods`

- `GET /api/v1/nodes`

According to this list, the minimum recommended permissions for Uyuni should be as follows:

- A ClusterRole to list all the nodes:

```
resources: ["nodes"]
verbs: ["list"]
```

- A ClusterRole to list pods in all namespaces (role binding must not restrict the namespace):

```
resources: ["pods"]
verbs: ["list"]
```

Due to a a 403 response from `/pods`, the entire cluster will be ignored by Uyuni.

For more information on working with RBAC Authorization, see https://kubernetes.io/docs/admin/authorization/rbac/.

Public Cloud

Some public cloud environments provide images for Uyuni Server and Proxy. This section discusses what you will need for running Uyuni in a public cloud, and how to set up your installation.

Public clouds provide Uyuni under a Bring Your Own Subscription (BYOS) model. This means that you must register them with the SUSE Customer Center. For more information about registering Uyuni with SUSE Customer Center, see [**Installation › General-requirements ›**].

Depending on the public cloud network you are using, you can locate the Uyuni installation images by searching for the keywords `suse`, `manager`, `proxy`, or `BYOS`.

For `SUSE Manager Server in Azure`, see [**Administration › Public-cloud-azure ›**].

Instance Requirements

Select a public cloud instance that meets the hardware requirements in [**Installation › Hardware-requirements ›**].

In addition, be aware of these considerations:

- The Uyuni setup procedure performs a forward-confirmed reverse DNS lookup. This must succeed in order for the setup procedure to complete successfully and for Uyuni to operate as expected. Therefore, it is important to perform hostname and IP configuration prior to running the Uyuni setup procedure.
- Uyuni Server and Proxy instances are expected to run in a network configuration that provides you control over DNS entries, but cannot access the wider internet. Within this network configuration DNS resolution must be provided: `hostname -f` must return the fully-qualified domain name (FQDN). DNS resolution is also important for connecting clients. DNS is dependent on the cloud framework you choose, refer to the cloud service provider documentation for detailed instructions.
- We recommend that you locate software repositories, the server database, and the proxy squid cache on an external virtual disk. This prevents data loss if the instance is unexpectedly terminated. Instructions for setting up an external virtual disk are contained in this section.

Network Setup

On a public cloud service, you must run Uyuni within a restricted network, such as VPC private subnet with an appropriate firewall setting. The instance must only be able to be accessed by machines in your specified IP ranges.

A world-accessible Uyuni instance violates the terms of the Uyuni EULA, and it will not be supported by SUSE.

To access the Uyuni Web UI, allow HTTPS when you set up your networking environment.

Set the Hostname

Uyuni requires a stable and reliable hostname. Changing the hostname at a later point can create errors.

In most public cloud environments, the method shown in this section will work correctly. However, you will have to perform the same modification for every client.

You might prefer to manage DNS resolution by creating a DNS entry in your network environment instead.

You can also manage hostname resolution by editing the `/etc/resolv.conf` file. Depending on the order of your setup, if you start the Uyuni instance prior to setting up DNS services the file may not contain the appropriate `search` directive. Check that the proper search directive exists in `/etc/resolv.conf` and add it if it is missing.

Procedure: Setting the hostname locally

1. Disable hostname setup by editing the DHCP configuration file at `/etc/sysconfig/network/dhcp`, and adding this line:

```
DHCLIENT_SET_HOSTNAME="no"
```

2. Set the hostname locally with the `hostnamectl` command. Ensure you use the system name, not the FQDN. For example, if the FQDN is `system_name.example.com`, the system name is `system_name`, and the domain name is `example.com`.

```
# hostnamectl set-hostname system_name
```

3. Create a DNS entry in your network environment for domain name resolution, or force correct resolution by editing the `/etc/hosts` file. You can find the IP address by checking your public cloud web console, or from the command line:
    - Amazon EC2 instance:

```
# ec2metadata --local-ipv4
```

    - Google Compute Engine:

```
# gcemetadata --query instance --network-interfaces --ip
```

    - Microsoft Azure:

```
# azuremetadata --internal-ip
```

In the following command, replace `<ip_address>` with IP address you retrieve from the command line above:

```
# echo "<ip_address> suma.cloud.net suma" >> /etc/hosts
```

Set up DNS Resolution

You will need to update the DNS records for the instance within the DNS service of your network environment. Refer to the cloud service provider documentation for detailed instructions:

- DNS setup on Amazon EC2
- DNS setup on Google Compute Engine
- DNS setup on Microsoft Azure

If you run a Uyuni Server instance, ensure the external storage is attached and prepared correctly, and that DNS resolution is set up as described. Start the `susemanager_setup` with YaST:

```
# /sbin/yast2 susemanager_setup
```

The Uyuni setup procedure in YaST is designed as a one pass process with no rollback or cleanup capability. Therefore, if the setup procedure is interrupted or ends with an error, it is not recommended that you repeat the setup process or attempts to manually fix the configuration. These methods are likely to result in a faulty Uyuni installation. If you experience errors during setup, start a new instance, and begin the setup procedure again on a clean system.

If you receive a message that there is not enough space available for setup, ensure that your root volume is at least 20 GB and double check that the instructions in Using Separate Storage Volume have been completed correctly.

Prior to registering instances started from on demand images remove the following packages from the instance to be registered: … cloud-regionsrv-client … **For Amazon EC2**

+ regionServiceClientConfigEC2

+ regionServiceCertsEC2 … **For Google Compute Engine**

+ cloud-regionsrv-client-plugin-gce

+ regionServiceClientConfigGCE

+ regionServiceCertsGCE … **For Microsoft Azure**

+ regionServiceClientConfigAzure

+ regionServiceCertsAzure

+ If these packages are not removed it is possible to create interference between the repositories provided by Uyuni and the repositories provided by the SUSE operated update infrastructure.

+ Additionally remove the line from the `/etc/hosts` file that contains the **susecloud.net** reference. ** If you run a Uyuni Proxy instance

+ Launch the instance, optionally with external storage configured. If you use external storage (recommended), prepare it according to Using Separate Storage Volume. It is recommended but not required to prepare the storage before configuring Uyuni proxy, as the suma-storage script will migrate any existing cached data to the external storage. After preparing the instance, register the system with the parent SUSE Manager, which could be a Uyuni Server or another Uyuni Proxy. See the **[Installation › Proxy-setup ›]** for details. When registered, run

+

```
$ /usr/sbin/configure-proxy.sh
```

+ to configure your Uyuni Proxy instance. . After the completion of the configuration step, Uyuni should be functional and running. For Uyuni Server, the setup process created an administrator user with following user name:

+ * User name: `admin`

+

Table 1. Account credentials for admin user

Amazon EC2	Google Compute Engine	Microsoft Azure
`Instance-ID`	`Instance-ID`	`Instance-Name`**-suma**

+ The current value for the `Instance-ID` or `Instance-Name` in case of the Azure Cloud, can be obtained from the public cloud Web console or from within a terminal session as follows: ** Obtain instance id from within Amazon EC2 instance

+

```
$ ec2metadata --instance-id
```

- Obtain instance id from within Google Compute Engine instance

  ```
  $ gcemetadata --query instance --id
  ```

- Obtain instance name from within Microsoft Azure instance

```
$ azuremetadata --instance-name
```

After logging in through the Uyuni Server Web UI, **change** the default password.

Uyuni Proxy does not have administration access to the Web UI. It can be managed through its parent Uyuni Server.

Using Separate Storage Volume

We recommend that the repositories and the database for Uyuni be stored on a virtual storage device. This best practice will avoid data loss in cases where the Uyuni instance may need to be terminated. These steps **must** be performed **prior** to running the YaST Uyuni setup procedure.

1. Provision a disk device in the public cloud environment, refer to the cloud service provider documentation for detailed instructions. The size of the disk is dependent on the number of distributions and channels you intend to manage with Uyuni. For sizing information refer to SUSE Manager sizing examples. A rule of thumb is 25 GB per distribution per channel.
2. Once attached the device appears as Unix device node in your instance. For the following command to work this device node name is required. In many cases the attached storage appears as **/dev/sdb**. In order to check which disk devices exists on your system, call the following command:

   ```
   $ hwinfo --disk | grep -E "Device File:"
   ```

3. With the device name at hand the process of re-linking the directories in the filesystem Uyuni uses to store data is handled by the suma-storage script. In the following example we use `/dev/sdb` as the device name.

   ```
   $ /usr/bin/suma-storage /dev/sdb
   ```

   After the call all database and repository files used by SUSE Manager Server are moved to the newly created xfs based storage. In case your instance is a Uyuni Proxy, the script will move the Squid cache, which caches the software packages, to the newly created storage. The xfs partition is mounted below the path `/manager_storage`. .

4. Create an entry in /etc/fstab (optional)

   Different cloud frameworks treat the attachment of external storage devices differently at instance boot time. Please refer to the cloud environment documentation for guidance about the fstab entry.

   If your cloud framework recommends to add an fstab entry, add the following line to the **/etc/fstab** file.

   ```
   /dev/sdb1 /manager_storage xfs defaults,nofail 1 1
   ```

Registration of Cloned Systems

Uyuni cannot distinguish between different instances that use the same system ID. If you register a second instance with the same system ID as a previous instance, Uyuni will overwrite the original system data with the new system data. This can occur when you launch multiple instances from the same image, or when an image is created from a running instance. However, it is possible to clone systems and register them successfully by deleting the cloned system's ID, and generating a new ID.

Procedure: Registering Cloned Systems

1. Clone the system using your preferred hypervisor's cloning mechanism.
2. On the cloned system, change the hostname and IP addresses, and check the `/etc/hosts` file to ensure you have the right host entries.
3. On traditional clients, stop the `rhnsd` daemon with `/etc/init.d/rhnsd stop` or, on newer systemd-based systems, with `service rhnsd stop`. Then `service osad stop`.
4. For SLES 11 or Red Hat Enterprise Linux 5 or 6 clients, run these commands:

```
# rm /var/lib/dbus/machine-id
# dbus-uuidgen --ensure
```

5. For SLES 12 or Red Hat Enterprise Linux 7 clients, run these commands:

```
# rm /etc/machine-id
# rm /var/lib/dbus/machine-id
# dbus-uuidgen --ensure
# systemd-machine-id-setup
```

6. If you are using Salt, then you will also need to run these commands:

```
# service salt-minion stop
# rm -rf /var/cache/salt
```

7. If you are using a traditional client, clean up the working files with:

```
# rm -f /etc/sysconfig/rhn/{osad-auth.conf,systemid}
```

The bootstrap should now run with a new system ID, rather than a duplicate.

If you are onboarding Salt client clones, then you will also need to check if they have the same Salt minion ID. You will need to delete the minion ID on each cloned client, using the `rm` command. Each operating system type stores this file in a slightly different location, check the table for the appropriate command.

Minion ID File Location

Each operating system stores the minion ID file in a slightly different location, check the table for the appropriate command.

Operating System	Commands

SLES 12	`rm /etc/salt/minion_id`
	
	`rm -f /etc/zypp/credentials.d/{SCCcredentials,NCCcredentials}`

SLES 11	`rm /etc/salt/minion_id`
	
	`suse_register -E`

SLES 10	`rm -rf /etc/{zmd,zypp}`
	
	`rm -rf /var/lib/zypp/` Do not delete `/var/lib/zypp/db/products/`
	
	`rm -rf /var/lib/zmd/`

Red Hat Enterprise Linux 5, 6, 7	`rm -f /etc/NCCcredentials`

When you have deleted the minion ID file, re-run the bootstrap script, and restart the client to see the cloned system in Uyuni with the new ID.

Inter-Server Synchronization

If you have more than one Uyuni installation, you will probably want to ensure that they stay aligned on content and permissions. Inter-Server Synchronization (ISS) allows you to connect two or more Uyuni servers and keep them up-to-date.

To set up ISS, you need to define one Uyuni server as a master, with the other as a slave. If conflicting configurations exist, the system will prioritize the master configuration.

Procedure: Setting up an ISS Master

1. In the Uyuni Web UI, navigate to **Admin › ISS Configuration › Slave Setup**, and click **[Add new master]**.
2. In the Details for new Master dialog, provide these details for the server to use as the ISS master:
    - In the Master Fully-Qualified Domain Name field, enter the FQDN of the ISS master (for example: http://server1.example.com).
    - In the Filename of this Master's CA Certificate field, enter the absolute path to the CA certificate on the ISS master (for example: /etc/pki/trust/anchors-org-ssl). Click **[Add new master]** to add the ISS master.

Procedure: Setting up an ISS Slave

1. In the Uyuni Web UI, navigate to **Admin › ISS Configuration › Master Setup**, and click **[Add new slave]**.
2. In the Edit Slave Details dialog, for the server to use as the ISS slave provide these details:
    - In the Slave Fully-Qualified Domain Name field, enter the FQDN of the ISS slave (for example: http://server2.example.com).
    - Check the Allow Slave to Sync? checkbox to enable the slave to synchronize with the master.
    - Check the Sync All Orgs to Slave? checkbox to synchronize all organizations to this slave.
3. Click **[Create]** to add the ISS slave.
4. In the Allow Export of the Selected Organizations section, check the organizations you want to allow this slave to export to the master, and click **[Allow Orgs]**.

When you have the master and slaves set up, you can perform a synchronization from the command line on the slave, with this command:

```
mgr-inter-sync
```

Set up a Client to Master Validation Fingerprint

In highly secure network configurations you may wish to ensure your Salt clients are connecting a specific master. To set up validation from client to master enter the master's fingerprint within the `/etc/salt/minion` configuration file.

See the following procedure:

1. On the master enter the following command as root and note the fingerprint:

   ```
   salt-key -F master
   ```

   On your client, open the `/etc/salt/minion` configuration file. Uncomment the following line and enter the master's fingerprint replacing the example fingerprint:

   ```
   master_finger: 'ba:30:65:2a:d6:9e:20:4f:d8:b2:f3:a7:d4:65:11:13'
   ```

2. Restart the salt-minion service:

   ```
   # systemctl restart salt-minion
   ```

For information on configuring security from a client, see https://docs.saltstack.com/en/latest/ref/configuration/minion.html.

Signing Repository Metadata

TODO

Explain why repository metadata should/would be signed.

You will require a custom GPG key to be able to sign repository metadata.

Procedure: Generating a Custom GPG Key

1. As the root user, use the `gpg` command to generate a new key:

```
gpg --gen-key
```

2. At the prompts, select `RSA` as the key type, with a size of 2048 bits, and select an appropriate expiry date for your key. Check the details for your new key, and type `y` to confirm.
3. At the prompts, enter a name and email address to be associated with your key. You can also add a comment to help you identify the key, if desired. When you are happy with the user identity, type `O` to confirm.
4. At the prompt, enter a passphrase to protect your key.
5. The key should be automatically added to your keyring. You can check by listing the keys in your keyring:

```
gpg --list-keys
```

6. Add the password for your keyring to the `/etc/rhn/signing.conf` configuration file, by opening the file in your text editor and adding this line:

```
GPGPASS="password"
```

You can manage metadata signing on the command line using the `mgr-sign-metadata-ctl` command.

Procedure: Enabling Metadata Signing

1. You will need to know the short identifier for the key to use. You can list your available public keys in short format:

```
gpg --keyid-format short --list-keys
...
pub   rsa2048/3E7BFE0A 2019-04-02 [SC] [expires: 2021-04-01]
      A43F9EC645ED838ED3014B035CFA51BF3E7BFE0A
uid         [ultimate] SUSE Manager
sub   rsa2048/118DE7FF 2019-04-02 [E] [expires: 2021-04-01]
```

2. Enable metadata signing with the `mgr-sign-metadata-ctl` command:

```
mgr-sign-metadata-ctl enable 3E7BFE0A
OK. Found key 3E7BFE0A in keyring.
DONE. Set key 3E7BFE0A in /etc/rhn/signing.conf.
DONE. Enabled metadata signing in /etc/rhn/rhn.conf.
DONE. Exported key 4E2C3DD8 to /srv/susemanager/salt/gpg/mgr-keyring.gpg.
DONE. Exported key 4E2C3DD8 to /srv/www/htdocs/pub/mgr-gpg-pub.key.
NOTE. For the changes to become effective run:
   mgr-sign-metadata-ctl regen-metadata
```

3. You can check that your configuration is correct with this command:

```
mgr-sign-metadata-ctl check-config
```

4. Restart the services and schedule metadata regeneration to pick up the changes:

```
mgr-sign-metadata-ctl regen-metadata
```

You can also use the `mgr-sign-metadata-ctl` command to perform other tasks. Use `mgr-sign-metadata-ctl --help` to see the complete list.

Repository metadata signing is a global option. When it is enabled, it is enabled on all software channels on the server. This means that all clients connected to the server will need to trust the new GPG key to be able to install or update packages.

Procedure: Importing GPG keys on Clients

1. For RPM-based client systems, use these remote commands:

```
rpm --import http://server.example.com/pub/keyname.key
rpm --import http://server.example.com/pub/company.key
```

2. For Ubuntu clients, you will need to reassign the channels, which will automatically pick up the new GPG key. You can do this through the Uyuni Web UI, or from the command line on the server with this command:

```
salt <ubuntu-client> state.apply channels
```

3. OPTIONAL: For Salt clients, you might prefer to use a state to manage your GPG keys.

Mirror Source Packages

If you build your own packages locally, or if you require the source code for your packages for legal reasons, it is possible to mirror the source packages on Uyuni Server.

Mirroring source packages can consume a significant amount of disk space.

Procedure: Mirroring Source Packages

1. Open the `/etc/rhn/rhn.conf` configuration file, and add this line:

```
server.sync_source_packages = 1
```

2. Restart the Spacewalk service to pick up the changes:

```
spacewalk-service restart
```

Currently, this feature can only be enabled globally for all repositories. It is not possible to select individual repositories for mirroring.

When this feature has been activated, the source packages will become available in the Uyuni Web UI. They will be shown as sources for the binary package, and can be downloaded directly from the Web UI. Source packages cannot be installed on clients using the Web UI.

Authentication Methods

Authentication Via PAM

Uyuni supports network-based authentication systems via Pluggable Authentication Modules (PAM). PAM is a suite of libraries that allows you to integrate Uyuni with a centralized authentication mechanism, eliminating the need to remember multiple passwords. Uyuni supports LDAP, Kerberos, and other network-based authentication systems via PAM.

Enable PAM

1. Create a PAM service file. PAM service files are located in `/etc/pam.d/susemanager` by default. Enforce its use by adding this line to `/etc/rhn/rhn.conf`:

```
pam_auth_service = susemanager
```

This assumes the PAM service file is named `susemanager`.

2. In the Uyuni Web UI, navigate to **Create User** and enable a new or existing user to authenticate with PAM
3. Check the `Pluggable Authentication Modules (PAM)` checkbox. It is below the password and password confirmation fields.
4. To authenticate a SLES system against Kerberos, add these lines to `/etc/pam.d/susemanager`:

```
#%PAM-1.0
auth     include        common-auth
account  include        common-account
password include        common-password
session  include        common-session
```

To authenticate a Red Hat Enterprise Linux system against Kerberos, add these lines to `/etc/pam.d/susemanager`:

```
#%PAM-1.0
auth        required      pam_env.so
auth        sufficient    pam_krb5.so no_user_check
auth        required      pam_deny.so
account     required      pam_krb5.so no_user_check
```

+

YaST can be used to configure PAM. You will need to install the `yast2-ldap-client` and `yast2-kerberos-client` packages. For more information about configuring PAM, see the SUSE Linux Enterprise Server Security Guide https://www.suse.com/documentation/sles-15/book_security/data/

part_auth.html. This is a generic example that will also work for other network-based authentication methods.

+ .Changing Passwords IMPORTANT: Changing the password on the Uyuni Web UI changes only the local password on the Uyuni Server. If PAM is enabled for that user, the password might not be used at all. In the above example, for instance, the Kerberos password will not be changed.

Authentication with eDirectory and PAM

1. First check to ensure eDirectory authentication is working with your current OS for example:

```
getent passwd
```

2. If users are returned from eDirectory then create the `/etc/pam.d/susemanager` and add the following content:

```
#%PAM-1.0
auth     include        common-auth
account  include        common-account
password include        common-password
session  include        common-session
```

3. Add these lines to the Uyuni configuration file at `/etc/rhn/rhn.conf`:

```
pam_auth_service = susemanager
```

You can now create users with the same ID used by eDirectory, and check the `Use PAM` checkbox in the Uyuni Web UI.

Listing 4. Example of Quest VAS Active Directory Authentication Template

```
#%PAM-1.0
auth       required       pam_env.so
auth       sufficient     pam_vas3.so no_user_check
auth       requisite      pam_vas3.so echo_return
auth       required       pam_deny.so
account    required       pam_vas3.so no_user_check
```

Authentication Via Single Sign-On (SSO)

This feature is provided as a technical preview. It is not supported for use in production environments.

Uyuni supports single sign-on (SSO) by implementing the Security Assertion Markup Language (SAML) 2 protocol.

Single sign-on is an authentication process that allows a user to access multiple applications with one set of credentials. SAML is an XML-based standard for exchanging authentication and authorization data. A SAML identity service provider (IdP) provides authentication and authorization services to service providers (SP), such as Uyuni. Uyuni exposes three endpoints which must be enabled for single sign-on.

SSO in Uyuni supports:

- Log in with SSO.
- Log out with service provider-initiated single logout (SLO), and Identity service provider single logout service (SLS).
- Assertion and nameId encryption.
- Assertion signatures.
- Message signatures with AuthNRequest, LogoutRequest, and LogoutResponses.
- Enable an Assertion consumer service endpoint.
- Enable a single logout service endpoint.
- Publish the SP metadata (which can be signed).

SSO in Uyuni does not support:

- Product choosing and implementation for the Identity Service Provider (IdP).
- SAML support for other products (check with the respective product documentation).

Prerequisites

Before you begin, you will need to have configured an external Identity Service Provider with these parameters. Check your IdP documentation for instructions.

You will need these endpoints:

- Assertion Consumer Service (or ACS): an endpoint to accept SAML messages to establish a session into the Service Provider. The endpoint for ACS in Uyuni is: https://example.com/rhn/manager/sso/acs
- Single Logout Service (or SLS): an endpoint to initiate a logout request from the IdP. The endpoint for SLS in Uyuni is: https://example.com/rhn/manager/sso/sls
- Metadata: an endpoint to retrieve Uyuni metadata for SAML. The endpoint for Metadata in Uyuni is: https://example.com/rhn/manager/sso/metadata

Your IdP must have a SAML:Attribute containing the username of the IdP user domain, called `uid`. The `uid` attribute passed in the SAML:Attribute must be created in the Uyuni user base before you activate single sign-on.

After the authentication with the IdP using the user `orgadmin` is successful, you will be logged in into

Uyuni as the `orgadmin` user, provided that the `orgadmin` user exists in Uyuni.

Enable SOO

Using SSO is mutually exclusive with other types of authentication: it is either enabled or disabled. SSO is disabled by default.

Procedure: Enabling SSO

1. If your users do not yet exist in Uyuni, create them first.
2. Edit `/etc/rhn/rhn.conf` and add this line at the end of the file:

```
java.sso = true
```

3. Find the parameters you want to customize in `/usr/share/rhn/config-defaults/rhn_java_sso.conf`. Insert the parameters you want to customize into `/etc/rhn/rhn.conf` and prefix them with `java.sso.`.

   For example, in `/usr/share/rhn/config-defaults/rhn_java_sso.conf` find:

```
onelogin.saml2.sp.assertion_consumer_service.url = https://YOUR-PRODUCT-HOSTNAME-OR-IP/rhn/manager/sso/acs
```

   In order to customize it, create the corresponding option in `/etc/rhn/rhn.conf` by prefixing the option name with `java.sso.`:

```
java.sso.onelogin.saml2.sp.assertion_consumer_service.url = https://YOUR-PRODUCT-HOSTNAME-OR-IP/rhn/manager/sso/acs
```

   To find all the occurrences you need to change, search in the file for the placeholders `YOUR-PRODUCT` and `YOUR-IDP-ENTITY`. Every parameter comes with a brief explanation of what it is meant for.

4. Restart the spacewalk service to pick up the changes:

```
spacewalk-service restart
```

When you visit the Uyuni URL, you will be redirected to the IdP for SSO where you will be requested to authenticate. Upon successful authentication, you will be redirected to the Uyuni Web UI, logged in as the authenticated user. If you encounter problems with logging in using SSO, check the Uyuni logs for more information.

Using a Custom SSL Certificate

The following section will guide you through using a custom certificate with Uyuni 4.0 and SUSE Manager Proxy 4.0.

Prerequisites

The following list provides requirements for using a custom certificate.

- A Certificate Authority (CA) SSL public certificate file
- A Web server SSL private key file
- A Web server SSL public certificate file
- Key and Certificate files must be in PEM format

Hostname and SSL Keys

The hostname of the web server's SSL keys and relevant certificate files must match the hostname of the machine which they will be deployed on.

Intermediate Certificates

In case you want to use CAs with intermediate certificates, merge the intermediate and root CA certificates into one file. It is important that the intermediate certificate comes first within the combined file.

Setup

After completing YaST firstboot procedures, export your current environment variables and point them to the correct SSL files to be imported. Running these commands will make the default certificate obsolete after executing the `yast2 susemanagersetup` command. For more information on YaST firstboot, see https://www.suse.com/documentation/suse-manager-3/singlehtml/suse_manager21/book_susemanager_install/book_susemanager_install.html#sec.manager.inst.setup.

1. Export the environment variables and point to the SSL files to be imported:

```
export CA_CERT=`path_to_CA_certificate_file`export
SERVER_KEY=`path_to_web_server_key`export SERVER_CERT=`path_to_web_server_certificate`
```

2. Execute Uyuni setup with

```
yast2 susemanagersetup
```

   Proceed with the default setup. Upon reaching the Certificate Setup window during YaST installation, fill in random values, as these will be overridden with the values specified in

[bp.cert.custom.setup.proc.export].

Shell Requirements

Make sure that you execute `yast2 susemanagersetup` from within the same shell the environment variables were exported from.

Using a Custom Certificate with SUSE Manager Proxy

After completing the installation with yast found in [advanced.topics.proxy.quickstart] continue with a modified [at.manager.proxy.run.confproxy] procedure:

1. Execute `configure-proxy.sh`.
2. When prompted with:

   ```
   Do you want to import existing certificates?
   ```

   Answer with *y* .
3. Continue by following the script prompts.

Backup and Restore

Back up your Uyuni installation regularly, in order to prevent data loss. Because Uyuni relies on a database as well as the installed program and configurations, it is important to back up all components of your installation. This chapter contains information on the files you need to back up, and introduces the `smdba` tool to manage database backups. It also contains information about restoring from your backups in the case of a system failure.

Backup Space Requirements

Regardless of the backup method you use, you must have available at least three times the amount of space your current installation uses. Running out of space can result in backups failing, so check this often.

Backing up Uyuni

The most comprehensive method for backing up your Uyuni installation is to back up the relevant files and directories. This can save you time in administering your backup, and can be faster to reinstall and re-synchronize in the case of failure. However, this method requires significant disk space and could take a long time to perform the backup.

If you want to only back up the required files and directories, use the following list. To make this process simpler, and more comprehensive, we recommend backing up the entire `/etc` and `/root` directories, not just the ones specified here. Some files only exist if you are actually using the related SUSE Manager feature.

- `/etc/cobbler/`
- `/etc/dhcp.conf`
- `/etc/fstab` and any ISO mountpoints you require.
- `/etc/rhn/`
- `/etc/salt`
- `/etc/sudoers`
- `/etc/sysconfig/rhn/`
- `/root/.gnupg/`
- `/root/.ssh`

  This file exists if you are using an SSH tunnel or SSH `push`. You will also need to have saved a copy of the `id-susemanager` key.

- `/root/ssl-build/`

- /srv/formula_metadata
- /srv/pillar
- /srv/salt
- /srv/susemanager
- /srv/tftpboot/
- /srv/www/cobbler
- /srv/www/htdocs/pub/
- /srv/www/os-images
- /var/cache/rhn
- /var/cache/salt
- /var/lib/cobbler/
- /var/lib/cobbler/templates/ (before version 4.0 it was /var/lib/rhn/kickstarts/)
- /var/lib/Kiwi
- /var/lib/rhn/
- /var/spacewalk/
- Plus any directories containing custom data such as scripts, Kickstart or AutoYaST profiles, and custom RPMs.

You will also need to back up your database, which you can do with the smdba tool. For more information about the smdba tool, see [**Administration** › **Backup-restore** ›].

Procedure: Restore from a Manual Backup

1. Re-install Uyuni. For more information about recovering from a backup, see [**Administration** › **Backup-restore** ›].
2. Re-synchronize your Uyuni repositories with the mgr-sync tool. For more information about the mgr-sync tool, see [syncing.suse.mgr.repositories.scc].
3. You can choose to re-register your product, or skip the registration and SSL certificate generation sections.
4. Re-install the /root/ssl-build/rhn-org-httpd-ssl-key-pair-MACHINE_NAME-VER-REL.noarch.rpm package.
5. Schedule the re-creation of search indexes next time the rhn-search service is started:

```
rhn-search cleanindex
```

This command produces only debug messages. It does not produce error messages.

6. Check whether you need to restore `/var/spacewalk/packages/`. If `/var/spacewalk/packages/` was not in your backup, you will need to restore it. If the source repository is available, you can restore `/var/spacewalk/packages/` with a complete channel synchronization:

```
mgr-sync refresh --refresh-channels
```

Check the progress by running `tail -f /var/log/rhn/reposync/`<CHANNEL_NAME>`.log` as *root*.

Administering the Database with smdba

The `smdba` tool is used for managing a local PostgreSQL database. It allows you to back up and restore your database, and manage backups. It can also be used to check the status of your database, and perform administration tasks, such as restarting.

The `smdba` tool works with local PostgreSQL databases only, it will not work with remotely accessed databases, or Oracle databases.

The `smdba` tool requires `sudo` access, in order to execute system changes. Ensure you have enabled `sudo` access for the `admin` user before you begin, by checking the `/etc/sudoers` file for this line:

```
admin   ALL=(postgres) /usr/bin/smdba
```

Check the runtime status of your database with:

```
smdba db-status
```

This command will return either `online` or `offline`, for example:

```
Checking database core...       online
```

Starting and stopping the database can be performed with:

```
smdba db-start
```

And:

```
smdba db-stop
```

Database Backup with smdba

The `smdba` tool performs a continuous archiving backup. This backup method combines a log of every change made to the database during the current session, with a series of more traditional backup files. When a crash occurs, the database state is first restored from the most recent backup file on disk, then the log of the current session is replayed exactly, to bring the database back to a current state. A continuous archiving backup with `smdba` is performed with the database running, so there is no need for downtime.

This method of backing up is stable and generally creates consistent snapshots, however it can take up a lot of storage space. Ensure you have at least three times the current database size of space available for backups. You can check your current database size by navigating to `/var/lib/pgsql/` and running `df -h`.

The `smdba` tool also manages your archives, keeping only the most recent backup, and the current archive of logs. The log files can only be a maximum file size of 16 MB, so a new log file will be created when the files reach this size. Every time you create a new backup, previous backups will be purged to release disk space. We recommend you use `cron` to schedule your `smdba` backups to ensure that your storage is managed effectively, and you always have a backup ready in case of failure.

Performing a Manual Database Backup

The `smdba` tool can be run directly from the command line. We recommend you run a manual database backup immediately after installation, or if you have made any significant changes to your configuration.

When `smdba` is run for the first time, or if you have changed the location of the backup, it will need to restart your database before performing the archive. This will result in a small amount of downtime. Regular database backups will not require any downtime.

Procedure: Performing a Manual Database Backup

1. Allocate permanent storage space for your backup. This example uses a directory located at `/var/spacewalk/`. This will become a permanent target for your backup, so ensure it will remain accessible by your server at all times.
2. In your backup location, create a directory for the backup:

   ```
   sudo -u postgres mkdir /var/spacewalk/db-backup
   ```

   Or, as root:

   ```
   install -d -o postgres -g postgres -m 700 /var/spacewalk/db-backup
   ```

3. Ensure you have the correct permissions set on the backup location:

```
chown postgres:postgres /var/spacewalk/db-backup
```

4. To create a backup for the first time, run the `smdba backup-hot` command with the `enable` option set. This will create the backup in the specified directory, and, if necessary, restart the database:

```
smdba backup-hot --enable=on --backup-dir=/var/spacewalk/db-backup
```

   This command produces debug messages and finishes sucessfully with the output:

```
INFO: Finished
```

5. Check that the backup files exist in the `/var/spacewalk/db-backup` directory, to ensure that your backup has been successful.

Scheduling Automatic Backups

You do not need to shut down your system in order to perform a database backup with `smdba`. However, because it is a large operation, database performance can slow down while the backup is running. We recommend you schedule regular database backups for a low-traffic period, to minimize disruption.

Ensure you have at least three times the current database size of space available for backups. You can check your current database size by navigating to `/var/lib/pgsql/` and running `df -h`.

Procedure: Scheduling Automatic Backups

1. Create a directory for the backup, and set the appropriate permissions (as root):

```
install -m 700 -o postgres -g postgres /var/spacewalk/db-backup
```

2. Open `/etc/cron.d/db-backup-mgr`, or create it if it does not exist, and add the following line to create the cron job:

```
0 2 * * * root /usr/bin/smdba backup-hot --enable=on --backup-dir=/var/spacewalk/db
-backup
```

3. Check the backup directory regularly to ensure the backups are working as expected.

Restoring from Backup

The `smdba` tool can be used to restore from backup in the case of failure.

Procedure: Restoring from Backup

1. Shut down the database:

```
smdba db-stop
```

2. Start the restore process and wait for it to complete:

```
smdba backup-restore start
```

3. Restart the database:

```
smdba db-start
```

4. Check if there are differences between the RPMs and the database.

```
spacewalk-data-fsck
```

Archive Log Settings

Archive logging allows the database management tool `smdba` to perform hot backups. In Uyuni with an embedded database, archive logging is enabled by default.

PostgreSQL maintains a limited number of archive logs. Using the default configuration, approximately 64 files with a size of 16 MiB are stored.

Creating a user and syncing the channels:

- SLES12-SP2-Pool-x86_64
- SLES12-SP2-Updates-x86_64
- SLE-Manager-Tools12-Pool-x86_64-SP2
- SLE-Manager-Tools12-Updates-x86_64-SP2

PostgreSQL will generate an additional roughly 1 GB of data. So it is important to think about a backup strategy and create a backups in a regular way.

Archive logs are stored at `/var/lib/pgsql/data/pg_xlog/` (postgresql).

Retrieving an Overview of Occupied Database Space

Database administrators may use the subcommand `space-overview` to get a report about occupied table spaces, for example:

```
smdba space-overview
```

outputs:

```
SUSE Manager Database Control. Version 1.5.2
Copyright (c) 2012 by SUSE Linux Products GmbH


Tablespace  | Size (Mb) | Avail (Mb) | Use %
------------+-----------+------------+------
postgres    | 7         | 49168      | 0.013
susemanager | 776       | 48399      | 1.602
```

The `smdba` command is available for PostgreSQL. For a more detailed report, use the `space-tables` subcommand. It lists the table and its size, for example:

```
smdba space-tables
```

outputs:

```
SUSE Manager Database Control. Version 1.5.2
Copyright (c) 2012 by SUSE Linux Products GmbH


Table                                | Size
-------------------------------------+-----------
public.all_primary_keys              | 0 bytes
public.all_tab_columns               | 0 bytes
public.allserverkeywordsincereboot   | 0 bytes
public.dblink_pkey_results           | 0 bytes
public.dual                          | 8192 bytes
public.evr_t                         | 0 bytes
public.log                           | 32 kB
...
```

Moving the Database

It is possible to move the database to another location. For example, move the database if the database storage space is running low. The following procedure will guide you through moving the database to a new location for use by Uyuni.

Procedure: Moving the Database

1. The default storage location for Uyuni is `/var/lib/pgsql/`. If you would like to move it, for example to `/storage/postgres/`, proceed as follows.

2. Stop the running database with (as root):

```
rcpostgresql stop
```

Shut down the running Spacewalk services with:

```
spacewalk-service stop
```

3. Copy the current working directory structure with `cp` using the `-a, --archive` option. For example:

```
cp --archive /var/lib/pgsql/ /storage/postgres/
```

This command will copy the contents of `/var/lib/pgsql/` to `/storage/postgres/pgsql/`.

The contents of the `/var/lib/pgsql` directory needs to remain the same, otherwise the Uyuni database may malfunction. You also should ensure that there is enough available disk space.

4. Mount the new database directory with:

```
mount /storage/postgres/pgsql
```

5. Make sure ownership is `postgres:postgres` and not `root:root` by changing to the new directory and running the following commands:

```
cd /storage/postgres/pgsql/
ls -l
```

Outputs:

```
total 8
drwxr-x---  4 postgres postgres   47 Jun  2 14:35 ./
```

6. Add the new database mount location to your servers fstab by editing `etc/fstab`.
7. Start the database with:

```
rcpostgresql start
```

8. Start the Spacewalk services with:

```
spacewalk-service start
```

Recovering from a Crashed Root Partition

This section provides guidance on restoring your server after its root partition has crashed. This section assumes you have setup your server similar to the procedure explained in Installation guide with separate partitions for the database and for channels mounted at `/var/lib/pgsql` and `/var/spacewalk/`.

Procedure: Recovering from a Crashed Root Partition

1. Install Uyuni. Do not mount the `/var/spacewalk` and `/var/lib/pgsql` partitions. Wait for the installation to complete before going on to the next step.
2. Shut down the services with `spacewalk-service shutdown`.
3. Shut down the database with `rcpostgresql stop`.
4. Mount `/var/spacewalk` and `/var/lib/pgsql` partitions.
5. Restore the directories listed in Backing up Uyuni.
6. Start the Spacewalk services with `spacewalk-services start`.
7. Start the database with `rcpostgresql start`.

Uyuni should now operate normally without loss of your database or synced channels.

Database Connection Information

The information for connecting to the Uyuni database is located in `/etc/rhn/rhn.conf`:

```
db_backend = postgresql
db_user = susemanager
db_password = susemanager
db_name = susemanager
db_host = localhost
db_port = 5432
db_ssl_enabled =
```

Content Lifecycle Management

Content Lifecycle Management allows you to customize and test packages before updating production systems. This is especially useful if you need to apply updates during a limited maintenance window.

Content Lifecycle Management allows you to select software channels as sources, adjust them as required for your environment, and thoroughly test them before installing onto your production systems.

While you cannot directly modify vendor channels, you can clone them and then modify the clones by adding or removing packages and custom patches. You can then assign these cloned channels to test systems to ensure they work as expected and, once all tests pass, apply them to production servers.

This is achieved through a series of environments that your software channels can move through on their lifecycle. Most environment lifecycles include at least test and production environments, but you can have as many environments as you require.

Create a Content Lifecycle Project

1. In the Uyuni Web UI, navigate to **Content Lifecycle › Content Lifecycle Projects**, and click **[Create Project]**.
2. In the `Label` field, enter a label for your project. The `Label` field only accepts lowercase letters, numbers, periods (`.`), hyphens (`-`) and underscores (`_`).
3. In the `Name` field, enter a descriptive name for your project.
4. Click the **[Create]** button to create your project and return to the project page.
5. Click **[Attach/Detach Sources]**.
6. In the `Sources` dialog, select the source type, and select a base channel for your project. The available child channels for the selected base channel are displayed, including information on whether the channel is mandatory or recommended.
7. Check the child channels you require, and click **[Save]** to return to the project page. The software channels you selected should now be showing.
8. Click **[Attach/Detach Filters]**.
9. In the `Filters` dialog, select the filters you want to attach to the project. To create a new filter, click **[Create new Filter]**.
10. Click **[Add Environment]**.
11. In the `Environment Lifecycle` dialog, give the first environment a name and a description, and click **[Save]**. The `Name` field only accepts lowercase letters, numbers, periods (`.`), hyphens (`-`) and underscores (`_`).
12. Continue creating environments until you have all the environments for your lifecycle completed. You can select the order of the environments in the lifecycle by selecting an environment in the `Insert before` field when you create it.

Advanced: Filter Types

Uyuni allows you to create various types of filters to control the content on project build. This is the list of supported filters:

- package filtering
  - by name
  - by name, epoch, version, release and architecture
- patch filtering
  - by advisory name
  - by synopsis
  - by date
  - by affected package
  - by type

No package dependency solving is done on content filtering.

Filter `rule` Parameter

Moreover, each filter has a `rule` parameter set to either `Allow` or `Deny`. The filters are processed: - if a package or patch satisfies a `Deny` filter, it will be excluded from the result, * If a package or patch satisfies an `Allow` filter, it will be included in the result (even if it was excluded by a `Deny` filter).

This behavior is useful when you want to exclude large number of packages or patches using a general `Deny` filter and "cherry-pick" specific packages or patches with specific `Allow` filters.

Content filters are global in your organization and can be shared between projects.

If your project already contains built sources, when you add an environment it will automatically populate with the existing content. Content will be drawn from the previous environment of the cycle if it had one. If there is no previous environment, it will be left empty until the project sources are built again.

Build a Content Lifecycle Project

When you have created your project, defined environments, and attached sources and filters, you can build the project for the first time.

Building applies filters to the attached sources and clones them to the first environment in the project.

Procedure: Building a Content Lifecycle Project

1. In the Uyuni Web UI, navigate to **Content Lifecycle › Content Lifecycle Projects**, and select the project you want to build.
2. Review the attached sources and filters, and click **[Build]**.
3. You can monitor build progress in the `Environment Lifecycle` section.

After the build is finished, the environment version is increased by one and the built sources, such as software channels, can be assigned to your systems.

Promote Environments

When the project has been built, the built sources can be sequentially promoted to the environments.

Procedure: Promoting Environments

1. In the Uyuni Web UI, navigate to **Content Lifecycle › Content Lifecycle Projects**, and select the project you want to work with.
2. In the `Environment Lifecycle` section, locate the environment to promote to its successor, and click **[Promote]**.
3. You can monitor build progress in the `Environment Lifecycle` section.

Assign Systems to Environments

When you build and promote content lifecycle projects, it creates a tree of software channels. To add systems to the environment, assign the base and child software channels to your system using **Software › Software Channels** in the `System Details` page for the system.

Newly added cloned channels are not assigned to systems automatically. If you add or promote sources you will need to manually check and update your channel assignments.

Automatic assignment is intended to be added to Uyuni in a future version.

Tuning Changelogs

Some packages have a long list of changelog entries. This data is downloaded by default, but it is not always useful information to keep. In order to limit the amount of changelog metadata which is downloaded and to save disk space, you can put a limit on how many entries to keep on disk.

This configuration option is in the `/etc/rhn/rhn.conf` configuration file. The parameter defaults to `0`, which means unlimited.

```
java.max_changelog_entries = 0
```

If you set this parameter, it will come into effect only for new packages when they are synchronized. You might like to delete the cached data to remove older data.

Deleting cached data can take a long time. Depending on the number of channels you have and the amount of data to be deleted, it can potentially take several hours. The task is run in the background by Taskomatic, so you can continue to use Uyuni while the operation completes, however you should expect some performance loss.

You can delete cached data from the command line, using the SQL interface:

```
spacewalk-sql -i
SQL> delete from rhnPackageRepodata;
SQL> insert into rhnRepoRegenQueue (id, CHANNEL_LABEL, REASON, FORCE)
(select sequence_nextval('rhn_repo_regen_queue_id_seq'),
        C.label,
        'package summary modification',
        'Y'
   from rhnChannel C);
SQL> \q
```

Maintenance Window Checklist

If you work on a Uyuni Server with scheduled maintenance windows, you might find it difficult to remember all the things that you need to during that critical downtime.

This section contains a checklist for your maintenance window, with links to further information on performing each of the steps, if you need it.

1. Stop services. You will need to stop the spacewalk, SAP, and database services, along with any others you have running.
2. Check if the configuration is still correct.
3. On Salt systems, apply the highstate.
4. Apply the latest updates. See **[Upgrade › Server-upgrade ›]**.

5. Update to the latest service pack, if required. See **[Upgrade › Sp-migration ›]**.
6. Reboot the server.
7. Check if the configuration is still correct.
8. On Salt systems, apply the highstate.
9. Start any stopped services.

Troubleshooting

Troubleshooting

This section contains some common problems you might encounter with Uyuni, and solutions to resolving them.

Before you begin troubleshooting, you might want to produce some reports from your system to help you understand what is going on.

Producing Reports

The `spacewalk-report` command is used to produce a variety of reports for system administrators. These reports can be helpful for taking inventory of your entitlements, subscribed systems, users, and organizations. Using reports is often simpler than gathering information manually from the SUSE Manager Web UI, especially if you have many systems under management.

`spacewalk-reports` *Package*

To use `spacewalk-report`, you must have the `spacewalk-reports` package installed.

`spacewalk-report` allows administrators to organize and display reports about content, systems, and user resources across Uyuni. Using `spacewalk-report`, you can receive reports on:

1. System Inventory: lists all of the systems registered to Uyuni.
2. Entitlements: lists all organizations on Uyuni, sorted by system or channel entitlements.
3. Patches: lists all the patches relevant to the registered systems and sorts patches by severity, as well as the systems that apply to a particular patch.
4. Users: lists all the users registered to Uyuni and any systems associated with a particular user.

To get the report in CSV format, run the following at the command line of your Uyuni Server.

```
spacewalk-report <report_name>
```

The following reports are available:

Table 2. `spacewalk-report` *Reports*

Report	Invoked as	Description
Channel Packages	`channel-packages`	List of packages in a channel.
Channel Report	`channels`	Detailed report of a given channel.
Cloned Channel Report	`cloned-channels`	Detailed report of cloned channels.

Report	Invoked as	Description
Custom Info	`custom-info`	System custom information.
Entitlements	`entitlements`	Lists all organizations on Uyuni with their system or channel entitlements.
Patches in Channels	`errata-channels`	Lists of patches in channels.
Patches Details	`errata-list`	Lists all patches that affect systems registered to Uyuni.
All patches	`errata-list-all`	Complete list of all patches.
Patches for Systems	`errata-systems`	Lists applicable patches and any registered systems that are affected.
Host Guests	`host-guests`	List of host-guests mapping.
Inactive Systems	`inactive-systems`	List of inactive systems.
System Inventory	`inventory`	List of systems registered to the server, together with hardware and software information.
Kickstart Trees	`kickstartable-trees`	List of kickstartable trees.
All Upgradable Versions	`packages-updates-all`	List of all newer package versions that can be upgraded.
Newest Upgradable Version	`packages-updates-newest`	List of only newest package versions that can be upgraded.
Result of SCAP	`scap-scan`	Result of OpenSCAP sccdf eval.
Result of SCAP	`scap-scan-results`	Result of OpenSCAP sccdf eval, in a different format.
System Data	`splice-export`	System data needed for splice integration.
System Groups	`system-groups`	List of system groups.
Activation Keys for System Groups	`system-groups-keys`	List of activation keys for system groups.
Systems in System Groups	`system-groups-systems`	List of systems in system groups.
System Groups Users	`system-groups-users`	Report of system groups users.
Installed Packages	`system-packages-installed`	List of packages installed on systems.
Users in the System	`users`	Lists all users registered to Uyuni.

Report	Invoked as	Description
Systems administered	`users-systems`	List of systems that individual users can administer.

For more information about an individual report, run `spacewalk-report` with the option `--info` or `--list-fields-info` and the report name. The description and list of possible fields in the report will be shown.

For further information on program invocations and options, see the `spacewalk-report(8)` man page as well as the `--help`parameter of the `spacewalk-report`.

Troubleshooting Corrupt Repositories

The information in the repository data file can become corrupt or out of date. This can create problems with updating the server. You can fix this by removing the files and regenerating it. With an new repository data file, updates should operate as expected.

Procedure: Resolving Corrupt Repository Data

1. Remove all files from `/var/cache/rhn/repodata/sles15-{sp-vert}-updates-x86_64`
2. Regenerate the file from the command line:

```
spacecmd softwarechannel_regenerateyumcache sles{sles-version}-{sp-vert}-updates-x86_64
```

Troubleshooting Disk Space

Running out of disk space can have a severe impact on the Uyuni database and file structure which, in most cases, is not recoverable. You can recover disk space by removing unused custom channels and redundant database entries before you run out of space entirely.

For instructions on how to delete custom channels, see **[Administration › Channel-management ›]**.

Procedure: Resolving redundant database entries

1. Use the `spacewalk-data-fsck` command to list any redundant database entries.
2. Use the `spacewalk-data-fsck --remove` command to delete them.

Troubleshooting Local Issuer Certificates

Some older bootstrap scripts create a link to the local certificate in the wrong place. This results in zypper returning an `Unrecognized error` about the local issuer certificate. You can ensure that the link to the local issuer certificate has been created correctly by checking the `/etc/ssl/certs/` directory. If you come across this problem, you should consider updating your bootstrap scripts to ensure that zypper operates as expected.

Troubleshooting OSAD and jabberd

Open File Count Exceeded

The number of maximum files that a jabber user can open is lower than the number of connected clients. OSAD clients cannot contact the SUSE Manager Server, and jabberd requires long periods of time to respond on port 5222. Each client requires one permanently open TCP connection and each connection requires one file handler.

To resolve OSAD connection problems, edit such lines in `/etc/security/limits.conf`:

```
jabber   soft   nofile   <#clients + 100>
jabber   hard   nofile   <#clients + 1000>
```

Itwill vary according to your setup. For example, in the case of 5000 clients:

```
jabber   soft   nofile   5100
jabber   hard   nofile   6000
```

Ensure you also update the `max_fds` parameter in `/etc/jabberd/c2s.xml`. For example: `<max_fds>6000</max_fds>`

The soft file limit is the limit of the maximum number of open files for a single process. In Uyuni the highest consuming process is c2s, which opens a connection per client. 100 additional files are added, here, to accommodate for any non-connection file that c2s requires to work correctly. The hard limit applies to all processes belonging to the jabber user, and accounts for open files from the router, s2s and sm processes additionally.

jabberd Database Corruption

`SYMPTOMS`: After *a disk is full error* or a *disk crash event*, the `jabberd` database may have become corrupted. `jabberd` may then fail starting Spacewalk services:

```
Starting spacewalk services...
   Initializing jabberd processes...
       Starting router                                                    done
       Starting sm startproc:  exit status of parent of /usr/bin/sm: 2
failed
   Terminating jabberd processes...
```

`/var/log/messages` shows more details:

```
jabberd/sm[31445]: starting up
jabberd/sm[31445]: process id is 31445, written to /var/lib/jabberd/pid/sm.pid
jabberd/sm[31445]: loading 'db' storage module
jabberd/sm[31445]: db: corruption detected! close all jabberd processes and run db_recover
jabberd/router[31437]: shutting down
```

CURE: Remove the jabberd database and restart. jabberd will automatically re-create the database. Enter at the command prompt:

```
spacewalk-service stop
rm -rf /var/lib/jabberd/db/*
spacewalk-service start
```

Capturing XMPP Network Data for Debugging Purposes

If you are experiencing bugs regarding OSAD, it can be useful to dump network messages in order to help with debugging. The following procedures provide information on capturing data from both the client and server side.

Procedure: Server Side Capture

1. Install the tcpdump package on the server as root:

   ```
   zypper in tcpdump
   ```

2. Stop the OSA dispatcher and Jabber processes:

   ```
   rcosa-dispatcher stop
   rcjabberd stop
   ```

3. Start data capture on port 5222:

   ```
   tcpdump -s 0 port 5222 -w server_dump.pcap
   ```

4. Open a second terminal and start the OSA dispatcher and Jabber processes:

   ```
   rcosa-dispatcher start
   rcjabberd start
   ```

5. Operate the server and clients so the bug you formerly experienced is reproduced.

6. When you have finished your capture re-open the first terminal and stop the data capture with *CTRL+C*.

Procedure: Client Side Capture

1. Install the tcpdump package on your client as root:

```
zypper in tcpdump
```

2. Stop the OSA process:

```
rcosad stop
```

3. Begin data capture on port 5222:

```
tcpdump -s 0 port 5222 -w client_client_dump.pcap
```

4. Open a second terminal and start the OSA process:

```
rcosad start
```

5. Operate the server and clients so the bug you formerly experienced is reproduced.
6. When you have finished your capture re-open the first terminal and stop the data capture with *CTRL+C*.

Troubleshooting Package Inconsistencies

Packages can sometimes be locked or taskomatic can experience problems, which creates problems with metadata regeneration. When this occurs, package updates will be available in the Web UI, but will not appear on the client, and attempts to update the client will fail. To correct this, determine if any processes are running, or if a crash could have occured. Check package locks and exclude lists to determine if packages are locked or excluded on the client. When you have located the problematic process, the metadata can be regenerated and synchronization occurs as expected.

Procedure: Resolving Package Inconsisties

1. On the server, check the `/var/log/rhn/rhn_taskomatic_daemon.log` file to determine if any processes are still running or a crash occured.
2. Restart taskomatic:

```
service taskomatic restart
```

3. On the client, check package locks and exclude lists to determine if packages are locked or excluded:
    - On an Expanded Support Platform, check `/etc/yum.conf` and search for `exclude=`.
    - On SUSE Linux Enterprise, use the `zypper locks` command.

Troubleshooting Registering Cloned Clients

Sometimes a cloned client (either traditional or Salt) will use the same machine ID as the system they are a clone of. This results in Uyuni only recognizing one system, rather than two different systems. This can be resolved by changing the machine ID of the cloned system, so that Uyuni recognizes them as two different clients.

Each step of this procedure is performed on the cloned system. This procedure does not manipulate the original system, which will still be registered to Uyuni. The cloned virtual machine must have a different UUID than the original (the UUID is generated by your hypervisor), otherwise Uyuni will override the original system data with the new data.

Procedure: Resolving Duplicate Machine IDs in Cloned Salt Clients

1. For SLES 12: If your machines have the same machine ID, delete the file on each client and re-create it:

```
# rm /etc/machine-id
# rm /var/lib/dbus/machine-id
# dbus-uuidgen --ensure
# systemd-machine-id-setup
```

2. For SLES 11: As there is no systemd machine ID, generate one from dbus:

```
# rm /var/lib/dbus/machine-id
# dbus-uuidgen --ensure
```

3. If your machines still have the same Salt client ID, delete the `minion_id` file on each client (FQDN will be used when it is regenerated on client restart):

```
# rm /etc/salt/minion_id
```

4. Delete accepted keys from the Onboarding page and the system profile from Uyuni, and restart the client with:

```
# service salt-minion restart
```

5. Re-register the clients. Each client will now have a different `/etc/machine-id` and should now be correctly displayed on the Uyuni System Overview page.

Procedure: Resolving Duplicate Machine IDs in Cloned Traditional Clients

1. On the cloned machine, change the hostname and IP addresses. Make sure `/etc/hosts` contains the changes you made and the correct host entries.

2. Stop the `rhnsd` daemon, on Red Hat Enterprise Linux Server 6 and SUSE Linux Enterprise 11 with:

```
# /etc/init.d/rhnsd stop
```

or, on newer systemd-based systems, with:

```
# service rhnsd stop
```

3. Stop `osad` with:

```
# /etc/init.d/osad stop
```

or:

```
# service osad stop
```

or:

```
# rcosad stop
```

4. Remove the `osad` authentication configuration file and the system ID:

```
# rm -f /etc/sysconfig/rhn/{osad-auth.conf,systemid}
```

5. Delete the files containing the machine IDs:
   - SLES 12:

```
# rm /etc/machine-id
# rm /var/lib/dbus/machine-id
# dbus-uuidgen --ensure
# systemd-machine-id-setup
```

   - SLES 11:

```
# suse_register -E
```

   - SLES 10:

```
# rm -rf /etc/{zmd,zypp}
# rm -rf /var/lib/zypp/!(db)
# rm -rf /var/lib/zmd/
```

6. Remove the credential files:
    - SLES clients:

```
# rm -f /etc/zypp/credentials.d/{SCCcredentials,NCCcredentials}
```

    - Red Hat Enterprise Linux clients:

```
# rm  -f /etc/NCCcredentials
```

7. Re-run the bootstrap script. You should now see the cloned system in Uyuni without overriding the system it was cloned from.

Troubleshooting RPC Connection Timeouts

RPC connections can sometimes time out due to slow networks or a network link going down. This results in package downloads or batch jobs hanging or taking longer than expected. You can adjust the maximum time that an RPC connection can take by editing the configuration file. While this will not resolve networking problems, it will cause a process to fail rather than hang.

Procedure: Resolving RPC connection timeouts

1. On the Uyuni Server, open the `/etc/rhn/rhn.conf` file and set a maximum timeout value (in seconds):

```
server.timeout =`number`
```

2. On the Uyuni Proxy, open the `/etc/rhn/rhn.conf` file and set a maximum timeout value (in seconds):

```
proxy.timeout =`number`
```

3. On a SUSE Linux Enterprise Server client that uses zypper, open the `/etc/zypp/zypp.conf` file and set a maximum timeout value (in seconds):

```
## Valid values:  [0,3600]
## Default value: 180
download.transfer_timeout = 180
```

4. On a Red Hat Enterprise Linux client that uses yum, open the `/etc/yum.conf` file and set a

maximum timeout value (in seconds):

```
timeout =`number`
```

If you limit RPC timeouts to less than `180` seconds, you risk aborting perfectly normal operations.

Troubleshooting the Saltboot Formula

Because of a problem in the computed partition size value, the saltboot formula can sometimes fail when it is created on SLE 11 SP3 clients, with an error like this:

```
          ID: disk1_partitioned
    Function: saltboot.partitioned
        Name: disk1
      Result: false
     Comment: An exception occurred in this state: Traceback (most recent call last):
  File "/usr/lib/python2.6/site-packages/salt/state.py", line 1767, in call
    **cdata['kwargs'])
  File "/usr/lib/python2.6/site-packages/salt/loader.py", line 1705, in wrapper
    return f(*args, **kwargs)
  File "/var/cache/salt/minion/extmods/states/saltboot.py", line 393, in disk_partitioned
    existing = __salt__['partition.list'](device, unit='MiB')
  File "/usr/lib/python2.6/site-packages/salt/modules/parted.py", line 177, in list_
    'Problem encountered while parsing output from parted')
CommandExecutionError: Problem encountered while parsing output from parted
```

This problem can be resolved by manually configuring the size of the partition containing the operating system. When the size is set correctly, formula creation will work as expected.

Procedure: Manually Configuring the Partition Size in the Saltboot Formula

1. In the Uyuni Web UI, navigate to **Systems** › **System Groups** and select the `Hardware Type Group` that contains the SLE 11 SP3 client that is causing the error. In the `Formulas` tab, navigate to the `Saltboot` subtab.
2. Locate the partition that contains the operating system, and in the `Partition Size` field, type the appropriate size (in MiB).
3. Click **`[Save Formula]`**, and apply the highstate to save your changes.

AutoYast Example File

Minimalist AutoYaST Profile for Automated Installations and Useful Enhancements

The AutoYaST profile in this section installs a SUSE Linux Enterprise Server system with all default installation options including a default network configuration using DHCP. After the installation is finished, a bootstrap script located on the Uyuni server is executed in order to register the freshly installed system with Uyuni. You need to adjust the IP address of the Uyuni server, the name of the bootstrap script, and the root password according to your environment:

```
<user>
 ...
 <username>root</username>
 <user_password>`linux`</user_password>
</user>

<location>http://`192.168.1.1`/pub/bootstrap/`my_bootstrap.sh`</location>
```

The complete AutoYaST file:

```
<?xml version="1.0"?>
<!DOCTYPE profile>
<profile xmlns="http://www.suse.com/1.0/yast2ns"
         xmlns:config="http://www.suse.com/1.0/configns">
 <general>
  <mode>
   <confirm config:type="boolean">false</confirm>
  </mode>
 </general>
 <networking>
  <keep_install_network config:type="boolean">true</keep_install_network>
 </networking>
 <software>
  <install_recommended config:type="boolean">true</install_recommended>
   <patterns config:type="list">
    <pattern>base</pattern>
   </patterns>
 </software>
 <users config:type="list">
  <user>
   <encrypted config:type="boolean">false</encrypted>
   <fullname>root</fullname>
   <gid>0</gid>
   <home>/root</home>
   <password_settings>
    <expire></expire>
    <flag></flag>
    <inact></inact>
    <max></max>
    <min></min>
    <warn></warn>
   </password_settings>
   <shell>/bin/bash</shell>
   <uid>0</uid>
   <username>root</username>
   <user_password>linux</user_password>
  </user>
 </users>
 <scripts>
  <init-scripts config:type="list">
   <script>
    <interpreter>shell</interpreter>
    <location>http://192.168.1.1/pub/bootstrap/my_bootstrap.sh</location>
   </script>
  </init-scripts>
 </scripts>
</profile>
```

Use this enhancement fragment to add child channels:

```
<add-on>
 <add_on_products config:type="list">
  <listentry>
   <ask_on_error config:type="boolean">true</ask_on_error>
   <media_url>http://$c_server/ks/dist/child/`channel-label`/`distribution-label`</media_url>
   <name>$c_name</name>
   <product>$c_product</product>
   <product_dir>/</product_dir>
  </listentry>
  <listentry>
  <!-- SLES SUSE Manager tools Pool -->
    <media_url>http://$c_server/ks/dist/child/`channel-label`/`sle-manager-
tools`/`distribution-label`</media_url>
    ...
  </listentry>
  ...
 </add_on_products>
</add-on>
```

Replace `channel-label` and `distribution-label` with the correct labels (such as `sles12-sp4-updates-x86_64` and `sles12-sp4-x86_64`). Ensure that the distribution label corresponds to the Autoinstallable Distribution. Set the variables (such as `$c_server`) according to your environment. For more information about variables, see **[Reference › Systems ›]**.

Here is a literal example for `sles12-sp4-x86_64`:

```
<add-on>
  <add_on_products config:type="list">
    <listentry>
      <!-- SLES12 Updates -->
      <media_url>http://192.168.150.10/ks/dist/child/dev-sles12-sp4-updates-x86_64/dev-
sles12sp4</media_url>
      <product>SLES 12 Updates</product>
      <product_dir>/</product_dir>
      <name>SLES12 Updates</name>
    </listentry>
    <listentry>
      <!-- SLES12 SUSE Manager Tools Pool -->
      <media_url>http://192.168.150.10/ks/dist/child/dev-sle-manager-tools12-pool-x86_64-
sp4/dev-sles12sp4</media_url>
      <product>SLES 12 Pool SUSE Manager Tools</product>
      <product_dir>/</product_dir>
      <name>SLES12 Pool SUSE Manager Tools</name>
    </listentry>
    <listentry>
      <!-- SLES12  SUSE Manager Tools Updates -->
      <media_url>http://192.168.150.10/ks/dist/child/dev-sle-manager-tools12-updates-x86_64-
sp4/dev-sles12sp4</media_url>
      <product>SLES 12 Updates SUSE Manager Tools</product>
      <product_dir>/</product_dir>
      <name>SLES12 Updates SUSE Manager Tools</name>
    </listentry>
  </add_on_products>
</add-on>
```

Add the Updates Channel

It is required that you add the updates tools channel to the `<add-on>` AutoYaST snippet section. This ensures your systems are provided with an up-to-date version of the `libzypp` package. If you do not include the updates tools channel, you will encounter `400` errors. In this example, the (DISTRIBUTION_NAME) is replaced with the name of the autoinstallation distribution from **Systems › Autoinstallation › Distributions**.

```
<listentry>
  <ask_on_error config:type="boolean">true</ask_on_error>
   <media_url>http://$redhat_management_server/ks/dist/child/sles12-
sp2-updates-x86_64/(DISTRIBUTION_NAME)</media_url>
  <name>sles12 sp2 updates</name>
  <product>SLES12</product>
  <product_dir>/</product_dir>
</listentry>
```