Uyuni 2026.11

Installation and Upgrade Guide



UYUNI

Chapter 1. Preface

Installation, Deployment and Upgrade
Uyuni 2026.11

This guide provides comprehensive, step-by-step instructions for deploying, upgrading, and managing Uyuni Server and Proxy.

It is organized into the following sections:

- **Requirements:** Outlines the essential hardware, software, and networking prerequisites to ensure a smooth setup.
- **Deployment and Installation:** Guides you through deploying Uyuni as a container and completing the initial configuration.
- **Upgrade and Migration:** Details the process for upgrading and migrating Uyuni while minimizing downtime.
- **Basic Server Management:** Covers fundamental server operations, helping you get started with Uyuni efficiently.

Publication Date: 2026-11-25

Chapter 2. Terminology

This guide uses the following terms consistently:

Upgrade

Moving to a newer Uyuni release. Update the host operating system packages, then run mgradm upgrade or mgrpxy upgrade.

Migration

Changing the host operating system, for example from openSUSE Leap Micro 5.5 to openSUSE Tumbleweed, or changing the deployment form.

Uyuni releases are rolling and have no separate maintenance-update concept.

Table of Contents

1. Preface 1
2. Terminology 2
3. Requirements 5
3.1. General Requirements 5
3.1.1. Server Requirements 5
3.1.2. Proxy Requirements 5
3.2. Network Requirements 6
3.2.1. Fully qualified domain name (FQDN) 7
3.2.2. Hostname and IP address 7
3.2.3. Reenable router advertisements 7
3.2.4. Deployment behind HTTP or HTTPS OSI level 7 proxy 8
3.2.5. Air-gapped deployment 9
3.2.6. Required network ports 9
3.3. Public Cloud Requirements 16
3.3.1. Network requirements 17
3.3.2. Prepare storage volumes 17
4. Deployment and Installation 19
4.1. Install Uyuni Server 19
4.1.1. Uyuni Server Deployment on openSUSE Tumbleweed 19
4.1.2. Uyuni Server Air-gapped Deployment 22
4.2. Install Uyuni Proxy 24
4.2.1. Uyuni Proxy Deployment on openSUSE Tumbleweed 25
4.2.2. Containerized Uyuni Proxy Setup 30
4.2.3. Proxy conversion from client 35
5. Upgrade and Migration 39
5.1. Server 39
5.1.1. Server Upgrade 39
5.1.2. Migrating the Uyuni Server to openSUSE Tumbleweed 41
5.2. Proxy 45
5.2.1. Proxy Upgrade 45
5.2.2. Migrating the Uyuni Proxy to openSUSE Tumbleweed 47
5.2.3. Legacy Proxy Migration to Container 50
5.3. Clients 54
5.3.1. Upgrade Clients 54
6. Basic Server and Proxy Management 55
6.1. Custom YAML Configuration and Deployment with mgradm 55
6.2. Starting and Stopping Containers 56
6.3. Containers used by Uyuni 56
6.4. Persistent Container Volumes 57
6.4.1. Server 57
6.4.2. Proxy 59
6.5. Understanding mgr-storage-server and mgr-storage-proxy 60
6.5.1. What these tools do 60

6.5.2. What these tools do **not** do 61
6.5.3. Post-installation storage management 61
6.5.4. When to use, or not use 62
6.5.5. Summary 62
7. GNU Free Documentation License 63

Chapter 3. Requirements

3.1. General Requirements

The following tables specify the minimum server and proxy requirements.



Do not use NFS for storage because it does not support SELinux file labeling.

3.1.1. Server Requirements

Table 1. Server Requirements for x86-64 Architecture

Software and Hardware	Details	Recommendation
Tumbleweed	Clean installation, up-to-date	Tumbleweed
CPU	-	Minimum 4 dedicated 64-bit CPU cores (x86-64)
RAM	Test or Base Installation	Minimum 16 GB
	Recommended	Minimum 32 GB
Disk Space	/ (root directory)	40 GB
	/var/lib/containers/storage/volumes	Minimum 150 GB (depending on the number and type of the product*) * 50 GB for each SUSE product and Package Hub * 360 GB for each Red Hat product
	/var/lib/containers/storage/volumes/var-pgsql	Minimum 50 GB
Swap space	Systems can benefit from additional swap space. It is recommended to use a swap file instead of a swap partition.	8 to 12 GB

3.1.2. Proxy Requirements

Table 2. Proxy Requirements

Software and Hardware	Details	Recommendation
Tumbleweed	Clean installation, up-to-date	Tumbleweed
CPU		Minimum 2 dedicated 64-bit CPU cores
RAM	Test Server	Minimum 2 GB
	Production Server	Minimum 8 GB
Disk Space	/ (root directory)	Minimum 40 GB
	/var/lib/containers/storage/volumes	Minimum 100 GB

Uyuni Proxy caches packages in the /var/lib/containers/storage/volumes/uyuni-proxy-squid-cache/ directory. If there is not enough space available in /var/lib/containers/storage/volumes/uyuni-proxy-squid-cache/, the proxy will remove old, unused packages and replace them with newer packages.

As a result of this behavior:

- The larger /var/lib/containers/storage/volumes/uyuni-proxy-squid-cache/ directory is on the proxy, the less traffic there will be between it and the Uyuni Server.
- By making the /var/lib/containers/storage/volumes/uyuni-proxy-squid-cache/ directory on the proxy the same size as /var/lib/containers/storage/volumes/var-spacewalk/ on the Uyuni Server, you avoid a large amount of traffic after the first synchronization.
- The /var/lib/containers/storage/volumes/uyuni-proxy-squid-cache/ directory can be small on the Uyuni Server compared to the proxy. For a guide to size estimation, see the Server Requirements section.

3.2. Network Requirements

This section details the networking and port requirements for Uyuni.



IP forwarding will be enabled by containerized installation. This means Uyuni Server and Proxies will behave as a router. This behavior is done by podman directly. Podman containers do not run if IP forwarding is disabled.

Consider achieving network isolation of the Uyuni environment according to your policies.

For more information, see https://www.suse.com/support/kb/doc/?id=000020166.

3.2.1. Fully qualified domain name (FQDN)

The Uyuni server must resolve its FQDN correctly. If the FQDN cannot be resolved, it can cause serious problems in a number of different components.

For more information about configuring the hostname and DNS, see https://documentation.suse.com/sles/15-SP7/html/SLES-all/cha-network.html#sec-network-yast-change-host.

3.2.2. Hostname and IP address

To ensure that the Uyuni domain name can be resolved by its clients, both server and client machines must be connected to a working DNS server. You also need to ensure that reverse lookups are correctly configured.

For more information about setting up a DNS server, see https://documentation.suse.com/sles/15-SP7/html/SLES-all/cha-dns.html.

3.2.3. Reenable router advertisements

When the Uyuni is installed using mgradm install or mgrpxy install, it sets up Podman which enables IPv4 and IPv6 forwarding. This is needed for communication from the outside of the container.

However, if your system previously had /proc/sys/net/ipv6/conf/eth0/accept_ra set to 1, it will stop using router advertisements. As a result, the routes are no longer obtained via router advertisements and the default IPv6 route is missing.

To recover correct functioning of the IPv6 routing, follow the procedure depending on whether:

- server and proxy are based on 15 SP7 (without Network manager)
- server and proxy are based on SL Micro 6.2 (with Network manager)

Procedure: Reenabling router advertisements without Network Manager

1. Create a file in /etc/sysctl.d, for example 99-ipv6-ras.conf.
2. Add the following parameter and value to the file:

```
net.ipv6.conf.eth0.accept_ra = 2
```

3. Reboot.



Network management is not working when you have no wicked.

Procedure: Reenabling router advertisements with Network Manager

1. List your connections with nmcli connection show.
2. Create or modify the file /etc/NetworkManager/system-connections/<name of connection>.nmconnection to add this setting:

```
[ipv6]
addr-gen-mode=eui64
```

3. Reboot.
4. The file should look similar to this:

```
[connection]
id=Wired connection 1
type=ethernet
interface-name=eth0

[ethernet]

[ipv4]
dns-priority=20
method=auto

[ipv6]
addr-gen-mode=eui64
method=auto
```

3.2.4. Deployment behind HTTP or HTTPS OSI level 7 proxy

Some environments enforce internet access through a HTTP or HTTPS proxy. This could be a Squid server or similar. To allow the Uyuni Server internet access in such configuration, you need to configure the following.

Procedure: Configuring HTTP or HTTPS OSI level 7 proxy

1. For operating system internet access, modify /etc/sysconfig/proxy according to your needs:

```
PROXY_ENABLED="no"
HTTP_PROXY=""
HTTPS_PROXY=""
NO_PROXY="localhost, 127.0.0.1"
```

2. For Podman container internet access, modify /etc/systemd/system/uyuni-server.service.d/custom.conf according to your needs. For example, set:

```
[Service]
Environment=TZ=Europe/Berlin
Environment="PODMAN_EXTRA_ARGS="
Environment="https_proxy=user:password@http://192.168.10.1:3128"
```

3. For Java application internet access, modify /etc/rhn/rhn.conf according to your needs. On the container host, execute mgrctl term to open a command line inside the server container:
    a. Modify /etc/rhn/rhn.conf according to your needs. For example, set:

```
# Use proxy FQDN, or FQDN:port
server.satellite.http_proxy =
server.satellite.http_proxy_username =
server.satellite.http_proxy_password =
# no_proxy is a comma separated list
server.satellite.no_proxy =
```

4. On the container host, restart the server to enforce the new configuration:

```
systemctl restart uyuni-server.service
```

3.2.5. Air-gapped deployment

If you are on an internal network and do not have access to SUSE Customer Center, you can use an **Installation and Upgrade Guide › Container Deployment › Air-gapped Deployment**.

In a production environment, the Uyuni Server and clients should always use a firewall. For a comprehensive list of the required ports, see **Installation and Upgrade Guide › Network Requirements › Ports**.

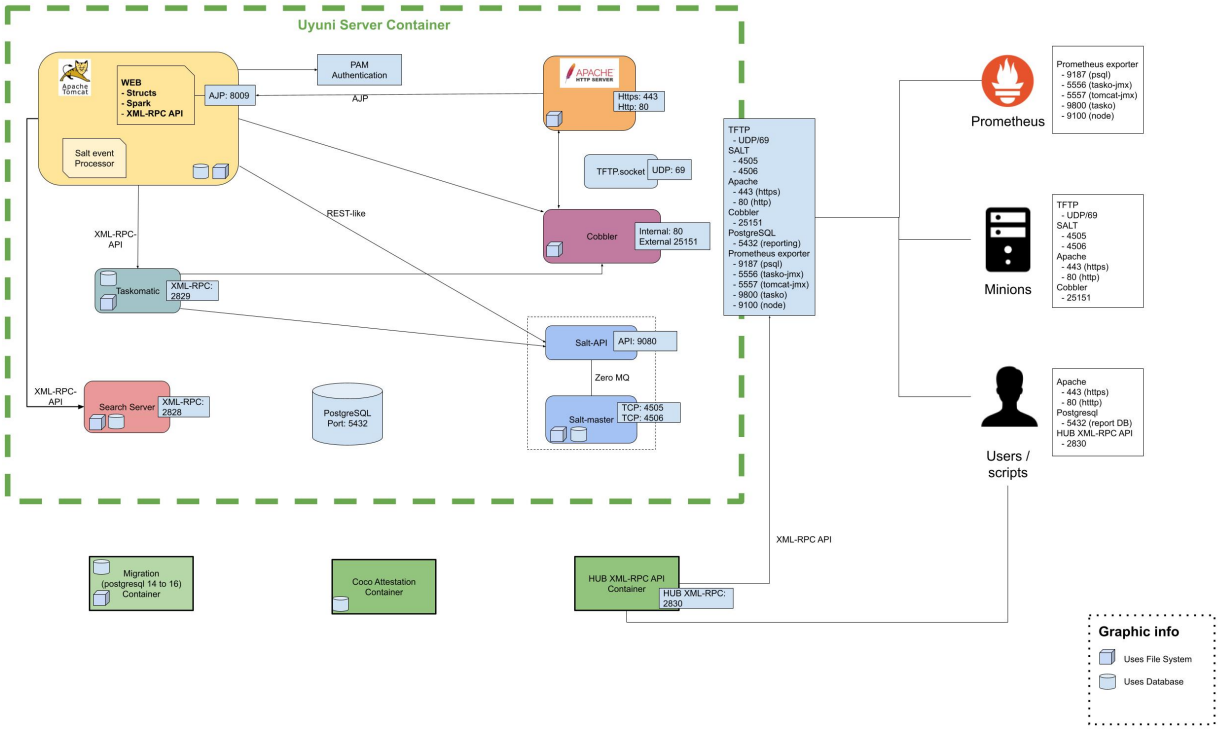
3.2.6. Required network ports

This section contains a comprehensive list of ports that are used for various communications within Uyuni.

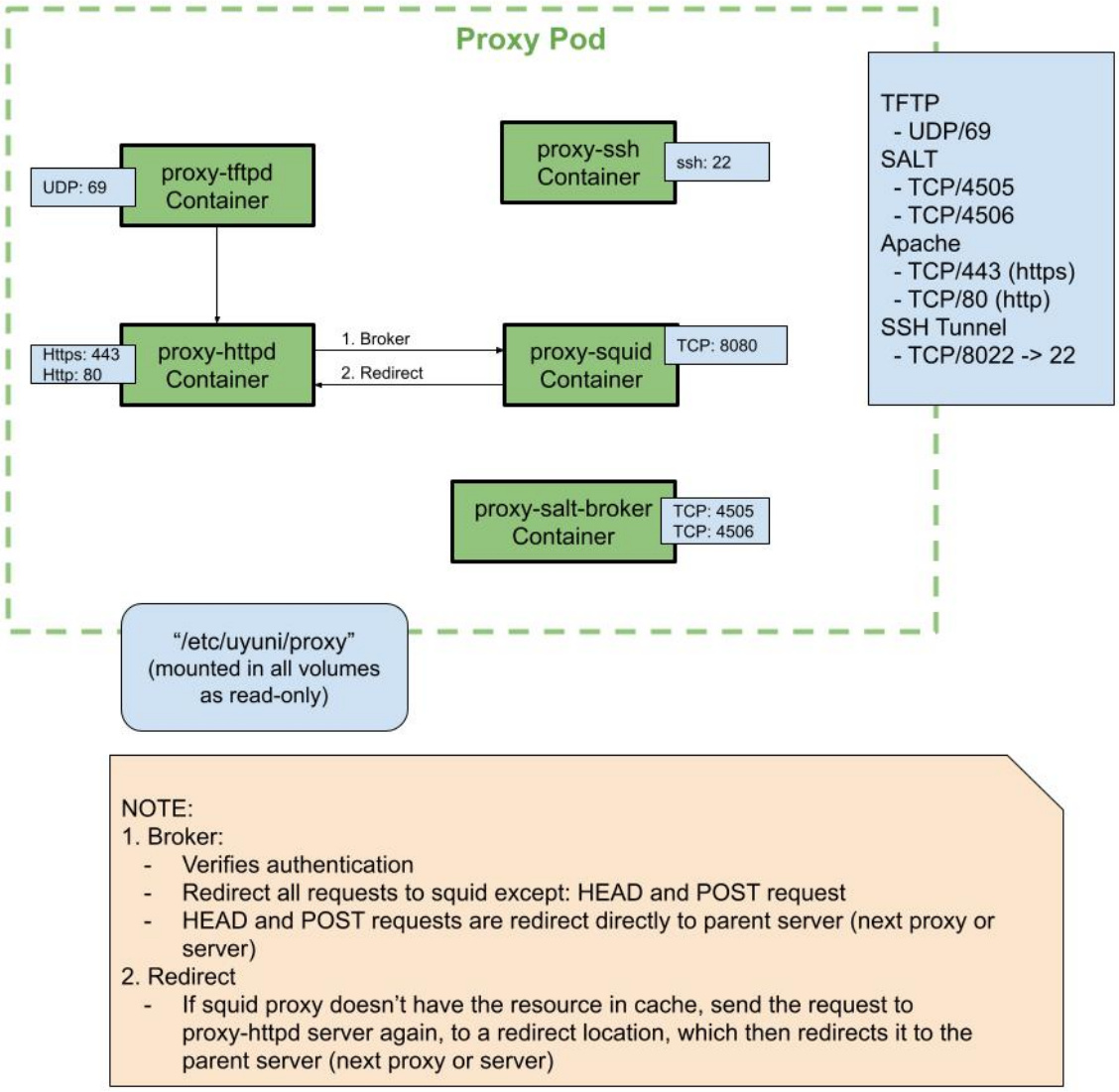
You will not need to open all of these ports. Some ports only need to be opened if you are using the service that requires them.

3.2.6.1. Overview

3.2.6.1.1. Server


Uyuni Server Container
WEB
- Structs
- Spark
- XML-RPC API
AJP: 8009
PAM Authentication
AJP
Salt event Processor
Https: 443
Http: 80
TFTP socket
UDP: 69
REST-like
Cobbler
Internal: 80
External 25151
XML-RPC-API
Taskomatic
XML-RPC: 2829
Salt-API
API: 9080
Zero MQ
Salt-master
TCP: 4505
TCP: 4506
XML-RPC-API
Search Server
XML-RPC: 2828
PostgreSQL Port: 5432
TFTP
- UDP/69
SALT
- 4505
- 4506
Apache
- 443 (https)
- 80 (http)
Cobbler
- 25151
PostgreSQL
- 5432 (reporting)
Prometheus exporter
- 9187 (psql)
- 5556 (tasko-jmx)
- 5557 (tomcat-jmx)
- 9800 (tasko)
- 9100 (node)
Prometheus
Prometheus exporter
- 9187 (psql)
- 5556 (tasko-jmx)
- 5557 (tomcat-jmx)
- 9800 (tasko)
- 9100 (node)
Minions
TFTP
- UDP/69
SALT
- 4505
- 4506
Apache
- 443 (https)
- 80 (http)
Cobbler
- 25151
Users / scripts
Apache
- 443 (https)
- 80 (http)
Postgresql
- 5432 (report DB)
HUB XML-RPC API
- 2830
XML-RPC API
Migration (postgresql 14 to 16) Container
Coco Attestation Container
HUB XML-RPC API Container
HUB XML-RPC: 2830
Graphic info
Uses File System
Uses Database

3.2.6.1.2. Proxy



3.2.6.2. External inbound server ports

External inbound ports must be opened to configure a firewall on the Uyuni Server to protect the server from unauthorized access.

Opening these ports allows external network traffic to access the Uyuni Server.

Table 3. External Port Requirements for Uyuni Server

Port number	Protocol	Used By	Notes
67	TCP/UDP	DHCP	Required only if clients are requesting IP addresses from the server.

Port number	Protocol	Used By	Notes
69	TCP/UDP	TFTP	Required if server is used as a PXE server for automated client installation (starting with version 5.2, TFTP is disabled on the server by default and is recommended to run on the proxy).
80	TCP	HTTP	Required temporarily for some bootstrap repositories and automated installations.
443	TCP	HTTPS	Serves the Web UI, client, and server and proxy (tftpsync) requests.
4505	TCP	salt	Required to accept communication requests from clients. The client initiates the connection, and it stays open to receive commands from the Salt master.
4506	TCP	salt	Required to accept communication requests from clients. The client initiates the connection, and it stays open to report results back to the Salt master.
5432	TCP	PostgreSQL	Required to access the reporting database.
5556	TCP	Prometheus	Required for scraping Taskomatic JMX metrics.
5557	TCP	Prometheus	Required for scraping Tomcat JMX metrics.
9100	TCP	Prometheus	Required for scraping Node exporter metrics.
9187	TCP	Prometheus	Required for scraping PostgreSQL metrics.
9800	TCP	Prometheus	Required for scraping Taskomatic metrics.

3.2.6.3. External outbound server ports

External outbound ports must be opened to configure a firewall on the Uyuni Server to restrict what the server can access.

Opening these ports allows network traffic from the Uyuni Server to communicate with external services.

Table 4. External Port Requirements for Uyuni Server

Port number	Protocol	Used By	Notes
80	TCP	HTTP	Required for SUSE Customer Center. Port 80 is not used to serve the Web UI.

Port number	Protocol	Used By	Notes
443	TCP	HTTPS	Required for SUSE Customer Center.

3.2.6.4. Internal server ports

Internal ports are used internally by the Uyuni Server. Internal ports are only accessible from localhost.

In most cases, you will not need to adjust these ports.

Table 5. Internal Port Requirements for Uyuni Server

Port number	Notes
2828	Satellite-search API, used by the RHN application in Tomcat and Taskomatic.
2829	Taskomatic API, used by the RHN application in Tomcat.
8005	Tomcat shutdown port.
8009	Tomcat to Apache HTTPD (AJP).
8080	Tomcat to Apache HTTPD (HTTP).
9080	Salt-API, used by the RHN application in Tomcat and Taskomatic.
32000	Port for a TCP connection to the Java Virtual Machine (JVM) that runs Taskomatic and satellite-search.

Port 32768 and higher are used as ephemeral ports. These are most often used to receive TCP connections. When a TCP connection request is received, the sender will choose one of these ephemeral port numbers to match the destination port.

You can use this command to find out which ports are ephemeral ports:

```
cat /proc/sys/net/ipv4/ip_local_port_range
```

3.2.6.5. External inbound proxy ports

External inbound ports must be opened to configure a firewall on the Uyuni Proxy to protect the proxy from unauthorized access.

Opening these ports allows external network traffic to access the Uyuni proxy.

Table 6. External Port Requirements for Uyuni Proxy

Port number	Protocol	Used By	Notes
22			Only required if the user wants to manage the proxy host with Salt SSH.
67	TCP/UDP	DHCP	Required only if clients are requesting IP addresses from the server.
69	TCP/UDP	TFTP	Required if the proxy is used as a PXE server for automated client installation (recommended).
443	TCP	HTTPS	Web UI, client, and server and proxy (tftpsync) requests.
4505	TCP	salt	Required to accept communication requests from clients. The client initiates the connection, and it stays open to receive commands from the Salt master.
4506	TCP	salt	Required to accept communication requests from clients. The client initiates the connection, and it stays open to report results back to the Salt master.
8022			Required for ssh-push and ssh-push-tunnel contact methods. Clients connected to the proxy initiate check in on the server and hop through to clients.

3.2.6.6. External outbound proxy ports

External outbound ports must be opened to configure a firewall on the Uyuni Proxy to restrict what the proxy can access.

Opening these ports allows network traffic from the Uyuni Proxy to communicate with external services.

Table 7. External Port Requirements for Uyuni Proxy

Port number	Protocol	Used By	Notes
80			Used to reach the server.
443	TCP	HTTPS	Required for SUSE Customer Center.
4505	TCP	Salt	Required to connect to Salt master either directly or via proxy.
4506	TCP	Salt	Required to connect to Salt master either directly or via proxy.

3.2.6.7. External client ports

External client ports must be opened to configure a firewall between the Uyuni Server and its clients.

In most cases, you will not need to adjust these ports.

Table 8. External Port Requirements for Uyuni Clients

Port number	Direction	Protocol	Notes
22	Inbound	SSH	Required for ssh-push and ssh-push-tunnel contact methods.
80	Outbound		Used to reach the server or proxy.
443	Outbound		Used to reach the server or proxy.
4505	Outbound	TCP	Required to connect to Salt master either directly or via proxy.
4506	Outbound	TCP	Required to connect to Salt master either directly or via proxy.
9090	Outbound	TCP	Required for Prometheus user interface.
9093	Outbound	TCP	Required for Prometheus alert manager.
9100	Outbound	TCP	Required for Prometheus node exporter.
9117	Outbound	TCP	Required for Prometheus Apache exporter.
9187	Outbound	TCP	Required for Prometheus PostgreSQL.

3.2.6.8. Required URLs

There are some URLs that Uyuni must be able to access to register clients and perform updates. In most cases, allowing access to these URLs is sufficient:

- scc.suse.com
- updates.suse.com
- installer-updates.suse.com
- registry.suse.com
- registry-storage.suse.com
- opensuse.org

In order to perform confidential computing attestation for IBM, the system needs to be to be able to autonomously download the certificates and the certificate-revocation lists. Therefore the following addresses must be reachable, even behind a HTTP proxy:

- www.ibm.com/support/resourcelink/api/content/public/* (this includes IBM certificates and IBM CRL distribution points)
- crl3.digicert.com/ and crl4.digicert.com (this includes CRL distribution points for DigiCertCA.crt, since the chain must be checked for`` the CRLs)

Additionally, you may need access to these URLs for non-SUSE products:

- download.nvidia.com
- public.dhe.ibm.com
- nu.novell.com

You can find additional details on whitelisting the specified URLs and their associated IP addresses in this article: Accessing SUSE Customer Center and SUSE registry behind a firewall and/or through a proxy.

If you are using non-SUSE clients you might also need to allow access to other servers that provide specific packages for those operating systems. For example, if you have Ubuntu clients, you will need to be able to access the Ubuntu server.

For more information about troubleshooting firewall access for non-SUSE clients, see **Administration Guide › Troubleshooting › Tshoot Firewalls**.

3.3. Public Cloud Requirements

This section provides the requirements for installing Uyuni on public cloud infrastructure. We have tested these instructions on Amazon EC2, Google Compute Engine, and Microsoft Azure, but they should work on other providers as well, with some variation.

Before you begin, here are some considerations:

- The Uyuni setup procedure performs a forward-confirmed reverse DNS lookup. This must succeed in order for the setup procedure to complete and for Uyuni to operate as expected. It is important to perform hostname and IP configuration before you set up Uyuni.
- Uyuni Server and Proxy instances need to run in a network configuration that provides you control over DNS entries, but cannot be accessed from the internet at large.
- 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 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. This section includes instructions for setting up an external virtual disk.

3.3.1. Network requirements

When you use Uyuni on a public cloud, you must use a restricted network. We recommend using a VPC private subnet with an appropriate firewall setting. Only machines in your specified IP ranges must be able to access the instance.



Running Uyuni on the public cloud means implementing robust security measures. It is essential to limit, filter, monitor, and audit access to the instance. SUSE strongly advises against a globally accessible Uyuni instance that lacks adequate perimeter security.

To access the Uyuni Web UI, allow HTTPS when configuring the network access controls. This allows you to access the Uyuni Web UI.

In EC2 and Azure, create a new security group, and add inbound and outbound rules for HTTPS. In GCE, check the Allow HTTPS traffic box under the Firewall section.

3.3.2. Prepare storage volumes

We recommend that the repositories and the database for Uyuni are stored on separate storage devices from the root volume. This will help to avoid data loss and possibly increase performance.

The Uyuni container utilizes default storage locations. These locations should be configured prior to deployment for custom storage. For more information see **Installation and Upgrade Guide › Container Management › Persistent container volumes**



Do not use logical volume management (LVM) for public cloud installations.

The size of the disk for repositories storage is dependent on the number of distributions and channels you intend to manage with Uyuni. When you attach the virtual disks, they will appear in your instance as Unix device nodes. The names of the device nodes will vary depending on your provider, and the instance type selected.

Ensure the root volume of the Uyuni Server is 100 GB or larger. Add an additional storage disk of 500 GB or more, and choose SSD storage if you can. The cloud images for Uyuni Server use a script to assign this separate volume when your instance is launched.

When you launch your instance, you can log in to the Uyuni Server and use this command to find all available storage devices:

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

If you are not sure which device to choose, use the lsblk command to see the name and size of each device. Choose the name that matches with the size of the virtual disk you are looking for.

You can set up the external disk with the mgr-storage-server command. This creates an XFS partition mounted at /manager_storage and uses it as the location for the database and repositories:

```
/usr/bin/mgr-storage-server <devicename>
```

Chapter 4. Deployment and Installation

4.1. Install Uyuni Server

There are various scenarios to deploy a Uyuni Server.

4.1.1. Uyuni Server Deployment on openSUSE Tumbleweed

4.1.1.1. Deployment Preparations

In this section, you will gain expertise in setting up and deploying a Uyuni Server. The process encompasses the installation of Podman, Uyuni container utilities, deployment, and then initiating interaction with the container through mgrctl.



This section assumes you have already configured an openSUSE Tumbleweed host server, whether it is running on a physical machine or within a virtual environment.

https://download.opensuse.org/tumbleweed/

4.1.1.2. Container Host General Requirements

For general requirements, see **Installation and Upgrade Guide › General requirements**.

An openSUSE Tumbleweed server should be installed from installation media.

https://download.opensuse.org/tumbleweed/

This procedure is described below.

4.1.1.3. Container Host Requirements

For CPU, RAM, and storage requirements, see **Installation and Upgrade Guide › Hardware requirements**.



To guarantee that clients can resolve the FQDN domain name, both the containerized server and the host machines must be linked to a functional DNS server. Additionally, it is essential to ensure correct configuration of reverse lookups.

4.1.1.4. Installing Uyuni Tools For Use With Containers

Procedure: Installing Uyuni Tools on openSUSE Tumbleweed

1. On your local host, open a terminal window and log in.
2. Add the following repository to your openSUSE Tumbleweed server. You might need to use sudo for the following commands.

```
zypper ar
https://download.opensuse.org/repositories/systemsmanagement:/Uyuni:/Stable/ima
ges/repo/Uyuni-Server-POOL-$(arch)-Media1/ uyuni-server-stable
```

3. Refresh the repository list and import the key:

```
zypper ref
```

When prompted, trust and import the new repository GPG key.

4. Install the container tools:

```
zypper in mgradm mgrctl mgradm-bash-completion mgrctl-bash-completion uyuni-
storage-setup-server
```

For more information on the Uyuni Container Utilities, see Uyuni Container Utilities.

4.1.1.5. Configure Custom Persistent Storage

This step is optional. However, if custom persistent storage is required for your infrastructure, use the mgr-storage-server tool.

- For more information, see mgr-storage-server --help. This tool simplifies creating the container storage and database volumes.

Use the command in the following manner:

```
mgr-storage-server <storage-disk-device> [<database-disk-device>]
```

For example:

```
mgr-storage-server /dev/nvme1n1 /dev/nvme2n1
```



This command will create the persistent storage volumes at /var/lib/containers/storage/volumes.

> For more information, see **Installation and Upgrade Guide › Container Management › Persistent Container Volumes**.

4.1.1.6. Install and Enable Podman

Before deploying Uyuni, you must install and enable podman.

Procedure: Installing and Enabling Podman

1. Log in as root and install:

```
zypper in podman
```

2. Start and enable the Podman service:

```
systemctl enable --now podman.service
```

4.1.1.7. Deploying an Uyuni Container With Podman

4.1.1.7.1. mgradm Overview

Uyuni is deployed as a container using the mgradm tool. There are two methods of deploying a Uyuni server as a container. In this section we will focus on basic container deployment.

For information on using a custom configuration file to deploy, see **Installation and Upgrade Guide › Container Management › Mgradm Yaml Custom Configuration**.

For additional information, you can explore further by running mgradm --help from the command line.



> Uyuni server hosts that are hardened for security may restrict execution of files from the /tmp folder. In such cases, as a workaround, export the TMPDIR environment variable to another existing path before running mgradm.
>
> For example:
>
> ```
> export TMPDIR=/path/to/other/tmp
> ```
>
> In Uyuni updates, tools will be changed to make this workaround unnecessary.

Procedure: Deploying an Uyuni container with Podman

1. From the terminal run the following command as the sudo user or as root.

```
sudo mgradm install
```



You must deploy the container as sudo or root. The following error will be displayed at the terminal if you miss this step.

```
INF Setting up uyuni network
9:58AM INF Enabling system service
9:58AM FTL Failed to open /etc/systemd/system/uyuni-server.service
for writing
error="open /etc/systemd/system/uyuni-server.service: permission
denied"
```

2. Wait for deployment to complete.
3. Open a browser and proceed to your servers FQDN.

4.1.1.7.2. Persistent Volumes

Many users will want to specify locations for their persistent volumes.



If you are just testing out Uyuni you do not need to specify these volumes. mgradm will setup the correct volumes by default.

Specifying volume locations will generally be used for larger production deployments.

By default podman stores its volumes in /var/lib/containers/storage/volumes/.

You can provide custom storage for the volumes by mounting disks on this path or the expected volume path inside it such as: /var/lib/containers/storage/volumes/var-spacewalk. This is especially important for the database and package mirrors.

For a list of all persistent volumes in the container, see:

- **Installation and Upgrade Guide › Container Management › Persistent Container Volumes**
- **Administration Guide › Troubleshooting › Tshoot Container Full Disk**

4.1.2. Uyuni Server Air-gapped Deployment

4.1.2.1. What is air-gapped deployment?

Air-gapped deployment refers to the setup and operation of any networked system that is physically isolated from insecure networks, especially the internet. This type of deployment is commonly used in high-security environments such as military installations, financial systems, critical infrastructure, and anywhere sensitive data is handled and must be protected from external threats.

You can easily pull container images using Podman or Docker on a machine with internet access.

Procedure: Pulling the images

1. Pull the desired images, then save the images as a tar archive. For example:

Listing 1. Podman

```
podman pull registry.opensuse.org/uyuni/server:latest
registry.opensuse.org/uyuni/server-postgresql:latest
podman save -m --output images.tar registry.opensuse.org/uyuni/server:latest
registry.opensuse.org/uyuni/server-postgresql:latest
```

Listing 2. Docker

```
docker pull registry.opensuse.org/uyuni/server:latest
registry.opensuse.org/uyuni/server-postgresql:latest
docker save --output images.tar registry.opensuse.org/uyuni/server:latest
registry.opensuse.org/uyuni/server-postgresql:latest
```

2. Transfer the resulting images.tar to the Server container host and load it using the following command:

Listing 3. Load the server image

```
podman load -i images.tar
```

4.1.2.2. Obtaining container images for Salt formulas in air-gapped environments

Some formulas, like Bind and DHCP (Kea), also use containers. If you plan to use them in an air-gapped environment, you need to pull their images, save them to an archive, and load them on your Uyuni Proxy or another managed system.

The images are available from registry.opensuse.org.

Procedure: Obtaining formula images for air-gapped environments

1. On a system with Internet access, pull the required images. For example:

```
podman pull registry.opensuse.org/opensuse/bind:latest
podman pull registry.opensuse.org/opensuse/kea:latest
```

2. Save the images to a TAR archive:

```
podman save -o formula-images.tar registry.opensuse.org/opensuse/bind:latest
registry.opensuse.org/opensuse/kea:latest
```

3. Transfer the formula-images.tar file to your air-gapped system.
4. Load the images on the air-gapped system:

```
podman load -i formula-images.tar
```

4.1.2.2.1. Deploy Uyuni on openSUSE Tumbleweed

Uyuni also provides all the needed container images in RPM packages that can be installed on the system.



User should make the needed RPM available on the internal network. That can be done by using a second Uyuni Server or any kind of mirror.

Procedure: Install Uyuni on openSUSE Tumbleweed in air-gapped environment

1. Install openSUSE Tumbleweed.
2. Update the system.
3. Install tools packages and image packages (replace $ARCH$ with the correct architecture):

```
zypper install mgradm* mgrctl* uyuni-server*-image* uyuni-server-proxy-tftpd-image*
```

The TFTPD image is now the same than on the proxy. If TFTPD is disabled on the server this image is not needed.

+

1. Deploy Uyuni with mgradm. In an Air-gapped environment you may want to use the option --pullPolicy Never.

For more detailed information about installing Uyuni Server on openSUSE Tumbleweed, see Server Deployment.

To upgrade Uyuni Server, apply available host updates with transactional-update, reboot if updates were applied, then run mgradm upgrade. For air-gapped installations, upgrade the container RPM packages before running mgradm upgrade.

4.2. Install Uyuni Proxy

There are various scenarios to deploy a Uyuni Proxy. All these scenarios presume you have already successfully deployed a Uyuni 2026.11 Server.

4.2.1. Uyuni Proxy Deployment on openSUSE Tumbleweed

This guide outlines the deployment process for the Uyuni 2026.11 Proxy. This guide presumes you have already successfully deployed a Uyuni 2026.11 Server. To successfully deploy, you will perform the following actions:

Checklist: Proxy Deployment

1. Review hardware requirements.
2. Install openSUSE Tumbleweed on a bare-metal machine.
3. Bootstrap the Proxy as a Salt minion.
4. Generate a Proxy configuration.
5. Transfer the Proxy configuration from Server to Proxy
6. Use the Proxy configuration to register the Salt minion as a Proxy with Uyuni.



Supported operating system for the Proxy Container Host

The supported operating system for the container host is openSUSE Tumbleweed.

Container host

A container host is a server equipped with a container engine like Podman, which lets it manage and deploy containers. These containers hold applications and their essential parts, such as libraries, but not a full operating system, making them lightweight. This setup ensures applications run the same way in different environments. The container host supplies the necessary resources such as CPU, memory, and storage for these containers.

4.2.1.1. Hardware requirements for the Proxy

This table shows the hardware requirements for deploying Uyuni Proxy.

Table 9. Proxy Hardware Requirements

Hardware	Details	Recommendation
CPU	x86-64, aarch64	Minimum 2 dedicated 64-bit CPU cores
RAM	Minimum	2 GB
	Recommended	8 GB
Disk Space	/ (root directory)	Minimum 40 GB

Hardware	Details	Recommendation
	/var/lib/containers/storage/volumes	Minimum 100 GB, Storage requirements should be calculated for the number of ISO distribution images, containers, and bootstrap repositories you will use.

4.2.1.2. Container host general requirements

For general requirements, see **Installation and Upgrade Guide › General requirements**.

An openSUSE Tumbleweed server should be installed from installation media. This procedure is described below.

4.2.1.3. Container host requirements

For CPU, RAM, and storage requirements, see **Installation and Upgrade Guide › Hardware requirements**.



To guarantee that clients can resolve the FQDN domain name, both the containerized server and the host machines must be linked to a functional DNS server. Additionally, it is essential to ensure correct configuration of reverse lookups.

4.2.1.4. Configure custom persistent storage

This step is optional. However, if custom persistent storage is required for your infrastructure, use the mgr-storage-proxy tool.

- For more information, see mgr-storage-proxy --help. This tool simplifies creating the container storage and Squid cache volumes.

Use the command in the following manner:

```
mgr-storage-proxy <storage-disk-device>
```

For example:

```
mgr-storage-proxy /dev/nvme1n1
```



This command will create the persistent storage volumes at /var/lib/containers/storage/volumes.

For more information, see

- **Installation and Upgrade Guide › Container Management › Persistent Container Volumes**
- **Administration Guide › Troubleshooting › Tshoot Container Full Disk**

4.2.1.5. Bootstrap the proxy host as a minion

Task: Bootstrap the Proxy Host

1. Select **Systems › Bootstrapping**.
2. Fill in the fields for your Proxy host.
3. Select the Activation key created in the previous step from the dropdown.
4. Click **[+ Bootstrap]**.
5. Wait for the Bootstrap process to complete successfully. Check the **Salt** menu and confirm the Salt minion key is listed and accepted.
6. Reboot the Proxy host.
7. Select the host from the **System** list and trigger a second reboot after all events are finished to conclude the onboarding.

Task: Update the Proxy Host

1. Select the host from the **Systems** list and apply all patches to update it.
2. Reboot the Proxy host.

4.2.1.6. Generate the proxy configuration

The configuration archive of the Uyuni Proxy is generated by the Uyuni Server. Each additional Proxy requires its own configuration archive.



The container host for the Uyuni Proxy must be registered as a salt minion to the Uyuni Server prior to generating this Proxy configuration.

You will perform the following tasks:

Procedure: Proxy deployment tasks

1. Generate a Proxy configuration file.
2. Transfer the configuration to the Proxy.

3. Start the Proxy with the mgrpxy command.

Task: Generating a Proxy Container Configuration using Web UI

1. In the Web UI, navigate to **Systems › Proxy Configuration** and fill the required data:
2. In the Proxy FQDN field type fully qualified domain name for the proxy.
3. In the Parent FQDN field type fully qualified domain name for the Uyuni Server or another Uyuni Proxy.
4. In the Proxy SSH port field type SSH port on which SSH service is listening on Uyuni Proxy. Recommended is to keep default 8022.
5. In the Max Squid cache size [MB] field type maximal allowed size for Squid cache. Typically this should be at most 60% of available storage for the containers.
6. In the SSL certificate selection list choose if new server certificate should be generated for Uyuni Proxy or an existing one should be used. You can consider generated certificates as Uyuni builtin (self signed) certificates.

   Depending on the choice then provide either path to signing CA certificate to generate a new certificate or path to an existing certificate and its key to be used as proxy certificate.

   The CA certificates generated on the server are stored in the /var/lib/containers/storage/volumes/root/_data/ssl-build directory.

   For more information about existing or custom certificates and the concept of corporate and intermediate certificates, see **Administration Guide › Ssl Certs Imported**.

7. Click **[Generate]** to register new proxy FQDN in Uyuni Server and generate configuration archive with details for container host.
8. After a few moments you are presented with file to download. Save this file locally.

4.2.1.7. Transfer the proxy configuration

The Web UI generates a configuration archive. This archive needs to be made available on the Proxy container host.

Task: Copy the Proxy configuration

1. Copy the files to the Proxy host:

```
scp config.tar.gz <proxy-FQDN>:/root
```

2. Install the Proxy with:

```
mgrpxy install config.tar.gz
```

4.2.1.8. Start the Uyuni 2026.11 proxy

Container can now be started with the mgrpxy command:

Task: Start and Check Proxy Status

1. Start the Proxy by calling:

```
mgrpxy start
```

2. Check container status by calling:

```
mgrpxy status
```

Five Uyuni Proxy containers should be present and should be part of the proxy-pod container pod:

- proxy-salt-broker
- proxy-httpd
- proxy-tftpd
- proxy-squid
- proxy-ssh

4.2.1.9. Installing Uyuni tools for use with containers

Procedure: Installing Uyuni tools on openSUSE Tumbleweed

1. Go to **System Details › Software › Packages › Install**.
2. Install the uyuni-tools package.

For more information on the Uyuni Container Utilities, see Uyuni Container Utilities.

4.2.1.9.1. Using a custom container image for a service

By default, the Uyuni Proxy suite is set to use the same image version and registry path for each of its services. However, it is possible to override the default values for a specific service using the install parameters ending with -tag and -image.

For example, use it like this:

```
mgrpxy install --httpd-tag 0.1.0 --httpd-image registry.opensuse.org/uyuni/proxy-httpd
/path/to/config.tar.gz
```

It adjusts the configuration file for the httpd service, where registry.opensuse.org/uyuni/proxy-httpds is the image to use and 0.1.0 is the version tag, before restarting it.

To reset the values to defaults, run the install command again without those parameters:

```
mgrpxy install /path/to/config.tar.gz
```

This command first resets the configuration of all services to the global defaults and then reloads it.

4.2.2. Containerized Uyuni Proxy Setup

Once container host for Uyuni Proxy containers is prepared, setup of containers require few additional steps to finish configuration.

Procedure

1. Generate Uyuni Proxy configuration archive file
2. Transfer configuration archive to the container host prepared in installation step and extract it
3. Start the proxy sevices with mgrpxy

4.2.2.1. Generate Proxy Configuration

The configuration archive of the Uyuni Proxy is generated by the Uyuni Server. Each additional Proxy requires its own configuration archive.

For the containerized Uyuni Proxy, you must build a new proxy configuration file and then redeploy the container for the changes to take effect. This is the process for updating settings, including the SSL certificate.



For Podman deployment, the container host for the Uyuni Proxy must be registered as a client to the Uyuni Server prior to generating this proxy configuration.

If a proxy FQDN is used to generate a proxy container configuration that is not a registered client (as in the Kubernetes use case), a new system entry will appear in system list. This new entry will be shown under previously entered Proxy FQDN value and will be of Foreign system type.



Peripheral servers are always using third-party SSL certificates. If the hub server has generated the certificates for the peripheral server, it needs to generate the certificate of each proxy too.

On the hub server, run the following command.

```
mgrctl exec -ti -- rhn-ssl-tool --gen-server --dir="/root/ssl-build"
--set-country="COUNTRY" \
  --set-state="STATE" --set-city="CITY" --set-org="ORGANIZATION" \
  --set-org-unit="ORGANIZATION UNIT" --set-email="name@example.com" \
  --set-hostname=PROXY --set-cname="proxy.example.com"
```

The files to use will be

1. /root/ssl-build/RHN-ORG-TRUSTED-SSL-CERT as the root CA,
2. /root/ssl-build/<hostname>/server.crt as the proxy certificate and
3. /root/ssl-build/<hostname>/server.key as the proxy certificate's key.

4.2.2.1.1. Generate the Proxy Configuration with Web UI

Procedure: Generating a Proxy Container Configuration Using Web UI

1. In the Web UI, navigate to **Systems › Proxy Configuration** and fill the required data:
2. In the Proxy FQDN field type fully qualified domain name for the proxy.
3. In the Parent FQDN field type fully qualified domain name for the Uyuni Server or another Uyuni Proxy.
4. In the Proxy SSH port field type SSH port on which SSH service is listening on Uyuni Proxy. Recommended is to keep default 8022.
5. In the Max Squid cache size [MB] field type maximal allowed size for Squid cache. Recommended is to use at most 80% of available storage for the containers.

   2 GB represents the default proxy squid cache size. This will need to be adjusted for your environment.

6. In the SSL certificate selection list choose if a new server certificate should be generated for Uyuni Proxy, an existing one should be used, or the SSL part should be skipped entirely. You can consider generated certificates as Uyuni builtin (self signed) certificates. If Uyuni server runs on Kubernetes, the generated certificate option is not possible and replaced with no SSL certificate as they are managed outside the containers.

Depending on the choice then provide either path to signing CA certificate to generate a new certificate or path to an existing certificate and its key to be used as proxy certificate. If the SSL part is skipped, the resulting configuration archive does not carry server_crt, server_key or ca_crt; the deployment is then responsible for providing the proxy certificate and root CA out-of-band (for Kubernetes deployments this means creating proxy-cert and uyuni-ca manually — see the TLS setup section of the Kubernetes proxy deployment page).

The CA certificates generated by the server are stored in the /var/lib/containers/storage/volumes/root/_data/ssl-build directory.

For more information about existing or custom certificates and the concept of corporate and intermediate certificates, see **Administration Guide › Ssl Certs Imported**.

7. Click **[Generate]** to register a new proxy FQDN in the Uyuni Server and generate a configuration archive (config.tar.gz) containing details for the container host.
8. After a few moments you are presented with file to download. Save this file locally.

4.2.2.1.2. Generate Proxy Configuration With spacecmd and Self-Signed Certificate

You can generate a Proxy configuration using spacecmd. This is only possible if Uyuni server runs on podman and has a self-signed root CA certificate.

Procedure: Generating Proxy Configuration with spacecmd and Self-Signed Certificate

1. SSH into your container host.
2. Execute the following command replacing the Server and Proxy FQDN:

```
mgrctl exec -ti 'spacecmd proxy_container_config_generate_cert -- dev-
pxy.example.com dev-srv.example.com 2048 email@example.com -o
```

```
/tmp/config.tar.gz'
```

3. Copy the generated configuration from the server container:

```
mgrctl cp server:/tmp/config.tar.gz .
```

4.2.2.1.3. Generate Proxy Configuration With spacecmd and Custom Certificate

You can generate a Proxy configuration using spacecmd for custom certificates rather than the default self-signed certificates.

Procedure: Generating Proxy Configuration with spacecmd and Custom Certificate

1. SSH into your Server container host.
2. Execute the following commands, replacing the Server and Proxy FQDN:

```
for f in ca.crt proxy.crt proxy.key; do
  mgrctl cp $f server:/tmp/$f
done
mgrctl exec -ti 'spacecmd proxy_container_config -- -p 8022 pxy.example.com
srv.example.com 2048 email@example.com /tmp/ca.crt /tmp/proxy.crt
/tmp/proxy.key -o /tmp/config.tar.gz'
```

3. If your setup uses an intermediate CA, copy it as well and include it in the command with the -i option (can be provided multiple times if needed) :

```
mgrctl cp intermediateCA.pem server:/tmp/intermediateCA.pem
mgrctl exec -ti 'spacecmd proxy_container_config -- -p 8022 -i
/tmp/intermediateCA.pem pxy.example.com srv.example.com 2048 email@example.com
/tmp/ca.crt /tmp/proxy.crt /tmp/proxy.key -o /tmp/config.tar.gz'
```

4. Copy the generated configuration from the server container:

```
mgrctl cp server:/tmp/config.tar.gz .
```

4.2.2.1.4. Generate Proxy Configuration With spacecmd and no Certificate

You can generate a Proxy configuration using spacecmd with no TLS certificates. This is needed for Uyuni running on Kubernetes as the certificates are handled outside of the containers.

Procedure: Generating Proxy Configuration with spacecmd and no Certificate

1. SSH into your Server container host.
2. Execute the following commands, replacing the Server and Proxy FQDN:

```
for f in ca.crt proxy.crt proxy.key; do
  mgrctl cp $f server:/tmp/$f
done
mgrctl exec -ti 'spacecmd proxy_container_config_nossl -- -p 8022
pxy.example.com srv.example.com 2048 email@example.com -o /tmp/config.tar.gz'
```

3. Copy the generated configuration from the server container:

```
mgrctl cp server:/tmp/config.tar.gz .
```

4.2.2.2. Transfer Uyuni Proxy Configuration

Both spacecmd command and generating via Web UI ways create a configuration archive. This archive needs to be made available on container host. Transfer this generated archive to the container host.

4.2.2.3. Start Uyuni Proxy Containers

Container can be started with the mgrpxy command.

Procedure: Start Uyuni Proxy Containers

1. Run command:

```
mgrpxy start uyuni-proxy-pod
```

2. Check if all containers started up as expected by calling:

```
podman ps
```

Five Uyuni Proxy containers should be present and should be part of proxy-pod container pod.

- proxy-salt-broker
- proxy-httpd
- proxy-tftpd
- proxy-squid
- proxy-ssh

4.2.3. Proxy conversion from client

4.2.3.1. Overview

This chapter describes how to convert a client system into a Uyuni Proxy using the Web UI.

It assumes that the proxy host system has already been bootstrapped and subscribed to the base operating system channel.

For information about client onboarding, see **Client Configuration Guide › Registration Overview**.

4.2.3.2. Requirements

Before starting the conversion, ensure the following requirements are fulfilled.

4.2.3.2.1. Client Must Be

- Already onboarded in Uyuni
- Reachable via the network

4.2.3.3. Preparation

Before proceeding with the proxy conversion, make sure the following preparations are completed to avoid interruptions during the conversion process.

4.2.3.3.1. SSL Certificates

Valid SSL certificates are required to secure communication between the proxy and other components.

You need:

- The public certificate of the Certificate Authority (CA) that signed the certificate on the Uyuni server
- A certificate for the proxy.
- The corresponding private key for the proxy certificate.



If your CA uses an intermediate certificate chain, you must include all intermediate certificates as well.

If you are not using third party certificates, you can generate them using the rhn-ssl-tool inside the Uyuni container.

Generate a proxy certificate

1. On the Uyuni server host, run:

```
mgrctl exec -ti -- rhn-ssl-tool --gen-server \
  --set-hostname="<PROXY-FQDN>" \
  --dir="/root/ssl-build"
```

   For more information about other parameters, see **Administration Guide › Ssl Certs Selfsigned**.

2. Transfer the certificates to Uyuni server host

```
mgrctl cp server:/root/ssl-build/<PROXY-FQDN>/server.crt /root/proxycert.pem
mgrctl cp server:/root/ssl-build/<PROXY-FQDN>/server.key /root/proxykey.pem
mgrctl cp server:/root/ssl-build/RHN-ORG-TRUSTED-SSL-CERT /root/rootca.pem
```



To confirm the exact folder where the certificates and key files were generated, you can list the directories with:

```
mgrctl exec -ti -- ls -ltd /root/ssl-build/*/
```

3. Transfer the certificates from Uyuni server host

```
scp <UYUNI-FQDN>:/root/proxycert.pem ./
scp <UYUNI-FQDN>:/root/proxykey.pem ./
scp <UYUNI-FQDN>:/root/rootca.pem ./
```

4.2.3.3.2. Packages Preparation

Install mgrpxy

The mgrpxy tool must be installed from a repository matching your system. Choose the appropriate repository from:

https://download.opensuse.org/repositories/systemsmanagement:/Uyuni:/Stable:/ContainerUtils/

Listing 4. Example openSUSE Tumbleweed installation:

```
zypper ar
https://download.opensuse.org/repositories/systemsmanagement:/Uyuni:/Stable:/ContainerUtil
s/openSUSE_Tumbleweed/ uyuni-containerutils
zypper ref
zypper in mgrpxy
```

Install Container Images

It is recommended to deploy the container images as RPM packages. Please ensure the following packages are installed on the client:

```
zypper ar
https://download.opensuse.org/repositories/systemsmanagement:/Uyuni:/Stable/containerfile/
uyuni-proxy-images
zypper ref
zypper in uyuni-proxy-httpd-image \
          uyuni-proxy-salt-broker-image \
          uyuni-proxy-squid-image \
          uyuni-proxy-ssh-image \
          uyuni-proxy-tftpd-image
```

For details on air-gapped deployment, see **Installation and Upgrade Guide › Container Deployment › Proxy Air Gapped Deployment Mlm**

4.2.3.4. Setup Proxy Client

1. Navigate to the client's Overview page.
2. Click button **[Convert to Proxy]**.

   Confirm you were redirected to the proxy configuration form.

   This page can be accessed later from the Details > Proxy > Configuration tab.

3. In the Web UI, navigate to **Proxy › Configuration** and fill in the required data:

   ### Procedure: Configuring the Proxy

   a. In the Parent FQDN field, type the fully qualified domain name for the parent server or proxy.

   b. In the Proxy SSH port field, type the SSH port on which the SSH service is listening on the Uyuni Proxy. It is recommended to keep the default: 8022.

   c. In the Max Squid cache size field, type the maximum allowed size for the Squid cache, in Gigabytes.

   d. In the Proxy admin email field, type the administrator's email address.

   e. In the Certificates section, provide the certificates for the Uyuni Proxy, obtained in the preparation step.

   f. In the Source section, select one of the two options: RPM or Registry.

   - The RPM option is recommended for air-gapped or restricted environments.
   - The Registry option can be used if connectivity to the container image registry is available.
     If selected, you will be prompted to choose between two sub-options: Simple or Advanced.

- If Simple is selected, provide values in the Registry URL and Containers Tag fields.
  - For Registry URL use: registry.opensuse.org/uyuni.
  - Select the tag from the drop-down list.
- If Advanced is selected, an additional section of the form is shown:
  - For each individual container URL field, use the registry: registry.opensuse.org/uyuni followed by the corresponding suffix, for example, proxy-httpd or salt-broker.
  - Select the tag from the drop-down list.

4. Once all fields are filled, click **[Apply]** to apply the configuration and schedule the proxy installation task.

4.2.3.5. Verify Proxy Activation

Check the client's event history to confirm task success.

(Optional) Access the proxy's HTTP endpoint to validate it shows a welcome page.

Chapter 5. Upgrade and Migration

5.1. Server

5.1.1. Server Upgrade

Before running the upgrade command, it is required to update the Uyuni container tooling on the host, such as the mgradm tool.



Before upgrading, verify that every root and intermediate CA certificate used by Uyuni marks X509v3 Basic Constraints as critical and includes CA:TRUE. Strict certificate validators, including Python 3.13 and later, reject CA certificates that miss the critical flag.

If your installation uses an older self-signed CA generated by Uyuni, you might need to create a new CA and new server certificates, then deploy the new root CA to all clients.

You can rotate the CA certificate on a Podman installation using the new mgradm ssl command. For more information, see **Administration Guide › Ssl Certs Ca Migration**.

Note that the command can only be used **after** upgrading both mgradm and Uyuni. You can safely upgrade affected installations as long as you rotate the CA certificates and ensure clients recognize them before starting the server.

While failing to rotate the CA certificate may not cause an immediate system failure, it is strongly discouraged.

Procedure: Upgrading Server

1. Refresh software repositories with zypper:

```
zypper ref
```

2. Update the Uyuni container tooling:

```
zypper up mgradm mgrctl mgradm-bash-completion mgrctl-bash-completion uyuni-
storage-setup-server
```

3. The Uyuni Server container can be updated using the following command:

```
mgradm upgrade
```

This command will bring the status of the container up-to-date and restart

the server.

4. Clean up the unused container images to free disk space:

```
podman image prune -a
```



In some upgrades, multiple *.rpmnew and *.rpmsave files may be generated. The presence or number of these files does not indicate that manual action is required. In Uyuni container environments, required configuration changes are applied automatically during the upgrade process. These files are created as a result of differences between packaged defaults and existing configuration files, and may also include internal or informational changes that are not intended to be merged. If manual action is required for a configuration change, it will be explicitly documented in the release notes or upgrade documentation.

Do not treat these files as a post-upgrade checklist or merge them in bulk. Only review a file if you are actively troubleshooting a specific issue and understand the impact of the configuration it contains. If a change is required, it should be applied intentionally based on a known requirement, not by copying differences from these files. If you are unsure, leave the file unchanged.

Upgrading with third-party SSL certificate



In previous versions, the database container needed to have an SSL certificate with the following Subject Alternate Names (SANs):

- db
- reportdb

This is no longer required. The database container needs only the externally facing fully qualified domain name. The old certificates with db and reportdb SANs can be still used.

The same certificate can be used for both the main container and the database one.

In order to pass the new certificates to the upgrade command, use the --ssl-db-ca-root, --ssl-db-cert and --ssl-db-key parameters.



Upgrading to specific version

If you do not specify the tag parameter, it will default to upgrading to the most recent version. To upgrade to a specific version, provide the tag parameter with the desired image tag.

For more information on the upgrade command and its parameters, use the following command:

```
mgradm upgrade -h
```

For air-gapped installations, first upgrade the container RPM packages, then run the mgradm command.

5.1.1.1. Database Backup Volume

During a PostgreSQL database major version upgrade , mgradm migration or mgradm upgrade create a backup copy of the existing database.

For this purpose mgradm creates a backup directory named _data_old directly inside the database volume path at /var/lib/containers/storage/volumes/var-pgsql/_data_old.



Because the _data_old directory is created on the same filesystem as the active database volume, it cannot be relocated to a separate storage volume during the upgrade process. Ensure that the storage volume hosting /var/lib/containers/storage/volumes/var-pgsql/ has at least twice the free space of your current database size, plus an extra 5–10% needed for running tasks during the upgrade, before starting mgradm migration or mgradm upgrade.

Once the migration or upgrade is complete and you have validated that the upgraded system is working as expected, you can safely delete the _data_old directory to reclaim disk space.

5.1.2. Migrating the Uyuni Server to openSUSE Tumbleweed

This page describes a simple, backup-and-restore migration of a Uyuni Server running on openSUSE Leap Micro 5.5 to a fresh host running openSUSE Tumbleweed as the base OS.

5.1.2.1. Overview of the Migration Process

You will:

- Create a full server backup with mgradm backup on the openSUSE Leap Micro 5.5 host.
- Reinstall the host with openSUSE Tumbleweed (server profile).
- Install Uyuni tools and prerequisites on Tumbleweed.
- Restore the backup with mgradm backup restore.
- Start services and verify the server.

5.1.2.2. Requirements and Considerations

- Source server: openSUSE Leap Micro 5.5 running Uyuni (for example: 2026.11).
- Target server: openSUSE Tumbleweed with the same hostname/FQDN and IP (recommended) to avoid client-side changes.
- SSH/scp access between machines for transferring the backup tarball.
- Sufficient free disk space on both source and target for the backup and restore.



Restore to the same Uyuni version you backed up, or a version explicitly documented as compatible for restore. If you use development or preview repositories (for example, Uyuni Master), expect changes and re-validate.

5.1.2.3. Migration Procedure

5.1.2.3.1. Step 1: Create a Backup on the openSUSE Leap Micro 5.5 Server

Procedure: Create a Backup

1. As root on the old server, create a backup directory and run the backup:

```
mgradm backup /tmp/uyuni-backup
```

2. Package the backup for transfer:

```
tar -C /tmp -cvf /tmp/uyuni-backup.tar uyuni-backup
```

3. Copy the backup to a safe location you can reach from the new host:

```
scp /tmp/uyuni-backup.tar <USER>@<HOST>:/path/to/store/
```



You can store the backup to external storage or an object store as long as you can fetch it on the new host.

5.1.2.3.2. Step 2: Reinstall the Host with openSUSE Tumbleweed

Procedure: Reinstalling the Host

1. Reprovision the VM or bare-metal host with openSUSE Tumbleweed.
2. Choose a basic “server profile” installation.
3. Set the same hostname/FQDN and IP address as the original server if you want clients to reconnect seamlessly.

5.1.2.3.3. Step 3: Install Uyuni Tools and Prerequisites on Tumbleweed

Procedure: Installing Tools and Prerequisites

1. Add the Uyuni Stable repository and install tools:

```
zypper ar
https://download.opensuse.org/repositories/systemsmanagement:/Uyuni:/Stable/ima
ges/repo/Uyuni-Server-POOL-x86_64-Media1 uyuni-server-stable
zypper ref
zypper in mgradm mgrctl mgradm-bash-completion mgrctl-bash-completion uyuni-
storage-setup-server
```

2. Install Podman if it was not automatically pulled in:

```
zypper in podman
```



The package uyuni-storage-setup-server provides the mgr-storage-server tool for preparing persistent volumes. Installing podman explicitly may be necessary on some installations.

5.1.2.3.4. Step 4: Optional - Prepare Persistent Storage

Procedure: Prepareing Persistent Storage

It is recommended to configure persistent storage with mgr-storage-server to avoid container full-disk issues.

```
mgr-storage-server <storage-disk-device> [<database-disk-device>]
```



Devices must be raw (no existing filesystem). The tool creates volumes at /var/lib/containers/storage/volumes.

For details, see:

- **Installation and Upgrade Guide › Container Management › Persistent Container Volumes**
- **Administration Guide › Troubleshooting › Tshoot Container Full Disk**

5.1.2.3.5. Step 5 Fetch and Restore the Backup on Tumbleweed

Procedure: Fetching and Restoring the Backup

1. Copy the backup to the new server and unpack it:

```
scp <USER>@<HOST>:/path/to/store/uyuni-backup.tar /tmp/
tar -C /tmp -xvf /tmp/uyuni-backup.tar
```

2. Restore using mgradm (point to the extracted backup directory):

```
mgradm backup restore /tmp/uyuni-backup
```

5.1.2.3.6. Step 6: Start Services and Verify

Procedure: Starting Services and Verifying

1. Start the server services:

```
mgradm start
```

2. Verify:
    - Check that all containers are up: mgrctl ps or podman ps.
    - Access the Web UI (HTTPS) and log in.
    - Review logs for errors: mgrctl logs server and other components as needed.

5.1.2.4. Notes and Troubleshooting

- If Podman wasn't installed automatically, install it with zypper in podman and rerun the restore/start steps.
- Ensure the target host has the same time, hostname, and IP configuration expected by your setup (especially if clients exist).
- For large environments, ensure adequate disk throughput and space. The backup and restore can take a long time.



If the restore fails or the new system cannot start, you can still boot the original openSUSE Leap Micro 5.5 system and continue service. Keep the original VM/snapshots until you fully validate the new Tumbleweed-based server.

5.2. Proxy

5.2.1. Proxy Upgrade

Before running the upgrade command, it is required to update the Uyuni container tooling on the host, such as the mgrpxy tool.

Procedure: Upgrading Proxy

1. Refresh software repositories with zypper:

```
zypper ref
```

2. Update the Uyuni container tooling:

```
zypper up mgrpxy mgrctl mgrpxy-bash-completion mgrctl-bash-completion
```

3. The Uyuni Proxy containers running on podman can be updated using the following command:

```
mgrpxy upgrade
```

For proxies running on a Kubernetes cluster, see **Specialized Guides › Kubernetes Guide › Proxy Kubernetes Upgrade**.

4. Clean up the unused container images to free disk space:

```
podman image prune -a
```

> In some upgrades, multiple *.rpmnew and *.rpmsave files may be generated. The presence or number of these files does not indicate that manual action is required. In Uyuni container environments, required configuration changes are applied automatically during the upgrade process. These files are created as a result of differences between packaged defaults and existing configuration files, and may also include internal or informational changes that are not intended to be merged. If manual action is required for a configuration change, it will be explicitly documented in the release notes or upgrade documentation.
>
> Do not treat these files as a post-upgrade checklist or merge them in bulk. Only review a file if you are actively troubleshooting a specific issue and understand the impact of the configuration it contains. If a change is required, it should be applied intentionally based on a known requirement, not by copying differences from these files. If you are unsure, leave the file unchanged.

If you do not specify the tag parameter when upgrading to specific version, it will default to upgrading to the most recent version. To upgrade to a specific version, provide the tag parameter with the desired image tag.

We highly recommend using the same tag for all proxy containers to ensure consistency under normal circumstances.

For air-gapped installations, first upgrade the container RPM packages, then run the mgrpxy upgrade command.

5.2.2. Migrating the Uyuni Proxy to openSUSE Tumbleweed

This page describes how to migrate a Uyuni Proxy host from openSUSE Leap Micro 5.5 to a fresh openSUSE Tumbleweed installation using the proxy administration tool mgrpxy.



This guide was tested on Tumbleweed only. There is no known reason it wouldn't work on other supported bases, but always validate in a test environment before production.

5.2.2.1. Overview of the Proxy Migration Process

You will:

- Save proxy configuration from the old system (including Apache/Squid tuning).
- Reinstall the host with openSUSE Tumbleweed.
- Re-register the host using the system reactivation key.
- Install mgrpxy (and Podman if needed).
- Restore configuration and run mgrpxy install with optional tuning files.

5.2.2.2. Requirements and Considerations

- Keep the same hostname/FQDN and IP when possible so the server and clients interact with the proxy as before.
- Ensure you have the "system reactivation key" for the existing proxy system (UI: Systems > select the proxy > Details > Reactivation).
- Ensure SSH/scp access to move configuration archives off and onto the machine.

5.2.2.3. Migration Procedure

5.2.2.3.1. Step 1: Save Proxy Configuration and Tuning Files

Procedure: Save Proxy Configuration and Tuning Files

1. Copy the Uyuni proxy configuration directory to a safe location:

```
scp -r /etc/uyuni <USER>@<HOST>:/some/where/safe/
```

2. Identify Apache and Squid tuning files currently in use by the legacy proxy services:

```
systemctl cat uyuni-proxy-httpd.service | grep EXTRA_CONF= | sed 's/.*=-
v\([^:]\+\):.*/\1/'
systemctl cat uyuni-proxy-squid.service | grep EXTRA_CONF= | sed 's/.*=-
v\([^:]\+\):.*/\1/'
```

3. Copy those tuning files to the same safe location as well.



Typical default paths after you copy them back will be:

- Apache tuning: /etc/uyuni/proxy/apache.conf
- Squid tuning: /etc/uyuni/proxy/squid.conf

5.2.2.3.2. Step 2: Reinstall the Host with openSUSE Tumbleweed

Procedure: Reinstalling the Host with openSUSE Tumbleweed

1. Reinstall the machine with openSUSE Tumbleweed (server profile recommended).
2. Set the same hostname/FQDN and IP as before when possible.

5.2.2.3.3. Step 3: Re-register the Host with the Reactivation Key

Procedure: Re-registering the Host with the Reactivation Key

1. From the Uyuni Web UI, obtain the system reactivation key for the existing proxy system record (Systems > Details > Reactivation).
2. Bootstrap/re-register the Tumbleweed host using that reactivation key so it claims the existing system entry.



Use your standard bootstrapping process for Tumbleweed hosts in your environment (for example, the bootstrap script or your configuration management), ensuring the reactivation key is applied.

5.2.2.3.4. Step 4: Install Uyuni Proxy Tools and Podman

Procedure: Installing Proxy Tools and Podman

1. Add the Uyuni Stable repository and install tools:

```
zypper ar
https://download.opensuse.org/repositories/systemsmanagement:/Uyuni:/Stable/ima
ges/repo/Uyuni-Proxy-POOL-x86_64-Media1 uyuni-proxy-stable
zypper ref
zypper in mgrpxy mgrctl mgrpxy-bash-completion mgrctl-bash-completion
```

2. Ensure Podman is installed (required to run containers):

```
zypper in podman
```

5.2.2.3.5. Step 5: Restore Configuration and Install the Proxy

Procedure: Restoring Configuration and Install the Proxy

1. Copy back the saved configuration directory to the new host:

```
scp -r <USER>@<HOST>:/some/where/safe/uyuni /etc/
```

2. If you saved Apache/Squid tuning files, place them at the expected default paths or note their locations for parameters in the next command:

```
# Default paths expected by mgrpxy parameters (adjust/move your files
accordingly)
# Apache tuning: /etc/uyuni/proxy/apache.conf
# Squid tuning:  /etc/uyuni/proxy/squid.conf
```

3. Run the proxy installation with Podman. If you do not use tuning files, omit the corresponding parameters:

```
# With tuning files
mgrpxy install \
    --tuning-httpd /etc/uyuni/proxy/apache.conf \
    --tuning-squid /etc/uyuni/proxy/squid.conf

# If you have no tuning files, remove the tuning parameters:
# mgrpxy install
```



In an upcoming release, if tuning files are placed at the default

paths noted above, the explicit parameters will not be required.

5.2.2.3.6. Step 6: Verify the Proxy

Procedure: Verifying the Proxy

1. Check containers are running:

```
mgrctl ps
# or
podman ps
```

2. Confirm the proxy appears healthy in the Uyuni Web UI and that clients using this proxy operate normally.

5.2.2.4. Troubleshooting

- If Podman was missing, install it and rerun the mgrpxy install step.
- Verify the host's time, hostname, and IP match expectations.
- If the host did not reattach to the existing system record, confirm you used the correct reactivation key and repeat the bootstrap.

5.2.3. Legacy Proxy Migration to Container

The containerized proxy now is managed by a set of systemd services. For managing the containerized proxy, use the mgrpxy tool.

This section will help you migrate from the legacy systemd proxy using the mgrpxy tool.



An in-place migration from previous releases of Uyuni to 2026.11 will remain unsupported due to the HostOS change from openSUSE Leap to openSUSE Leap Micro.

The traditional contact protocol is no longer supported in Uyuni 2026.11 and later. Before migrating from previous Uyuni releases to 2026.11, any existing traditional clients including the traditional proxies must be migrated to Salt.

5.2.3.1. Migrate From Legacy to Containerized Proxy With Systemd

5.2.3.1.1. Generate Proxy Configuration

Procedure: Generate the Proxy Configuration

1. Log in to the Uyuni Server Web UI.
2. Select **Systems › Proxy Configuration** from the left navigation.
3. Enter your Proxy FQDN. Use the same FQDN as the original proxy host.
4. Enter your Server FQDN.
5. Enter the Proxy port number. We recommend using the default port of 8022.
6. Certificate and private key are located on the Server container host in /var/lib/containers/storage/volumes/root/_data/ssl-build/.
    - RHN-ORG-TRUSTED-SSL-CERT
    - RHN-ORG-PRIVATE-SSL-KEY
7. Copy the certificate and key to your machine with:

```
scp root@uyuni-server-example.com:/root/ssl-build/RHN-ORG-PRIVATE-SSL-KEY .
scp root@uyuni-server-example.com:/root/ssl-build/RHN-ORG-TRUSTED-SSL-CERT .
```

8. Select **[Choose File]** and browse your local machine for the certificate.
9. Select **[Choose File]** and brose your local machine for the private key.
10. Enter the CA password.
11. Click **[Generate]**.

5.2.3.1.2. Transfer Proxy Configuration to New Host

Procedure: Transfering the Proxy Configuration

1. From the Server transfer the generated tar.gz file containing the proxy configuration to the new Proxy host:

```
scp config.tar.gz <uyuni-proxy-FQDN>:/root/
```

2. Disable the legacy proxy prior to executing the next step:

```
spacewalk-proxy stop
```

3. Deploy the new Proxy with:

```
systemctl start uyuni-proxy-pod
```

4. Enable the new Proxy with:

```
systemctl enable --now uyuni-proxy-pod
```

5. Run podman ps to verify all the containers are present and running:

```
proxy-salt-broker
proxy-httpd
proxy-tftpd
proxy-squid
proxy-ssh
```

5.2.3.2. Migrate Uyuni Proxy to Uyuni 2026.11 Containerized Proxy

Procedure: Migrate Uyuni Containerized Proxy to Uyuni 2026.11 New Containerized Proxy

1. Boot your new machine and begin installation of openSUSE Leap Micro 6.2.
2. Complete the installation.
3. Update the system:

```
transactional-update --continue
```

4. Install mgrpxy and optionally, mgrpxy-bash-completion:

```
transactional-update pkg install mgrpxy mgrpxy-bash-completion
```

5. Reboot.
6. Copy your tar.gz proxy configuration to the host.

5.2.3.3. Install Packages Using the Web UI

The mgrpxy and mgrpxy-bash-completion packages can also be installed via the web UI after the minion has been bootstrapped and registered with the Server.

Procedure: Installing Packages Using the Web UI

1. After installation, ensure that the SLE Micro 6.2 parent channel and Proxy child channels are added and synchronized from the **Admin › Setup Wizard → Products** page.
2. In the Web UI, go to **Systems › Activation Keys** and create an activation key linked for the synchronized SLE Micro 6.2 channel.

3. Bootstrap your system as a minion using the **Systems › Bootstrapping** page.
4. Once the new machine is onboarded and displayed in the systems list, select the system and navigate to the **System Details › Install Package** page.
5. Install the packages mgrpxy and mgrpxy-bash-completion.
6. Reboot the system.

5.2.3.4. Generate Proxy Config With spacecmd and Self-Signed Certificate

You can generate a Proxy configuration using spacecmd.

Procedure: Generate Proxy Config With spacecmd and Self-Signed Certificate

1. SSH into your container host.
2. Execute the following command replacing the Server and Proxy FQDN:

```
mgrctl exec -ti 'spacecmd proxy_container_config_generate_cert -- dev-pxy.example.com
dev-srv.example.com 2048 email@example.com -o /tmp/config.tar.gz'
```

3. Copy the generated config to the Proxy:

```
mgrctl cp server:/tmp/config.tar.gz .
```

4. Deploy the Proxy with:

```
mgrpxy install config.tar.gz
```

5.2.3.5. Generate Proxy Config With spacecmd and Custom Certificate

You can generate Proxy configuration using spacecmd for a custom certificates rather than default self-signed certificates.



2 GB represents the default proxy squid cache size. This will need to be adjusted for your environment.

Procedure: Generate Proxy Config With spacecmd and Custom Certificate

1. SSH into your Server container host.
2. Execute the following command replacing the Server and Proxy FQDN:

```
for f in ca.crt proxy.crt proxy.key; do
  mgrctl cp $f server:/tmp/$f
done
mgrctl exec -ti 'spacecmd proxy_container_config -- -p 8022 pxy.example.com
srv.example.com 2048 email@example.com /tmp/ca.crt /tmp/proxy.crt /tmp/proxy.key -o
/tmp/config.tar.gz'
```

3. Copy the generated config to the Proxy:

```
mgrctl cp server:/tmp/config.tar.gz .
```

4. Deploy the Proxy with:

```
mgrpxy install config.tar.gz
```

5.3. Clients

5.3.1. Upgrade Clients

Clients use the versioning system of their underlying operating system. For clients using SUSE operating systems, you can perform upgrades within the Uyuni Web UI.

For more information about upgrading clients, see **Client Configuration Guide › Client Upgrades**.

Chapter 6. Basic Server and Proxy Management

6.1. Custom YAML Configuration and Deployment with mgradm

You have the option to create a custom mgradm.yaml file, which the mgradm tool can utilize during deployment. All mgradm arguments have their YAML counterparts. For example, the mgradm argument --ssl-db-ca-intermediate can be specified in the mgradm.yaml file as follows:

```
ssl:
  db:
    ca:
      intermediate: /path/to/ca-intermediate.crt
```



mgradm will prompt for basic variables if they are not provided using command line parameters or the mgradm.yaml configuration file.

For security, **using command line parameters to specify passwords should be avoided**. Use a configuration file with proper permissions instead.

Procedure: Deploying the Uyuni Container with Podman Using a Custom Configuration File

1. Prepare a configuration file named mgradm.yaml similar to the following example:

```
# Database password. Randomly generated by default
db:
  password: MySuperSecretDBPass

# Password for the CA certificate
ssl:
  password: MySuperSecretSSLPassword

# Your SUSE Customer Center credentials
scc:
  user: ccUsername
  password: ccPassword

# Organization name
organization: YourOrganization

# Email address sending the notifications
emailFrom: notifications@example.com

# Administrators account details
admin:
  password: MySuperSecretAdminPass
  login: LoginName
  firstName: Admin
  lastName: Admin
  email: email@example.com
```

2. From the terminal, as root, run the following command. Entering your server's FQDN is optional.

```
mgradm -c mgradm.yaml install <FQDN>
```



You must deploy the container as sudo or root. The following error will be displayed on the terminal if you miss this step.

```
INF Setting up uyuni network
9:58AM INF Enabling system service
9:58AM FTL Failed to open /etc/systemd/system/uyuni-server.service
for writing
error="open /etc/systemd/system/uyuni-server.service: permission
denied"
```

3. Wait for deployment to complete.
4. Open a browser and proceed to your server's FQDN or IP address.

6.2. Starting and Stopping Containers

The Uyuni 2026.11 Server container can be restarted, started, and stopped using the following commands:

To restart the Uyuni 2026.11 Server execute the following command:

```
# mgradm restart
5:23PM INF Welcome to mgradm
5:23PM INF Executing command: restart
```

To start the server execute the following command:

```
# mgradm start
5:21PM INF Welcome to mgradm
5:21PM INF Executing command: start
```

To stop the server execute the following command:

```
# mgradm stop
5:21PM INF Welcome to mgradm
5:21PM INF Executing command: stop
```

6.3. Containers used by Uyuni

Below is a list of containers used by Uyuni 2026.11.

Table 10. Server Containers

Container Name	Description
uyuni-server	Main product container
uyuni-db	Database container for the product
uyuni-hub-xmlrpc	XML-RPC gateway for Hub deployment
uyuni-server-attestation	Server COCO attestation
uyuni-saline	Saline container for Salt observability
uyuni-server-migration	Migration helper container
uyuni-tftpd	Optional TFTP container for autoinstallation (disabled by default; enabled by passing --tftpd-enable to mgradm)

Table 11. Proxy Containers

Container Name	Description
uyuni-proxy-httpd	Main proxy container handling all HTTP communication
uyuni-proxy-squid	Squid cache
uyuni-proxy-salt-broker	Salt forwarder
uyuni-proxy-ssh	SSH forwarder
uyuni-proxy-tftpd	TFTPd to HTTP translator and forwarder

6.4. Persistent Container Volumes

Modifications performed within containers are not retained. Any alterations made outside of persistent volumes will be discarded. Below is a list of persistent volumes for Uyuni 2026.11.

To customize the default volume locations, ensure you create the necessary volumes before launching the pod for the first time, utilizing the podman volume create command.



Ensure that this table aligns precisely with the volumes mapping outlined in both the Helm chart and the systemctl services definitions.

6.4.1. Server

The following volumes are stored under the **Podman** default storage location on the server.

Table 12. Persistent Volumes: Podman Default Storage

Volume Name	Volume Directory
Podman Storage	/var/lib/containers/storage/volumes/

Table 13. Persistent Volumes: root

Volume Name	Volume Directory
root	/root

Table 14. Persistent Volumes: var/

Volume Name	Volume Directory
var-cobbler	/var/lib/cobbler
var-salt	/var/lib/salt
var-pgsql	/var/lib/pgsql/data
var-pgsql-backup	/var/lib/pgsql-backup
var-cache	/var/cache
var-spacewalk	/var/spacewalk
var-log	/var/log

Table 15. Persistent Volumes: srv/

Volume Name	Volume Directory
srv-salt	/srv/salt
srv-www	/srv/www/
srv-tftpboot	/srv/tftpboot
srv-formulametadata	/srv/formula_metadata
srv-pillar	/srv/pillar
srv-susemanager	/srv/susemanager
srv-spacewalk	/srv/spacewalk

Table 16. Persistent Volumes: etc/

Volume Name	Volume Directory
etc-apache2	/etc/apache2
etc-rhn	/etc/rhn
etc-systemd-multi	/etc/systemd/system/multi-user.target.wants
etc-systemd-sockets	/etc/systemd/system/sockets.target.wants
etc-salt	/etc/salt
etc-sssd	/etc/sssd
etc-tomcat	/etc/tomcat
etc-cobbler	/etc/cobbler
etc-sysconfig	/etc/sysconfig
etc-postfix	/etc/postfix
ca-cert	/etc/pki/trust/anchors

Table 17. Persistent Volumes: run/

Volume Name	Volume Directory
run-salt-master	/run/salt/master

6.4.2. Proxy

The following volumes are stored under the **Podman** default storage location on the proxy.

Table 18. Persistent Volumes: Podman Default Storage

Volume Name	Volume Directory
Podman Storage	/var/lib/containers/storage/volumes/

Table 19. Persistent Volumes: srv/

Volume Name	Volume Directory
uyuni-proxy-tftpboot	/srv/tftpboot

Table 20. Persistent Volumes: var/

Volume Name	Volume Directory
uyuni-proxy-rhn-cache	/var/cache/rhn
uyuni-proxy-squid-cache	/var/cache/squid

6.5. Understanding mgr-storage-server and mgr-storage-proxy

mgr-storage-server and mgr-storage-proxy are helper scripts provided with Uyuni.

They are designed to configure storage for Uyuni Server and Proxy.

The scripts take disk devices as arguments. mgr-storage-proxy requires a single argument for the storage disk device. mgr-storage-server requires a storage disk device and can optionally accept a second argument for a dedicated database disk device. While both normal and database storage can reside on the same disk, it is advisable to place the database on a dedicated, high-performance disk to ensure better performance and easier management.

6.5.1. What these tools do

Both mgr-storage-server and mgr-storage-proxy perform standard storage setup operations:

- Validate the provided storage devices.
- Ensure that devices are empty and suitable for use.
- Create XFS filesystems on the specified devices.
- Mount the devices temporarily for data migration.
- Move the relevant storage directories to the new devices.
- Create entries in /etc/fstab so that the storage mounts automatically on boot.
- Remount the devices at their final locations.

Table 21. Additional tool-specific behavior

mgr-storage-server	Optionally supports a separate device for database storage. Stops Uyuni services during migration, restarts them afterward. Moves Podman volumes directory /var/lib/containers/storage/volumes to the prepared storage, and optionally /var/lib/containers/storage/volumes/var-pgsql to the prepared database storage.

mgr-storage-proxy	Focuses only on proxy storage (no database storage support). Stops and restarts the proxy service during migration. Moves podman volumes directory /var/lib/containers/storage/volumes to the prepared storage.



Both tools automate standard Linux storage operations. There is no hidden or custom logic beyond what a Linux administrator would do manually.

6.5.2. What these tools do **not** do

- They do **not** create or manage LVM volumes.
- They do **not** configure RAID or complex storage topologies.
- They do **not** prevent you from managing storage using normal Linux tools after setup.
- They do **not** provide dynamic resizing or expansion capabilities — these must be handled using standard Linux storage tools.

6.5.3. Post-installation storage management

Once storage has been configured, you can safely manage it using standard Linux commands.

6.5.3.1. Examples

Listing 5. Example 1: Extending storage if using LVM

```
lvextend -L +10G /dev/your_vg/your_lv
xfs_growfs /var/lib/containers/storage/volumes
```

Example 2: Migrating to a larger disk

1. Add and format the new disk.
2. Mount it temporarily.
3. Use rsync to copy data.
4. Update /etc/fstab.
5. Remount at the correct location.

6.5.4. When to use, or not use



Always take a backup before making changes to your storage setup.

- Use these tools **only** during initial storage setup or when migrating to new storage where the tool is expected to handle data migration and update /etc/fstab.
- Do **not** rerun these scripts for resizing or expanding storage. Use standard Linux tools (e.g., lvextend, xfs_growfs) for such operations.

6.5.5. Summary

mgr-storage-server and mgr-storage-proxy help automate the initial persistent storage setup for Uyuni components using standard Linux storage practices. They do not limit or interfere with standard storage management afterward.

After setup, continue managing your storage using familiar Linux tools.



A full database volume can cause significant issues with system operation. As disk usage notifications have not yet been adapted for containerized environments, users are encouraged to monitor the disk space used by Podman volumes themselves, either through tools such as Grafana, Prometheus, or any other preferred method. Pay particular attention to the var-pgsql volume, located under /var/lib/containers/storage/volumes/.

Chapter 7. GNU Free Documentation License

Copyright © 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. 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 "GNU Free Documentation License".