SUSE Linux Enterprise Server 12 SP5

Subscription Management Tool Guide

Subscription Management Tool Guide

SUSE Linux Enterprise Server 12 SP5

by Tomáš Bažant, Jakub Friedl, and Florian Nadge

Publication Date: February 05, 2025

https://documentation.suse.com

Contents

About This Guide

Subscription Management Tool (SMT) for SUSE Linux Enterprise 12 SP5 allows enterprise customers to optimize the management of SUSE Linux Enterprise software updates and subscription entitlements. It establishes a proxy system for SUSE® Customer Center with repository (formerly known as catalog) and registration targets. This helps you centrally manage software updates within the firewall on a per-system basis, while maintaining your corporate security policies and regulatory compliance.

SMT allows you to provision updates for all of your devices running a product based on SUSE Linux Enterprise. By downloading these updates once and distributing them throughout the enterprise, you can set more restrictive firewall policies. This also reduces bandwidth usage, as there is no need to download the same updates for each device. SMT is fully supported and available as a download for customers with an active SUSE Linux Enterprise product subscription.

Subscription Management Tool provides functionality that can be useful in many situations, including the following:

- You want to update both SUSE Linux Enterprise and Red Hat Enterprise Linux servers.
- You want to get a detailed overview of your company's license compliance.
- Not all machines in your environment can be connected to SUSE Customer Center to register and retrieve updates for bandwidth or security reasons.
- There are SUSE Linux Enterprise hosts that are restricted and difficult to update without putting in place a custom update management solution.
- You need to integrate additional software update external or internal repositories into your update solution.
- You are looking for a turnkey box staging solution for testing updates before releasing them to the clients.
- You want to have a quick overview of the patch status of your SUSE Linux Enterprise servers and desktops.

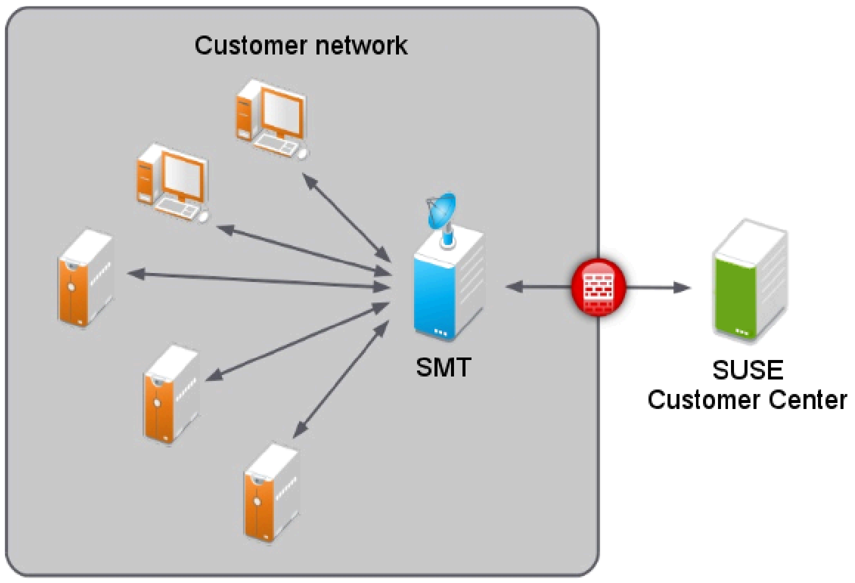


FIGURE 1: SMT

1 Overview

The Subscription Management Tool Guide is divided into the following chapters:

SMT Installation

Introduction to the SMT installation process and the SMT Configuration Wizard. You will learn how to install the SMT add-on on your base system during the installation process or on an already installed base system.

SMT Server Configuration

Description of the YaST configuration module SMT Server. This chapter explains how to set and configure organization credentials, SMT database passwords, and e-mail addresses to send SMT reports, or set the SMT job schedule, and activate or deactivate the SMT service.

Mirroring Repositories on the SMT Server

Explanation of how to mirror the installation and update sources with YaST.

Managing Repositories with YaST SMT Server Management

Description of how to register client machines on SUSE Customer Center. The client machines must be configured to use SMT.

SMT Reports

In-depth look at generated reports based on SMT data. Generated reports contain statistics of all registered machines and products used and of all active, expiring, or missing subscriptions.

SMT Tools and Configuration Files

Description of the most important scripts, configuration files and certificates supplied with SMT.

Configuring Clients to Use SMT

Introduction to configuring any client machine to register against SMT and download software updates from there instead of communicating directly with the SUSE Customer Center.

2 Available documentation

Online documentation

Our documentation is available online at https://documentation.suse.com ↗. Browse or download the documentation in various formats.

Note: Latest updates

The latest updates are usually available in the English-language version of this documentation.

SUSE Knowledgebase

If you have run into an issue, also check out the Technical Information Documents (TIDs) that are available online at https://www.suse.com/support/kb/ ↗. Search the SUSE Knowledgebase for known solutions driven by customer need.

Release notes

For release notes, see https://www.suse.com/releasenotes/ ↗.

In your system

For offline use, the release notes are also available under `/usr/share/doc/release-notes` on your system. The documentation for individual packages is available at `/usr/share/doc/packages`.

Many commands are also described in their *manual pages*. To view them, run `man`, followed by a specific command name. If the `man` command is not installed on your system, install it with **`sudo zypper install man`**.

3 Improving the documentation

Your feedback and contributions to this documentation are welcome. The following channels for giving feedback are available:

Service requests and support

For services and support options available for your product, see https://www.suse.com/support/ ↗.

To open a service request, you need a SUSE subscription registered at SUSE Customer Center. Go to https://scc.suse.com/support/requests ↗, log in, and click *Create New*.

Bug reports

Report issues with the documentation at https://bugzilla.suse.com/ ↗.

To simplify this process, click the *Report an issue* icon next to a headline in the HTML version of this document. This preselects the right product and category in Bugzilla and adds a link to the current section. You can start typing your bug report right away.

A Bugzilla account is required.

Contributions

To contribute to this documentation, click the *Edit source document* icon next to a headline in the HTML version of this document. This will take you to the source code on GitHub, where you can open a pull request.

A GitHub account is required.

Note: *Edit source document* only available for English

The *Edit source document* icons are only available for the English version of each document. For all other languages, use the *Report an issue* icons instead.

For more information about the documentation environment used for this documentation, see the repository's README.

Mail

You can also report errors and send feedback concerning the documentation to `doc-team@suse.com`. Include the document title, the product version, and the publication date of the document. Additionally, include the relevant section number and title (or provide the URL) and provide a concise description of the problem.

4 Documentation conventions

The following notices and typographic conventions are used in this document:

- `/etc/passwd`: Directory names and file names
- *`PLACEHOLDER`*: Replace *`PLACEHOLDER`* with the actual value
- `PATH`: An environment variable
- **`ls`**, `--help`: Commands, options, and parameters
- `user`: The name of a user or group
- `package_name`: The name of a software package
- **Alt** , **Alt**–**F1** : A key to press or a key combination. Keys are shown in uppercase as on a keyboard.
- *File*, *File › Save As*: menu items, buttons
- **AMD/Intel** This paragraph is only relevant for the AMD64/Intel 64 architectures. The arrows mark the beginning and the end of the text block.
  IBM Z, POWER This paragraph is only relevant for the architectures `IBM Z` and `POWER`. The arrows mark the beginning and the end of the text block.
- *Chapter 1, "Example chapter"*: A cross-reference to another chapter in this guide.
- Commands that must be run with `root` privileges. You can also prefix these commands with the **`sudo`** command to run them as a non-privileged user:

```
root # command
tux > sudo command
```

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

```
tux > command
```

- Commands can be split into two or multiple lines by a backslash character (`\`) at the end of a line. The backslash informs the shell that the command invocation will continue after the line's end:

```
tux > echo a b \
c d
```

- A code block that shows both the command (preceded by a prompt) and the respective output returned by the shell:

```
tux > command
output
```

- Notices

Warning: Warning notice

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

Important: Important notice

Important information you should be aware of before proceeding.

Note: Note notice

Additional information, for example about differences in software versions.

Tip: Tip notice

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

- Compact Notices

Additional information, for example about differences in software versions.

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

5 Support

Find the support statement for SUSE Linux Enterprise Server and general information about technology previews below. For details about the product lifecycle, see https://www.suse.com/lifecycle.

If you are entitled to support, find details on how to collect information for a support ticket at https://documentation.suse.com/sles-15/html/SLES-all/cha-adm-support.html.

5.1 Support statement for SUSE Linux Enterprise Server

To receive support, you need an appropriate subscription with SUSE. To view the specific support offers available to you, go to https://www.suse.com/support/ and select your product.

The support levels are defined as follows:

L1

Problem determination, which means technical support designed to provide compatibility information, usage support, ongoing maintenance, information gathering and basic troubleshooting using available documentation.

L2

Problem isolation, which means technical support designed to analyze data, reproduce customer problems, isolate a problem area and provide a resolution for problems not resolved by Level 1 or prepare for Level 3.

L3

Problem resolution, which means technical support designed to resolve problems by engaging engineering to resolve product defects which have been identified by Level 2 Support.

For contracted customers and partners, SUSE Linux Enterprise Server is delivered with L3 support for all packages, except for the following:

- Technology previews.
- Sound, graphics, fonts, and artwork.
- Packages that require an additional customer contract.

- Some packages shipped as part of the module *Workstation Extension* are L2-supported only.
- Packages with names ending in `-devel` (containing header files and similar developer resources) will only be supported together with their main packages.

SUSE will only support the usage of original packages. That is, packages that are unchanged and not recompiled.

5.2 Technology previews

Technology previews are packages, stacks, or features delivered by SUSE to provide glimpses into upcoming innovations. Technology previews are included for your convenience to give you a chance to test new technologies within your environment. We would appreciate your feedback. If you test a technology preview, please contact your SUSE representative and let them know about your experience and use cases. Your input is helpful for future development.

Technology previews have the following limitations:

- Technology previews are still in development. Therefore, they may be functionally incomplete, unstable, or otherwise *not* suitable for production use.
- Technology previews are *not* supported.
- Technology previews may only be available for specific hardware architectures.
- Details and functionality of technology previews are subject to change. As a result, upgrading to subsequent releases of a technology preview may be impossible and require a fresh installation.
- SUSE may discover that a preview does not meet customer or market needs, or does not comply with enterprise standards. Technology previews can be removed from a product at any time. SUSE does not commit to providing a supported version of such technologies in the future.

For an overview of technology previews shipped with your product, see the release notes at https://www.suse.com/releasenotes ↗.

1 Overview

The Subscription Management Tool (SMT) for SUSE Linux Enterprise 12 SP5 allows enterprise customers to optimize the management of SUSE Linux Enterprise software updates and subscription entitlements. It establishes a proxy system for SUSE® Customer Center with repositories and registration targets. This helps you to centrally manage software updates within a firewall on a per-system basis, while maintaining your corporate security policies and regulatory compliance.

SMT allows you to provision updates for all of your devices running a product based on SUSE Linux Enterprise. By downloading these updates once and distributing them throughout the enterprise, you can set more restrictive firewall policies. This also reduces bandwidth usage, as there is no need to download the same updates for each device. SMT is fully supported and available as a download for customers with an active SUSE Linux Enterprise product subscription.

Subscription Management Tool provides functionality that can be useful in many situations, including the following:

- You want to update SUSE Linux Enterprise servers.
- Not all machines in your environment can be connected to SUSE Customer Center to register and retrieve updates for bandwidth or security reasons.
- There are SUSE Linux Enterprise hosts that are restricted and difficult to update without putting in place a custom update management solution.
- You need to integrate additional external or internal repositories.

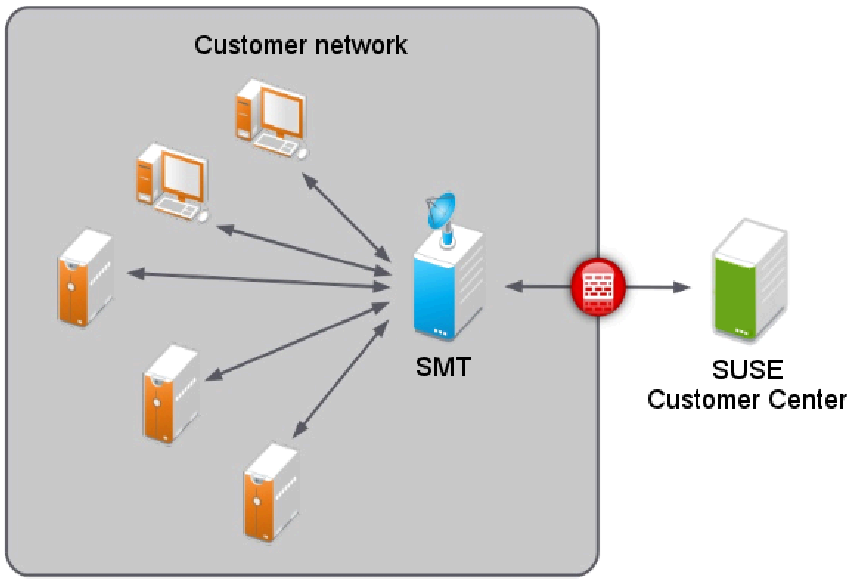


FIGURE 1.1: RMT

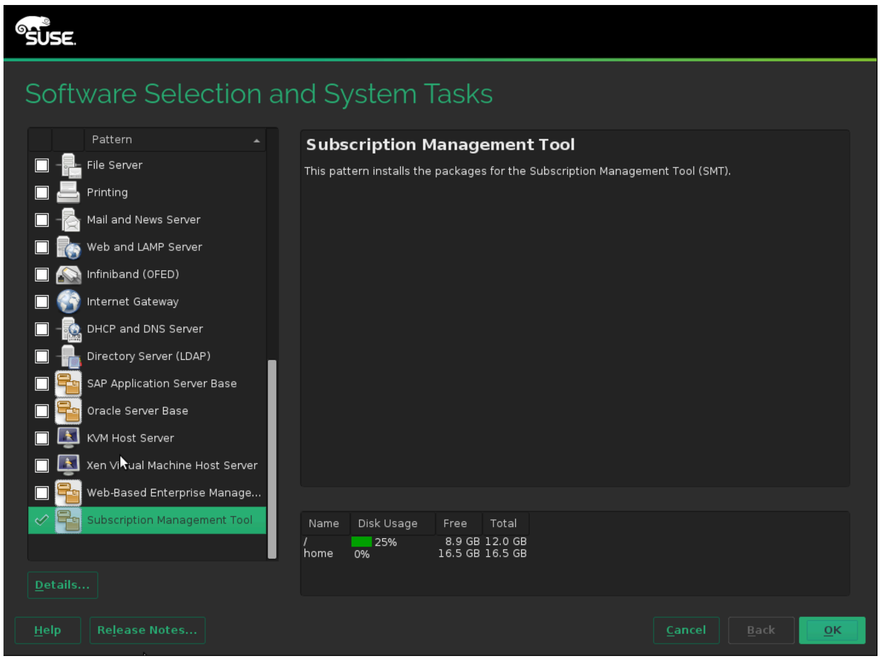
Important: Client Migration from SLE 12 to 15

Starting with SLE 15, SMT was replaced by RMT. If you are planning to migrate SLE 12 clients to version 15, RMT is the supported product to handle such migrations. If you still need to use SMT for these migrations, beware that the migrated clients will have *all* installation modules enabled.

For a feature comparison between RMT and SMT, see https://documentation.suse.com/sles/15/html/SLES-all/cha-rmt-migrate.html#sec-rmt-migrate-notes-features.

2 SMT Installation

SMT is included in SUSE Linux Enterprise Server starting with version 12 SP1. To install it, start SUSE Linux Enterprise Server installation, and click *Software* on the *Installation Settings* screen. Select the *Subscription Management Tool* pattern on the *Software Selection and System Tasks* screen, then click *OK*.



Subscription Management Tool

This pattern installs the packages for the Subscription Management Tool (SMT).

Name	Disk Usage	Free	Total
/	25%	8.9 GB	12.0 GB
home	0%	16.5 GB	16.5 GB

FIGURE 2.1: SMT PATTERN

Tip: Installing SMT on an Existing System

To install SMT on the existing SUSE Linux Enterprise Server system, run *YaST* › *Software* › *Software Management*, select *View* › *Patterns* and select the *SMT* pattern there.

It is recommended to check for available SMT updates immediately after installing SUSE Linux Enterprise Server using the `zypper patch` command. SUSE continuously releases maintenance updates for SMT, and newer packages are likely to be available.

After the system is installed and updated, perform an initial SMT configuration using *YaST* › *Network Services* › *SMT Configuration Wizard*.

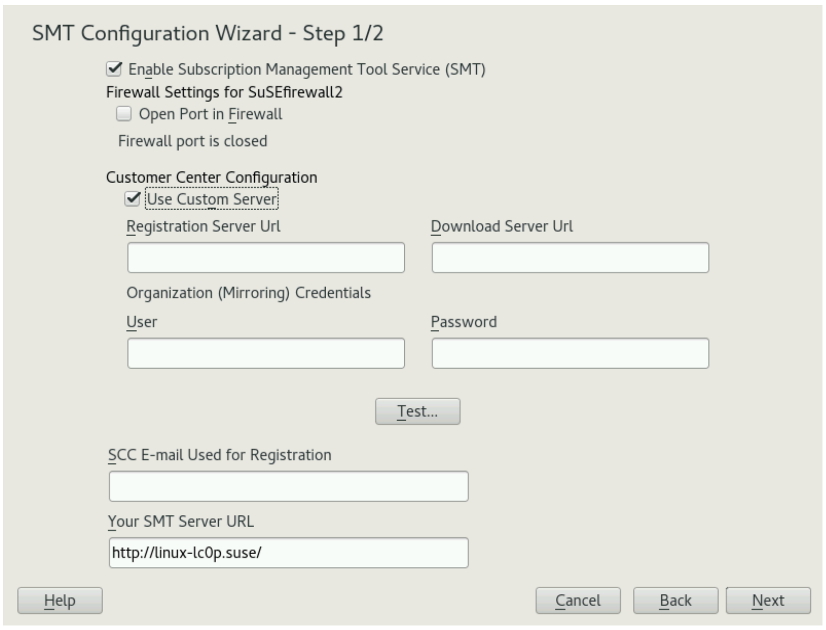
Note: Install smt-client

The `smt-client` package needs to be installed on clients connected to the SMT server. The package requires no configuration, and it can be installed using the **`sudo zypper in smt-client`** command.

2.1 SMT Configuration Wizard

The two-step *SMT Configuration Wizard* helps you configure SMT after SUSE Linux Enterprise Server installation is finished. You can change the configuration later using the YaST SMT Server Configuration module—see *Chapter 3, SMT Server Configuration*.

1. The *Run Subscription Management Tool service (SMT)* option is enabled by default. Toggle it only if you want to disable the SMT product.
   If the firewall is enabled, enable *Open Port in Firewall* to allow access to the SMT service from remote computers.
   Enter your SUSE Customer Center organization credentials in *User* and *Password*. If you do not know your SUSE Customer Center credentials, refer to *Section 4.1, "Mirroring Credentials"*.
   Test the entered credentials using the *Test* button. SMT will connect to the Customer Center server using the provided credentials and download testing data.
   Enter the e-mail address you used for the SUSE Customer Center registration into *SCC E-mail Used for Registration*.
   Your SMT Server URL should contain the URL of the SMT server being configured. It is populated automatically.
   Click *Next* to continue to the second configuration step.


SMT Configuration Wizard - Step 1/2

[✓] Enable Subscription Management Tool Service (SMT)

Firewall Settings for SuSEfirewall2

[ ] Open Port in Firewall

Firewall port is closed

Customer Center Configuration

[✓] Use Custom Server

Registration Server Url

Download Server Url

Organization (Mirroring) Credentials

User

Password

SCC E-mail Used for Registration

Your SMT Server URL

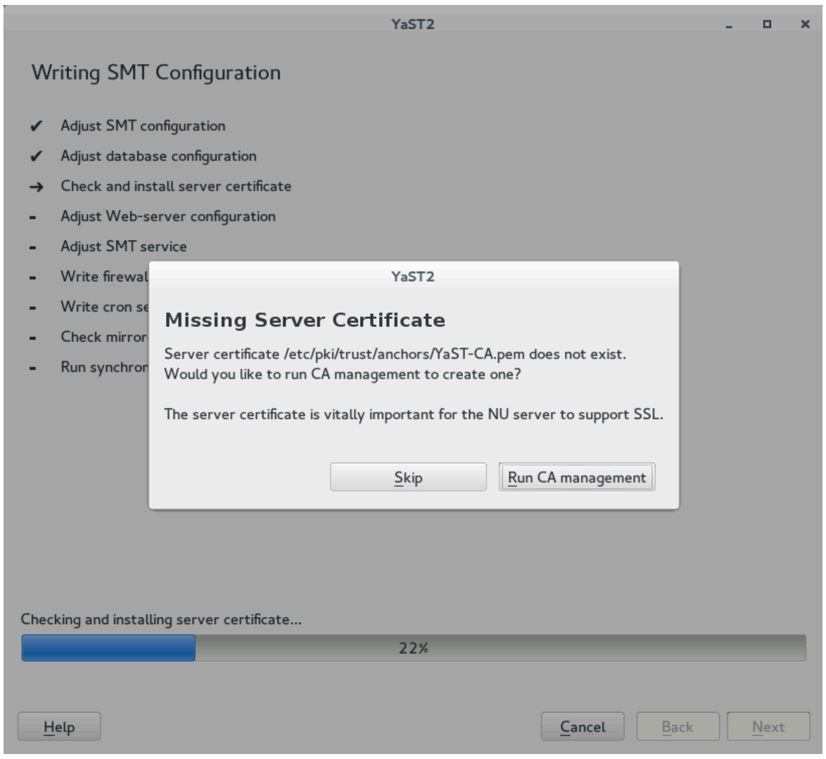
http://linux-lc0p.suse/

FIGURE 2.2: SMT WIZARD

2. For security reasons, SMT requires a separate user to connect to the database. In the *Database Password for smt User* screen, set the database password for this user.
   Enter all e-mail addresses for receiving SMT reports using the *Add* button. Use the *Edit* and *Delete* buttons to modify and delete the existing addresses. When you have done that, click *Next*.

3. If the current database root password is empty, you will be prompted to specify it.

4. By default, SMT is set to communicate with the client hosts via a secure protocol. For this, the server needs to have a server SSL certificate. The wizard displays a warning if the certificate does not exist. You can create a certificate using the *Run CA Management* button. Refer to *Book "Security and Hardening Guide", Chapter 18 "Managing X.509 Certification", Section 18.2 "YaST Modules for CA Management"* for detailed information on managing certificates with YaST.


YaST2

Writing SMT Configuration

✔ Adjust SMT configuration
✔ Adjust database configuration
→ Check and install server certificate
- Adjust Web-server configuration
- Adjust SMT service
- Write firewal
- Write cron se
- Check mirror
- Run synchron

YaST2

Missing Server Certificate

Server certificate /etc/pki/trust/anchors/YaST-CA.pem does not exist.
Would you like to run CA management to create one?

The server certificate is vitally important for the NU server to support SSL.

Checking and installing server certificate...

FIGURE 2.3: MISSING SERVER CERTIFICATE

2.2 Upgrading from Previous Versions of SMT

This section provides information on upgrading SMT from the previous versions.

Important: Upgrade from Versions Prior to 11 SP3

A direct upgrade path from SMT prior to version 11 SP3 is not supported. You need to do the following:

1. Upgrade the operating system to SUSE Linux Enterprise Server 11 SP3 or SP4 as described at https://documentation.suse.com/sles-11/html/SLES-all/cha-update-sle.html ↗.

2. At the same time upgrade SMT to version 11 SP3 as described at https://documentation.suse.com/smt/11.3/ ↗.

3. Follow the steps described in *Section 2.2.2, "Upgrade from SMT 11 SP3"*.

2.2.1 Upgrade from SMT 12 SP1

Upgrade from SMT 12 SP1 is performed automatically during the SUSE Linux Enterprise Server upgrade and requires no additional manual steps. For more information on SUSE Linux Enterprise Server upgrade, see *Book "Deployment Guide", Chapter 20 "Upgrading SUSE Linux Enterprise"*.

2.2.2 Upgrade from SMT 11 SP3

To upgrade SMT from version 11 SP3 to 12 SP2, follow the steps below.

1. If you have not already done so, migrate from Novell Customer Center to SUSE Customer Center as described in *Section 2.2.2.1, "Migration to SUSE Customer Center on SMT 11 SP3"*.
2. Back up and migrate the database. See the general procedure in *Book "Deployment Guide", Chapter 20 "Upgrading SUSE Linux Enterprise", Section 20.3.5 "Migrate your MySQL Database"*.
3. Upgrade to SUSE Linux Enterprise Server 12 SP2 as described in *Book "Deployment Guide", Chapter 20 "Upgrading SUSE Linux Enterprise"*.
4. Look if the new `/etc/my.cnf.rpmnew` exists and update it with any custom changes you need. Then copy it over the existing `/etc/my.cnf`:

   ```
   cp /etc/my.cnf.rpmnew /etc/my.cnf
   ```

5. Enable the `smt` target to start at the system boot:

   ```
   systemctl enable smt.target
   ```

   Start it immediately, if necessary:

   ```
   systemctl start smt.target
   ```

2.2.2.1 Migration to SUSE Customer Center on SMT 11 SP3

Before upgrading to SUSE Linux Enterprise Server 12, you need to switch the registration center on SUSE Linux Enterprise Server 11. SMT now registers with SUSE Customer Center instead of Novell Customer Center. You can do this either via a YaST module or command line tools.

Before performing the switch between customer centers, make sure that the target customer center serves all products that are registered with SMT. Both YaST and the command line tools perform a check to find out whether all products can be served with the new registration server.

To perform the migration to SUSE Customer Center via command line, use the following command:

```
smt ncc-scc-migration
```

The migration itself takes time, and during the migration process the SMT server may not be able to serve clients that are already registered.

The migration process itself changes the registration server and the proper type of API in the configuration files. No further (configuration) changes are needed on the SMT.

To migrate from Novell Customer Center to SUSE Customer Center via YaST, use the YaST smt-server module.

When migration has been completed, it is necessary to synchronize SMT with the customer center. It is recommended to ensure that the repositories are up to date. This can be done using the following commands:

```
   smt sync
   smt mirror
```

2.3 Enabling SLP Announcements

SMT includes the SLP service description file (`/etc/slp.reg.d/smt.reg`). To enable SLP announcements of the SMT service, open respective ports in your firewall and enable the SLP service:

```
sysconf_addword /etc/sysconfig/SuSEfirewall2 FW_SERVICES_EXT_TCP "427"
sysconf_addword /etc/sysconfig/SuSEfirewall2 FW_SERVICES_EXT_UDP "427"
systemctl enable slpd.service
systemctl start slpd.service
```

3 SMT Server Configuration

This chapter introduces the YaST configuration module for the SMT server. This module can be used to set and configure mirroring credentials, SMT database passwords, and e-mail addresses for receiving SMT reports. The module also lets you set the SMT job schedule, and activate or deactivate the SMT service.

To configure SMT with SMT Server Configuration, follow the steps below.

1. Start the YaST module *SMT Server Configuration* from the YaST control center or by running `yast smt-server` from the command line.

2. To activate SMT, toggle the *Enable Subscription Management Tool Service (SMT)* option in the *Customer Center Access* section. For more information about activating SMT with YaST, see *Section 3.1, "Activating and Deactivating SMT with YaST"*.

3. If the firewall is enabled, activate *Open Port in Firewall*.

4. In the *Customer Center Configuration* section of *Customer Center Access*, you can set the custom server URLs. Set and test credentials for the SUSE Update service. Correct credentials are necessary to enable mirroring from the download server and determine the products that should be mirrored. Also set the e-mail address used for the registration and the URL of your SMT server. For more information, see *Section 3.2, "Setting the Update Server Credentials with YaST"*.

5. In the *Database and Reporting* section, set the password for the SMT user in the MariaDB database and specify e-mail addresses for receiving reports. For more information, see *Section 3.3, "Setting SMT Database Password with YaST"* and *Section 3.4, "Setting E-mail Addresses to Receive Reports with YaST"*.

6. In the *Scheduled SMT Jobs* section, set a schedule for SMT jobs, such as synchronization of updates, SUSE Customer Center registration, and SMT report generation. For more information, see *Section 3.5, "Setting the SMT Job Schedule with YaST"*.

7. When you are satisfied with the configuration, click *OK*. YaST updates the SMT configuration and starts or restarts necessary services.
   If you want to abort the configuration and cancel any changes, click *Cancel*.

Note: Check for Certificate

When the SMT Configuration applies changes, it checks whether the common server certificate exists. If the certificate does not exist, you will be asked whether the certificate should be created.

3.1 Activating and Deactivating SMT with YaST

YaST provides an easy way to activate or deactivate the SMT service. To activate SMT using YaST, follow the steps below.

1. Switch to the *Customer Center Access* section in the SMT Configuration.
2. Activate the *Run Subscription Management Tool service (SMT)* option.

   Note: Organization Credentials

   Specify organization credentials before activating SMT. For more information on how to set organization credentials with YaST, see *Section 3.2, “Setting the Update Server Credentials with YaST”*.

3. Click *Finish* to apply the changes and leave the SMT Configuration.

To deactivate SMT with YaST, proceed as follows.

1. Switch to the *Customer Center Access* section in the SMT Configuration.
2. Disable the *Run Subscription Management Tool service (SMT)* option.
3. Click *Finish* to apply the changes and leave the SMT Configuration.

When activating SMT, YaST performs the following actions.

- The Apache configuration is changed by creating symbolic links in the `/etc/apache2/conf.d/` directory. Links to the `/etc/smt.d/nu_server.conf` and `/etc/smt.d/smt_mod_perl.conf` files are created there.
- The Apache Web server is started (or reloaded if already running).

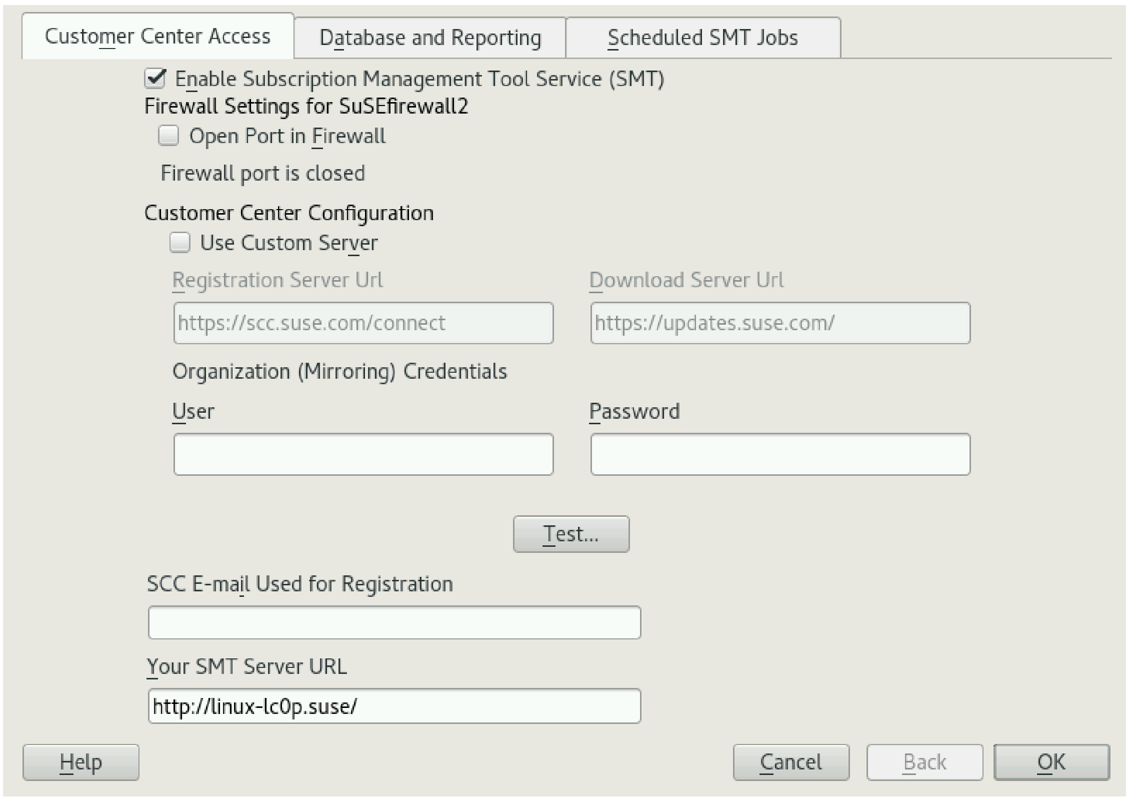
- The MariaDB server is started or restarted. The smt user and all necessary tables in the database are created, if needed.
- The schema of the SMT database is checked. If the database schema is outdated, the SMT database is upgraded to the current schema.
- Cron is updated by creating a symbolic link in the `/etc/cron.d/` directory. A link to the `/etc/smt.d/novell.com-smt` file is created there.

When deactivating SMT, YaST performs the following actions.

- Symbolic links that were created upon SMT activation in the /etc/apache2/conf.d/ and /etc/cron.d/ directories are deleted.
- The Cron daemon, the Apache server, and the MariaDB database daemon are restarted. Neither Apache nor MariaDB are stopped, as they may be used for other purposes than the SMT service.

3.2 Setting the Update Server Credentials with YaST

The following procedure describes how to set and test the download server credentials and the URL of the download server service using YaST.



☑ Enable Subscription Management Tool Service (SMT)

Firewall Settings for SuSEfirewall2

☐ Open Port in Firewall

Firewall port is closed

Customer Center Configuration

☐ Use Custom Server

Registration Server Url: https://scc.suse.com/connect

Download Server Url: https://updates.suse.com/

Organization (Mirroring) Credentials

User

Password

SCC E-mail Used for Registration

Your SMT Server URL: http://linux-lc0p.suse/

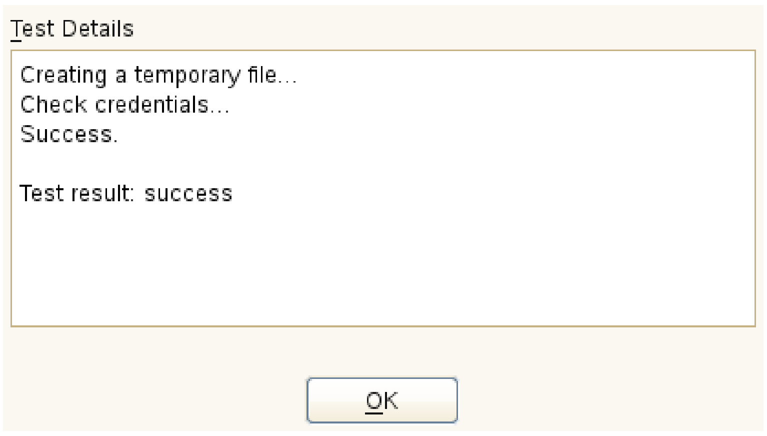
FIGURE 3.1: SETTING THE UPDATE SERVER CREDENTIALS WITH YAST

1. Switch to the *Customer Center Access* section in the SMT Configuration. If the credentials have been already set with YaST or via the `/etc/smt.conf` configuration file, they will be displayed in the *User* and *Password* fields.

2. If you do not have credentials, visit SUSE Customer Center to obtain them. For more details, see *Section 4.1, "Mirroring Credentials"*.

3. Enter your user name and password in the appropriate fields.

4. Click *Test* to check the credentials. YaST will try to download a list of available repositories with the provided credentials. If the test succeeds, the last line of the test results will read `Test result: success`. If the test fails, check the provided credentials and try again.



```
Creating a temporary file...
Check credentials...
Success.

Test result: success
```

FIGURE 3.2: SUCCESSFUL TEST OF THE UPDATE SERVER CREDENTIALS

5. Enter the *SCC E-mail Used for Registration*. This should be the address you used to register to SUSE Customer Center.
   Enter *Your SMT Server URL* if it has not been detected automatically.
6. Click *OK*.

3.3 Setting SMT Database Password with YaST

For security reasons, SMT uses its own user in the database. YaST provides an interface for setting up or changing the SMT database password. To set or change the SMT database password with YaST, follow the steps below.

1. Switch to the *Database and Reporting* section in the SMT Configuration module.
2. Enter the *Database Password for SMT User*. Confirm the password by re-entering it, then click *OK*.

3.4 Setting E-mail Addresses to Receive Reports with YaST

YaST SMT provides an interface for setting up a list of e-mail addresses for receiving reports from SMT. To edit this list of addresses, proceed as follows.

1. Switch to the *Database and Reporting* section in the SMT Configuration.

2. The list of e-mail addresses is shown in the table. Use the appropriate buttons to add, edit, and delete existing address entries.

3. Click *OK*.

The comma-separated list of addresses for SMT reports is written to the `reportEmail` section of the `/etc/smt.conf` configuration file.

3.5 Setting the SMT Job Schedule with YaST

The SMT Configuration module provides an interface to schedule recurring SMT jobs. YaST uses `cron` to schedule configured jobs. If needed, `cron` can be used directly. There are five types of recurring jobs that can be set:

Synchronization of Updates

Synchronizes with SUSE Customer Center, updates repositories, and downloads new updates.

Generation of Reports

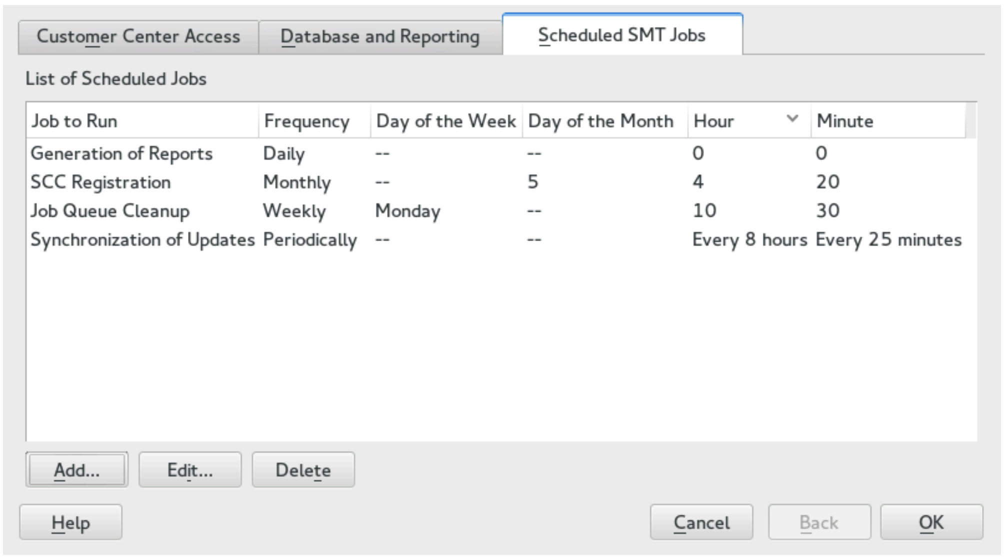
Generates and sends SMT Subscription Reports to addresses defined in *Section 3.4, "Setting E-mail Addresses to Receive Reports with YaST"*.

SCC Registration

Registers with SUSE Customer Center all clients that are not already registered or that changed their data since the last registration.

Job Queue Cleanup

Cleans up queued jobs. It removes finished or failed jobs from the job queue that are older than eight days. It also removes job artifacts that are left in the database as result of an error.



Job to Run	Frequency	Day of the Week	Day of the Month	Hour	Minute
Generation of Reports	Daily	--	--	0	0
SCC Registration	Monthly	--	5	4	20
Job Queue Cleanup	Weekly	Monday	--	10	30
Synchronization of Updates	Periodically	--	--	Every 8 hours	Every 25 minutes

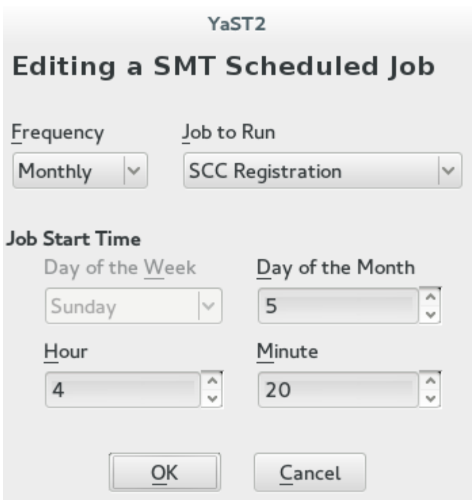
FIGURE 3.3: SMT JOB SCHEDULE CONFIGURATION

Use the following procedure to configure the schedule of SMT jobs with YaST.

1. Switch to the *Scheduled SMT Jobs* section in the SMT Configuration. The table contains a list of all scheduled jobs, their type, frequency, date, and time to run. You can add, delete, and edit the existing scheduled tasks.

2. To add a scheduled SMT job, click *Add*. This opens the *Adding New SMT Scheduled Job* dialog.
   Choose the synchronization job to schedule. You can choose between *Synchronization of Updates*, *Report Generation*, *SCC Registration*, and *Job Queue Cleanup*.
   Choose the *Frequency* of the new scheduled SMT job. Jobs can be performed *Daily*, *Weekly*, *Monthly*, or *Periodically* (every n-th hour or every m-th minute).
   Set the *Job Start Time* by entering *Hour* and *Minute*. In case of a recurring job, enter the relevant intervals. For weekly and monthly schedules, select *Day of the Week* or *Day of the Month*.
   Click *Add*.

3. To edit a scheduled SMT job (for example, change its frequency, time, or date), select the job in the table and click *Edit*. Then change the desired parameters and click *OK*.


YaST2

Editing a SMT Scheduled Job

Frequency: Monthly

Job to Run: SCC Registration

Job Start Time

Day of the Week: Sunday

Day of the Month: 5

Hour: 4

Minute: 20

FIGURE 3.4: SETTING SCHEDULED JOB WITH YAST

4. To cancel a scheduled job and delete it from the table, select the job in the table and click *Delete*.

5. Click *OK* to apply the settings and quit the SMT Configuration.

4 Mirroring Repositories on the SMT Server

You can mirror the installation and update repositories on the SMT server. This way, you do not need to download updates on each machine, which saves time and bandwidth.

Important: SUSE Linux Enterprise Server 9 Repositories

As SUSE Linux Enterprise Server 9 is no longer supported, SMT does not mirror SUSE Linux Enterprise Server 9 repositories.

4.1 Mirroring Credentials

Before you create a local mirror of the repositories, you need appropriate organization credentials. You can obtain the credentials from SUSE Customer Center.

To get the credentials from SUSE Customer Center, follow these steps:

1. Visit SUSE Customer Center at http://scc.suse.com and log in.
2. If you are a member of multiple organizations, select the organization you want to work with from the drop-down box in the top-right corner.
3. Click *Proxies* in the top menu.
4. Switch to the *Mirroring credentials* section.
5. To see the password, click the 👁 icon.

The obtained credentials should be set with the YaST SMT Server Configuration module or added directly to the `/etc/smt.conf` file. For more information about the `/etc/smt.conf` file, see *Section 8.2.1, "/etc/smt.conf"*.

Tip: Merging Multiple Organization Site Credentials

SMT can only work with one mirror credential at a time. Multiple credentials are not supported. When a customer creates a new company, this generates a new mirror credential. This is not always convenient, as some products are available via the first set and other products via the second set. To request a merge of credentials, contact your account manager.

4.2 Managing Software Repositories with SMT Command Line Tools

This section describes tools and procedures for viewing information about software repositories available through SMT, configuring these repositories, and setting up custom repositories on the command line. For details on the YaST SMT Server Management module, see *Chapter 5, Managing Repositories with YaST SMT Server Management*.

4.2.1 Updating the Local SMT Database

The local SMT database needs to be updated periodically with the information downloaded from SUSE Customer Center. These periodic updates can be configured with the SMT Management module, as described in *Section 3.5, “Setting the SMT Job Schedule with YaST”*.

To update the SMT database manually, use the **`smt-sync`** command. For more information about the **`smt-sync`** command, see *Section 8.1.2.7, “smt-sync”*.

4.2.2 Enabled Repositories and Repositories That Can Be Mirrored

The database installed with SMT contains information about all software repositories available on SUSE Customer Center. However, the used mirror credentials determine which repositories can really be mirrored. For more information about getting and setting organization credentials, see *Section 4.1, “Mirroring Credentials”*.

Repositories that can be mirrored have the `MIRRORABLE` flag set in the repositories table in the SMT database. That a repository can be mirrored does not mean that it needs to be mirrored. Only repositories with the `DOMIRROR` flag set in the SMT database will be mirrored. For more information about configuring which repositories should be mirrored, see *Section 4.2.4, “Selecting Repositories to Be Mirrored”*.

4.2.3 Getting Information about Repositories

Use the **`smt-repos`** command to list available software repositories and additional information. Using this command without any options lists all available repositories, including repositories that cannot be mirrored. In the first column, the enabled repositories (repositories set to be mirrored) are marked with `Yes`. Disabled repositories are marked with `No`. The other columns show ID, type, name, target, and description of the listed repositories. The last columns show whether the repository can be mirrored and whether staging is enabled.

Use the `--verbose` option, to get additional information about the URL of the repository and the path it will be mirrored to.

The repository listing can be limited to the repositories that can be mirrored or to the repositories that are enabled. To list the repositories that can be mirrored, use the `-m` or `--only-mirrorable` option: **`smt-repos -m`**.

To list only enabled repositories, use the `-o` or `--only-enabled` option: **`smt-repos -o`** (see *Example 4.1, "Listing All Enabled Repositories"*).

EXAMPLE 4.1: LISTING ALL ENABLED REPOSITORIES

```
tux:~ # smt-repos -o
.----------------------------------------------------------------------------------------------------------------------.
| Mirr| ID | Type | Name                    | Target        | Description                             | Can be M| Stag|
+-----+----+------+-------------------------+---------------+-----------------------------------------+---------+-----+
| Yes |  1 | zypp | ATI-Driver-SLE11-SP2    | --            | ATI-Driver-SLE11-SP2                    | Yes     | Yes |
| Yes |  2 | zypp | nVidia-Driver-SLE11-SP2 | --            | nVidia-Driver-SLE11-SP2                 | Yes     | No  |
| Yes |  3 | nu   | SLED11-SP2-Updates      | sle-11-x86_64 | SLED11-SP2-Updates for sle-11-x86_64    | Yes     | No  |
| Yes |  4 | nu   | SLES11-SP1-Updates      | sle-11-x86_64 | SLES11-SP1-Updates for sle-11-x86_64    | Yes     | Yes |
| Yes |  5 | nu   | SLES11-SP2-Core         | sle-11-x86_64 | SLES11-SP2-Core for sle-11-x86_64       | Yes     | No  |
| Yes |  6 | nu   | SLES11-SP2-Updates      | sle-11-i586   | SLES11-SP2-Updates for sle-11-i586      | Yes     | No  |
| Yes |  7 | nu   | WebYaST-Testing-Updates | sle-11-i586   | WebYaST-Testing-Updates for sle-11-i586 | Yes     | No  |
'-----+----+------+-------------------------+---------------+-----------------------------------------+---------+-----'
```

You can also list only repositories with a specific name or show information about a repository with a specific name and target. To list repositories with a particular name, use the **`smt-repos`** *`REPOSITORY_NAME`* command. To show information about a repository with a specific name and target, use the **`smt-repos`** *`repository_name`* *`TARGET`* command.

To get a list of installation repositories from remote, see *Section 9.7, "Listing Accessible Repositories"*.

4.2.4 Selecting Repositories to Be Mirrored

Only enabled repositories can be mirrored. In the database, the enabled repositories have the `DOMIRROR` flag set. Repositories can be enabled or disabled using the **`smt-repos`** command.

To enable one or more repositories, follow these steps:

1. To enable all repositories that can be mirrored or to choose one repository from the list of all repositories, run the **`smt-repos -e`** command.

You can limit the list of repositories by using the relevant options. To limit the list to the repositories that can be mirrored, use the `-m` option: **`smt-repos -m -e`**. To limit the list to the repositories with a specific name, use the **`smt-repos -e`** *`REPOSITORY_NAME`* command. To list a repository with a specific name and target, use the **`smt-repos -e`** *`REPOSITORY_NAME TARGET`* command.
To enable all repositories belonging to a specific product, use the `--enable-by-prod` or `-p` option, followed by the name of the product and optionally the version, architecture, and release:

```
smt-repos -p product[,version[,architecture[,release]]]
```

For example, to enable all repositories belonging to SUSE Linux Enterprise Server 10 SP3 for PowerPC architecture, use the **`smt-repos -p SUSE-Linux-Enterprise-Server-SP3,10,ppc`** command. The list of known products can be obtained with the **`smt-list-products`** command.

Tip: Installer Self-Update Repository

SMT supports mirroring the installer self-update repository (find more information in *Book "Deployment Guide", Chapter 6 "Installation with YaST", Section 6.4.1 "Self-Update Process"*). If you need to provide the self-update repository, identify and enable it, for example:

```
$ smt-repos -m | grep Installer
$ smt-repos -e SLES12-SP2-Installer-Updates sle-12-x86_64
```

2. If more than one repository is listed, choose the one you want to enable: specify its ID listed in the repository table and press Enter. If you want to enable all the listed repositories, use `a` and press Enter.

To disable one or more repositories, follow these steps:

1. To disable all enabled repositories or just choose one repository from the list of all repositories, run the **`smt-repos -d`** command.
   To choose the repository to be disabled from a shorter list, or to disable all repositories from a limited group, use any of the available options to limit the list of repositories. To limit the list to the enabled repositories, use the `-o` option: **`smt-repos -o -d`**. To limit

the list to repositories with a particular name, use the `smt-repos -d REPOSITORY_NAME` command. To show a repository with a specific name and target, use the `smt-repos -d REPOSITORY_NAME TARGET` command.

2. If more than one repository is listed, choose which one you want to disable: specify its ID listed in the repository table and press `Enter` . If you want to disable all the listed repositories, use `a` and press `Enter` .

4.2.5 Deleting Mirrored Repositories

You can delete mirrored repositories that are no longer used. If you delete a repository, it will be physically removed from the SMT storage area.

Use the `smt-repos --delete` command to delete a repository with a specific name. To delete the repository in a namespace, specify the `--namespace DIRNAME` option.

The `--delete` option lists all repositories. You can delete the specified repositories by entering the ID number or the name and target. To delete all repositories, enter *a*.

Note: Detecting Repository IDs

Every repository has an SHA-1 hash that you can use as an ID. You can get the repository's hash by calling `smt-repos -v`.

4.2.6 Mirroring Custom Repositories

SMT also makes it possible to mirror repositories that are not available at the SUSE Customer Center. These repositories are called “custom repositories”, and they can be mirrored using the `smt-setup-custom-repos` command. It is also possible to delete custom repositories.

When adding a new custom repository, the `smt-setup-custom-repos` command inserts a new record in the database and sets the *mirror* flag to `true`. You can disable mirroring later, if necessary.

To set up a custom repository to be available through SMT, follow these steps:

1. If you do not know the ID of the product the new repositories should belong to, use `smt-list-products` to get the ID. For the description of the `smt-list-products`, see *Section 8.1.2.4, “smt-list-products”*.

2. Run

```
smt-setup-custom-repos --productid PRODUCT_ID \
--name REPOSITORY_NAME --exturl REPOSITORY_URL
```

PRODUCT_ID is the ID of the product the repository belongs to, *REPOSITORY_NAME* is the name of the repository, and *REPOSITORY_URL* is the URL of the repository. If the added repository needs to be available for more than one product, specify the IDs of all products that should use the added repository.
For example, the following command sets `My repository` available at `http://example.com/My_repository` to the products with the IDs `423`, `424`, and `425`:

```
smt-setup-custom-repos --productid 423 --productid 424 \
--productid 425 --name 'My_repository' \
--exturl 'http://example.com/My_repository'
```

Note: Mirroring Unsigned Repositories

By default, SUSE Linux Enterprise 10 does not allow the use of unsigned repositories. So if you want to mirror unsigned repositories and use them on client machines, be aware that the package installation tool—YaST or **`zypper`**—will ask you whether to use repositories that are not signed.

To remove an existing custom repository from the SMT database, use **`smt-setup-custom-repos --delete`** *`ID`*, where *`ID`* is the ID of the repository to be removed.

4.3 The Structure of /srv/www/htdocs for SLE 11

The path to the directory containing the mirror is set by the `MirrorTo` option in the `/etc/smt.conf` configuration file. For more information about `/etc/smt.conf`, see *Section 8.2.1, “/etc/smt.conf”*. If the `MirrorTo` option is not set to the Apache `htdocs` directory `/srv/www/htdocs/`, the following links need to be created. If the directories already exist, they need to be removed prior to creating the link (the data in these directories will be lost). In the following examples, *`MIRRORTO`* needs to be replaced by the path the option `MirrorTo` is set to.

- `/srv/www/htdocs/repo/$RCE` must point to *`MIRRORTO`*`/repo/$RCE/`
- `/srv/www/htdocs/repo/RPMMD` must point to *`MIRRORTO`*`/repo/RPMMD/`

- `/srv/www/htdocs/repo/testing` must point to `MIRRORTO/repo/testing/`
- `/srv/www/htdocs/repo/full` must point to `MIRRORTO/repo/full/`

The directory specified using the `MirrorTo` option and the subdirectories listed above must exist. Files, directories, and links in `/MIRRORTO` must belong to the `smt` user and the `www` group. Here is an example where the `MirrorTo` is set to `/mirror/data`:

```
l /srv/www/htdocs/repo/
total 16
lrwxrwxrwx 1 smt  www    22 Feb  9 14:23 $RCE -> /mirror/data/repo/$RCE/
drwxr-xr-x 4 smt  www  4096 Feb  9 14:23 ./
drwxr-xr-x 4 root root 4096 Feb  8 15:44 ../
lrwxrwxrwx 1 smt  www    23 Feb  9 14:23 RPMMD -> /mirror/data/repo/RPMMD/
lrwxrwxrwx 1 smt  www    22 Feb  9 14:23 full -> /mirror/data/repo/full/
drwxr-xr-x 2 smt  www  4096 Feb  8 11:12 keys/
lrwxrwxrwx 1 smt  www    25 Feb  9 14:23 testing -> /mirror/data/repo/testing/
drwxr-xr-x 2 smt  www  4096 Feb  8 14:14 tools/
```

The links can be created using the **`ln -s`** commands. For example:

```
cd /srv/www/htdocs/repo
for LINK in \$RCE RPMMD full testing; do
 ln -s /mirror/data/repo/${LINK}/ && chown -h smt.www ${LINK}
done
```

Important: The `/srv/www/htdocs/repo` Directory

The `/srv/www/htdocs/repo` directory must not be a symbolic link.

Important: Apache and Symbolic Links

By default Apache on SUSE Linux Enterprise Server is configured to not follow symbolic links. To enable symbolic links for `/srv/www/htdocs/repo/` add the following snippet to `/etc/apache2/default-server.conf` (or the respective virtual host configuration in case you are running SMT on a virtual host):

```
<Directory "/srv/www/htdocs/repo">
 Options FollowSymLinks
</Directory>
```

After having made the change, test the syntax and reload the Apache configuration files to activate the change:

```
rcapache2 configtest && rcapache2 reload
```

4.4 The Structure of /srv/www/htdocs for SLE 12

The repository structure in the `/srv/www/htdocs` directory matches the structure specified in SUSE Customer Center. There are the following directories in the structure (selected examples, similar for other products and architectures):

```
repo/SUSE/Products/SLE-SDK/12/x86_64/product/
```

Contains the -POOL repository of SDK (the GA version of all packages).

```
repo/SUSE/Products/SLE-SDK/12/x86_64/product.license/
```

Contains EULA associated with the product.

```
repo/SUSE/Updates/SLE-SDK/12/x86_64/update/
repo/SUSE/Updates/SLE-SDK/12/s390x/update/
repo/SUSE/Updates/SLE-SERVER/12/x86_64/update/
```

Contain update repositories for respective products.

```
repo/full/SUSE/Updates/SLE-SERVER/12/x86_64/update/
repo/testing/SUSE/Updates/SLE-SERVER/12/x86_64/update/
```

Contain repositories created for staging of respective repositories.

4.5 Using the Test Environment

You can mirror repositories to a test environment instead of the production environment. The test environment can be used with a limited number of client machines before the tested repositories are moved to the production environment. The test environment can be run on the main SMT server.

The testing environment uses the same structure as the production environment, but it is located in the `/srv/www/htdocs/repo/testing/` subdirectory.

To mirror a repository to the testing environment, you can use the *Staging* tab in the YaST SMT Management module, or the command **`smt-staging`**.

To register a client in the testing environment, follow these steps:

1. De-register the client from the SMT server by running **`SUSEConnect --de-register`** on the client host.
2. Modify `/etc/SUSEConnect` on the client machine as follows:

   ```
   namespace: testing
   ```

3. Re-register the client host against SMT in order for the new namespace setting to take effect. See general information about registering SMT clients in *Chapter 9, Configuring Clients to Use SMT*.

To move the testing environment to the production environment, manually copy or move it using the **`cp -a`** or **`mv`** command.

You can enable “staging” for a repository in the *Repositories* tab of the SMT Management module or with the **`smt-repos`** command. The mirroring happens automatically to `repo/full/`.

If you have an SLE11-based update repository with patches, SMT tools can be used to manage them. Using these tools, you can select patches, create a snapshot and copy it into `repo/testing/`. After tests are finished, you can copy the contents of `repo/testing` into the production area `/repo`.

SLE10-based update repositories are not supported by SMT tools. Not all of these repositories support selective staging. In this case, you must mirror the complete package.

Recommended workflow:

```
customer center => repo/full => repo/testing, => repo/production
```

4.6 Testing and Filtering Update Repositories with Staging

You can test repositories on any clients using the **`smt-staging`** command before moving them to the production environment. You can select new update repositories to be installed on clients.

You can either use the **`smt-staging`** command or the YaST SMT Management module for staging. For more details, see *Section 5.3, “Staging Repositories”*.

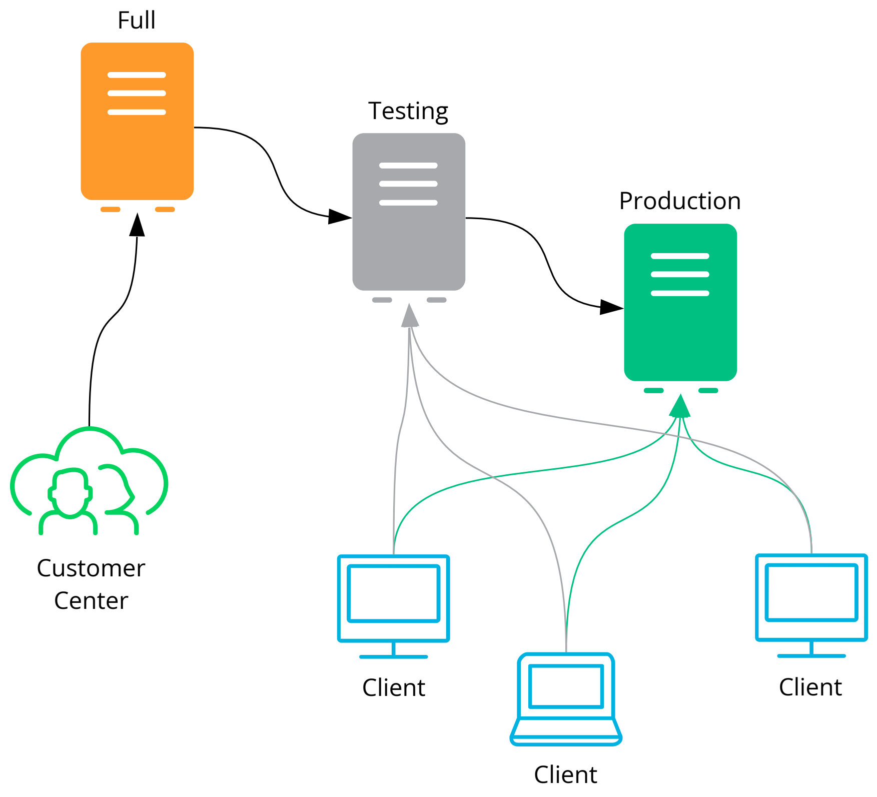


FIGURE 4.1: SMT STAGING SCHEMA

Repositories with staging enabled are mirrored to the `/MIRRORTO/repo/full` subdirectory. This subdirectory is usually not used by your clients. Incoming new updates are not automatically visible to the clients before you get a chance to test them. Later, you can generate a testing environment of this repository, which goes to the `/MIRRORTO/repo` directory.

If you have an SLE 11-based update repository with patches, you can use SMT tools to manage them. Using these tools, you can select patches, create a snapshot and put it into `repo/testing/`. After tests are finished, you can put the content of `repo/testing` into the `/repo` production area called the `default` staging group. You can create additional staging groups as needed using the **`smt-staging creategroup`** command.

Note: SLE 10-based Update Repositories

SLE 10-based update repositories are not supported by SMT tools. Not all of these repositories support selective staging. In this case, you need to mirror the complete package.

Enabling Staging

To enable or disable staging, use the **`smt-repos`** command with the `--enable-staging` or `-s` options:

```
smt-repos --enable-staging
```

You can enable the required repositories by entering the ID number or by entering the name and target. If you want to enable all repositories, enter *a*.

Generating Testing and Production Snapshots

To create the testing repository in the `default` staging group, run the following command:

```
smt-staging createrepo REPOSITORY_ID -–testing
```

You can then test the installation and functionality of the patches in testing clients. If testing was successful, create the production repository:

```
smt-staging createrepo REPOSITORY_ID --production
```

To create testing and production repositories in a named staging group, create the group and the repositories in this group:

```
smt-staging creategroup GROUPNAME TESTINGDIR PRODUCTIONDIR
smt-staging createrepo --group GROUPNAME REPOSITORY_ID -–testing
SMT-STAGING createrepo --group GROUPNAME REPOSITORY_ID -–production
```

This can be useful when you want to combine SLES11-SP1-Updates and SLES11-SP2-Updates of the `sle-11-x86_64` architecture into one repository of a group:

```
smt-staging creategroup SLES11SP1-SP2-Up test-sp1-sp2 prod-sp1-sp2
smt-staging createrepo --group SLES11SP1-SP2-Up \
  SLES11-SP1-Updates sle-11-x86_64 --testing
smt-staging createrepo --group SLES11SP1-SP2-Up \
  SLES11-SP2-Updates sle-11-x86_64 --testing
smt-staging createrepo --group SLES11SP1-SP2-Up \
  SLES11-SP1-Updates sle-11-x86_64 --production
smt-staging createrepo --group SLES11SP1-SP2-Up \
  SLES11-SP2-Updates sle-11-x86_64 --production
```

Group names can contain the following characters: `-_`, `a-z A-Z`, and `0-9`.

Filtering Patches

You can allow or forbid all or selected patches using the **`allow`** or **`forbid`** commands:

```
smt-staging forbid --patch ID
```

```
smt-staging forbid --category CATEGORYNAME
```

Signing Changed Repositories

Filtering one or more patches from a repository invalidates the original signature, and the repository needs to be signed again. The **`smt-staging createrepo`** command does that automatically, provided you configure the SMT server.

To enable signing of changed metadata, the admin needs to generate a new signing key. This can be done with GPG like this:

```
mkdir DIR
gpg --gen-key --homedir DIR
sudo mv DIR /var/lib/smt/.gnupg
sudo chown smt:users -R /var/lib/smt/.gnupg
sudo chmod go-rwx -R /var/lib/smt/.gnupg
```

The ID of the newly generated key can be obtained using the **`gpg --gen-key`** command. The ID must be added to the `signingKeyID` option in the `/etc/smt.conf` file.

At this point, the clients are not aware of the new key. Import the new key to clients during their registration as follows:

```
sudo -u smt gpg --homedir /var/lib/smt/.gnupg \
  --export -a SIGNING_KEYID \
  > /MIRRORTO/repo/keys/smt-signing-key.key
```

In this example, *`MIRRORTO`* is the base directory where repositories will be mirrored. After that, clients can import this key during the registration process.

Registering Clients in the Testing Environment

To register a client in the testing environment, follow these steps:

1. De-register the client from the SMT server by running **`SUSEConnect --de-register`** on the client host.

2. Modify `/etc/SUSEConnect` on the client machine as follows:

   ```
   namespace: testing
   ```

3. Re-register the client host against SMT in order for the new namespace setting to take effect. See general information about registering SMT clients in *Chapter 9, Configuring Clients to Use SMT*.

4.7 Repository Preloading

Deploying multiple SMT servers can take time if each new SMT server must mirror the same repositories.

To save time when deploying new SMT servers, the repositories can be preloaded from another server or disk instead. To do this, follow these steps:

1. Enable the repositories to be mirrored with the SMT, for example:

   ```
   smt-repos -e SLES12-Updates sle-12-x86_64
   ```

2. Perform a dry run of **`smt-mirror`** to create the required repository directories:

   ```
   smt-mirror -d --dryrun -L /var/log/smt/smt-mirror.log
   ```

   The following directories are created based on the repository above and the default `MirrorTo`:

   ```
   /srv/www/htdocs/repo/repoindex.xml
   /srv/www/htdocs/repo/$RCE/SLES12-Updates/sle-12-x86_64/*
   ```

3. Then copy the repositories from another SMT server, for example:

   ```
   rsync -av 'smt12:/srv/www/htdocs/repo/\$RCE/SLES12-Updates/sle-12-x86_64/' \
    '/srv/www/htdocs/repo/$RCE/SLES12-Updates/sle-12-x86_64/'
   ```

4. To get the repository data fixed, run the following command:

   ```
   smt-mirror -d -L /var/log/smt/smt-mirror.log
   ```

Important: Possible Error Messages

Errors, such as `repomd.xml is the same, but repo is not valid. Start mirroring.`, are considered normal. They occur because the metadata of the preloaded repositories in the server's database remains incorrect until the initial mirror of the repositories has completed.

5 Managing Repositories with YaST SMT Server Management

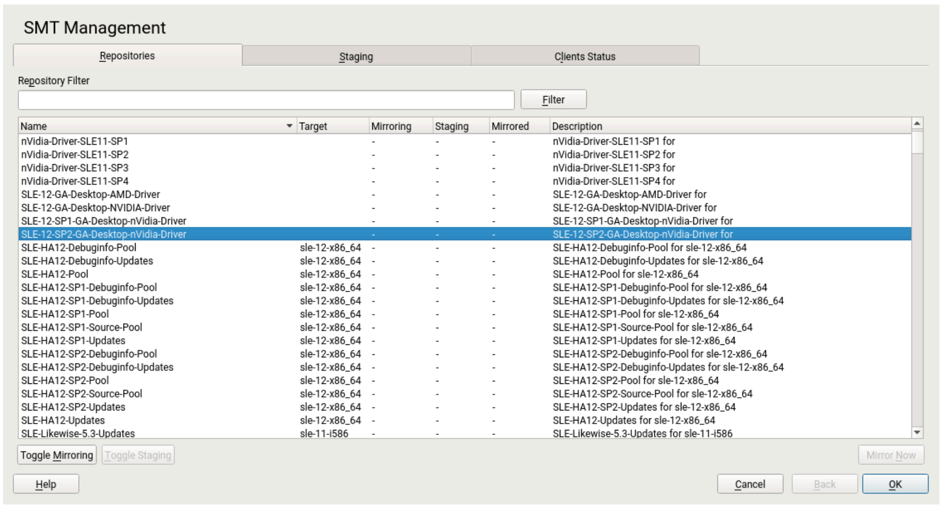
The YaST SMT Server Management module is designed to perform daily management tasks. It can be used to enable and disable the mirroring of repositories, the staging flag for repositories, and perform the mirroring and staging.

5.1 Starting SMT Management Module

SMT Management is a YaST module. There are two ways to start the module:

- Start YaST and select *Network Services*, then *SMT Server Management*
- Run the **`yast2 smt`** command in the terminal as `root`

This opens the SMT Management application window and switches to the *Repositories* section.



Name	Target	Mirroring	Staging	Mirrored	Description
nVidia-Driver-SLE11-SP1		-	-	-	nVidia-Driver-SLE11-SP1 for
nVidia-Driver-SLE11-SP2		-	-	-	nVidia-Driver-SLE11-SP2 for
nVidia-Driver-SLE11-SP3		-	-	-	nVidia-Driver-SLE11-SP3 for
nVidia-Driver-SLE11-SP4		-	-	-	nVidia-Driver-SLE11-SP4 for
SLE-12-GA-Desktop-AMD-Driver		-	-	-	SLE-12-GA-Desktop-AMD-Driver for
SLE-12-GA-Desktop-NVIDIA-Driver		-	-	-	SLE-12-GA-Desktop-NVIDIA-Driver for
SLE-12-SP1-GA-Desktop-nVidia-Driver		-	-	-	SLE-12-SP1-GA-Desktop-nVidia-Driver for
SLE-12-SP2-GA-Desktop-nVidia-Driver		-	-	-	SLE-12-SP2-GA-Desktop-nVidia-Driver for
SLE-HA12-Debuginfo-Pool	sle-12-x86_64	-	-	-	SLE-HA12-Debuginfo-Pool for sle-12-x86_64
SLE-HA12-Debuginfo-Updates	sle-12-x86_64	-	-	-	SLE-HA12-Debuginfo-Updates for sle-12-x86_64
SLE-HA12-Pool	sle-12-x86_64	-	-	-	SLE-HA12-Pool for sle-12-x86_64
SLE-HA12-SP1-Debuginfo-Pool	sle-12-x86_64	-	-	-	SLE-HA12-SP1-Debuginfo-Pool for sle-12-x86_64
SLE-HA12-SP1-Debuginfo-Updates	sle-12-x86_64	-	-	-	SLE-HA12-SP1-Debuginfo-Updates for sle-12-x86_64
SLE-HA12-SP1-Pool	sle-12-x86_64	-	-	-	SLE-HA12-SP1-Pool for sle-12-x86_64
SLE-HA12-SP1-Source-Pool	sle-12-x86_64	-	-	-	SLE-HA12-SP1-Source-Pool for sle-12-x86_64
SLE-HA12-SP1-Updates	sle-12-x86_64	-	-	-	SLE-HA12-SP1-Updates for sle-12-x86_64
SLE-HA12-SP2-Debuginfo-Pool	sle-12-x86_64	-	-	-	SLE-HA12-SP2-Debuginfo-Pool for sle-12-x86_64
SLE-HA12-SP2-Debuginfo-Updates	sle-12-x86_64	-	-	-	SLE-HA12-SP2-Debuginfo-Updates for sle-12-x86_64
SLE-HA12-SP2-Pool	sle-12-x86_64	-	-	-	SLE-HA12-SP2-Pool for sle-12-x86_64
SLE-HA12-SP2-Source-Pool	sle-12-x86_64	-	-	-	SLE-HA12-SP2-Source-Pool for sle-12-x86_64
SLE-HA12-SP2-Updates	sle-12-x86_64	-	-	-	SLE-HA12-SP2-Updates for sle-12-x86_64
SLE-HA12-Updates	sle-12-x86_64	-	-	-	SLE-HA12-Updates for sle-12-x86_64
SLE-Likewise-5.3-Updates	sle-11-i586	-	-	-	SLE-Likewise-5.3-Updates for sle-11-i586

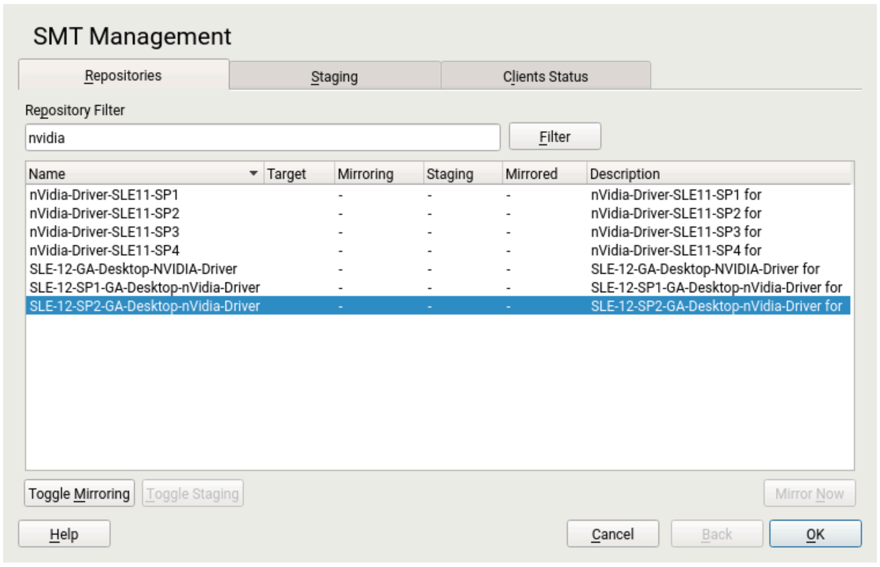
FIGURE 5.1: LIST OF REPOSITORIES

5.2 Viewing and Managing Repositories

In the *Repositories* section, you can see the list of all available package repositories for SMT. For each repository, the list shows the repository's name, target product and architecture, mirroring and staging flag, date of last mirroring, and a short description. Sort the list by clicking the desired column header, and scroll the list items using the scrollbar on the right side.

5.2.1 Filtering Repositories

You can also filter out groups of repositories using the *Repository Filter* text box. Enter the desired filter term and click *Filter* to see only the matching entries. To cancel the current filter and display all repositories, clear the *Repository Filter* field and click *Filter*.



Repository Filter: nvidia

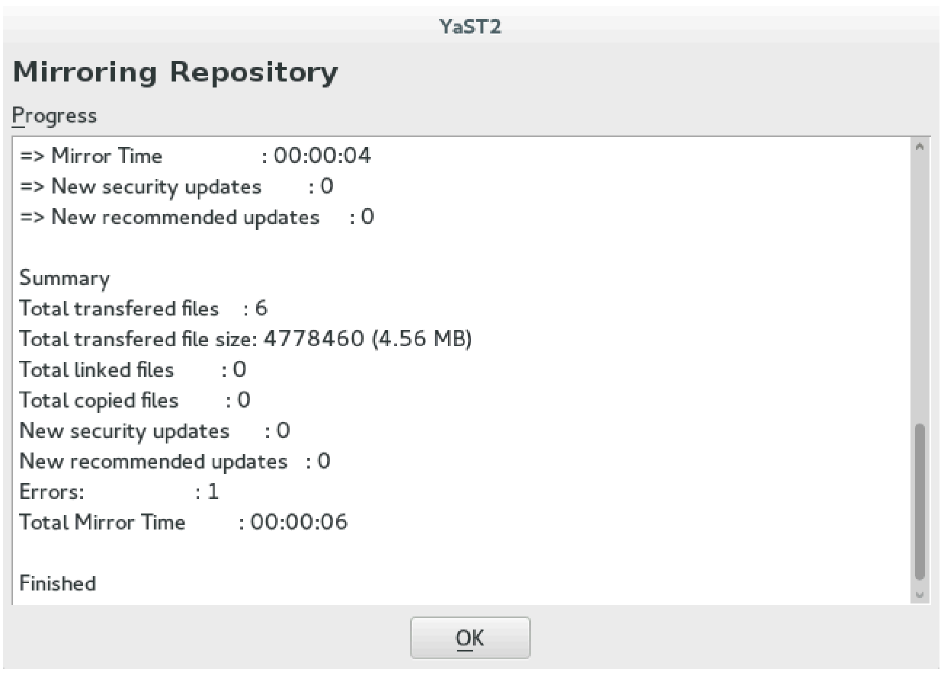
Name	Target	Mirroring	Staging	Mirrored	Description
nVidia-Driver-SLE11-SP1		-	-	-	nVidia-Driver-SLE11-SP1 for
nVidia-Driver-SLE11-SP2		-	-	-	nVidia-Driver-SLE11-SP2 for
nVidia-Driver-SLE11-SP3		-	-	-	nVidia-Driver-SLE11-SP3 for
nVidia-Driver-SLE11-SP4		-	-	-	nVidia-Driver-SLE11-SP4 for
SLE-12-GA-Desktop-NVIDIA-Driver		-	-	-	SLE-12-GA-Desktop-NVIDIA-Driver for
SLE-12-SP1-GA-Desktop-nVidia-Driver		-	-	-	SLE-12-SP1-GA-Desktop-nVidia-Driver for
SLE-12-SP2-GA-Desktop-nVidia-Driver		-	-	-	SLE-12-SP2-GA-Desktop-nVidia-Driver for

FIGURE 5.2: REPOSITORY FILTER

5.2.2 Mirroring Repositories

Before you can offer package repositories, you need to create a local mirror of their packages. To do this, follow the procedure below.

1. From the list, select the line containing the name of the repository you want to mirror.
2. Click the selected line to highlight it.
3. Click the *Toggle Mirroring* button in the lower-left part of the window. This enables the option in the *Mirroring* column of the selected repository. If the repository was already selected for mirroring, clicking the *Toggle Mirroring* button disables the mirroring.
4. Hit the *Mirror Now* button to mirror the repository.
5. A pop-up window appears with the information about mirroring status and result.
6. Click *OK* to refresh the list of repositories.



```
=> Mirror Time               : 00:00:04
=> New security updates       : 0
=> New recommended updates    : 0

Summary
Total transfered files    : 6
Total transfered file size: 4778460 (4.56 MB)
Total linked files       : 0
Total copied files       : 0
New security updates     : 0
New recommended updates  : 0
Errors:                 : 1
Total Mirror Time        : 00:00:06

Finished
```

FIGURE 5.3: STATUS OF MIRRORING PROCESS

5.3 Staging Repositories

After the mirroring is finished, you can stage the mirrored repositories. In SMT, *staging* is a process where you create either testing or production repositories based on the mirrored ones. The testing repository helps you examine the repository and its packages before you make them available in a production environment. To make repositories available for staging, follow the steps below.

1. From the repository list, select the line containing the name of the repository you want to manage.

2. Click the selected line to highlight it.

3. Click the *Toggle Staging* button next to the *Toggle Mirroring* button. This enables the option in the *Staging* column of the selected repository. If the repository was already selected for staging before, clicking the *Toggle Staging* button disables staging.

4. Repeat steps 1 to 3 for all directories you want to stage.

Important: Toggle Staging Button Not Active

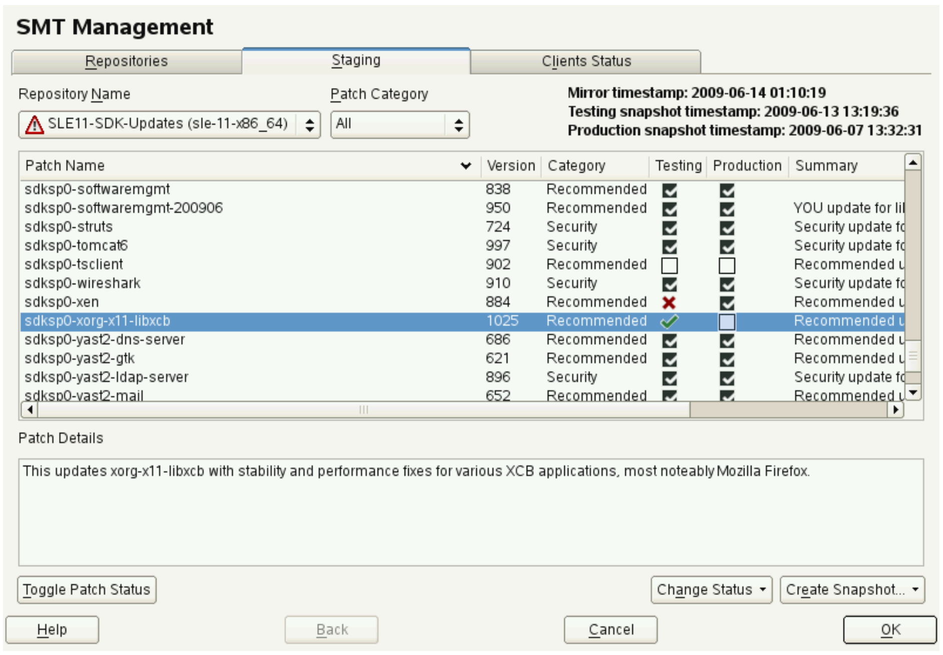
You can only stage the repositories that were previously selected for mirroring. Otherwise, the *Toggle Staging* button will disabled.

After you have mirrored the repositories and made them available for staging, click the *Staging* tab. In the upper-left part of the window, you will find the *Repository Name* drop-down box containing all repositories available for staging. The repository names there have the name of the attached staging group. Select the group you want to stage, and you should see a list of packages in this repository. For each patch, there is information about the patch name, its version and category, testing and production flags, and a short summary.

Next to the *Repository Name* drop-down box, there is a *Patch Category* filter. It can be used for listing only the patches that belong to one of the predefined categories.

If the selected repository allows for patch filtering, you can toggle the status flag for individual patches. This is done by clicking the *Toggle Patch Status* button.

Before creating a repository of packages that are available in the *production* environment, you need to create and test the *testing* repository. Select the *From Full Mirror to Testing* item from the *Create Snapshot* drop-down list. A small pop-up window appears informing you about the staging process. After the testing repository snapshot has been created, you should see the appropriate options enabled in the *Testing* column.



Mirror timestamp: 2009-06-14 01:10:19
Testing snapshot timestamp: 2009-06-13 13:19:36
Production snapshot timestamp: 2009-06-07 13:32:31

Patch Name	Version	Category	Testing	Production	Summary
sdksp0-softwaremgmt	838	Recommended	✓	✓	
sdksp0-softwaremgmt-200906	950	Recommended	✓	✓	YOU update for lil
sdksp0-struts	724	Security	✓	✓	Security update fo
sdksp0-tomcat6	997	Security	✓	✓	Security update fo
sdksp0-tsclient	902	Recommended	☐	☐	Recommended u
sdksp0-wireshark	910	Security	✓	✓	Security update fo
sdksp0-xen	884	Recommended	✗	✓	Recommended u
sdksp0-xorg-x11-libxcb	1025	Recommended	✓	☐	Recommended u
sdksp0-yast2-dns-server	686	Recommended	✓	✓	Recommended u
sdksp0-yast2-gtk	621	Recommended	✓	✓	Recommended u
sdksp0-yast2-ldap-server	896	Security	✓	✓	Security update fo
sdksp0-yast2-mail	652	Recommended	✓	✓	Recommended u

Patch Details

This updates xorg-x11-libxcb with stability and performance fixes for various XCB applications, most noteably Mozilla Firefox.

FIGURE 5.4: TESTING CREATED SNAPSHOT

Important: Creating a Production Snapshot

After you have enabled *staging* for an update repository, you need to create its *production* snapshot to make it available to the clients. Otherwise, the clients cannot find the update repository.

Select the *From Testing to Production* item from the *Create Snapshot* drop-down box. A small pop-up window appears informing you about linking the testing repository to the production one. After the production snapshot has been created, you should see the appropriate options enabled in the *Production* column. Also, a green check mark appears in the *Repository Name* drop-down box.

5.4 Jobs and Client Status Monitoring

For each client that is registered against the SMT server, SMT creates a job queue. To use the job queue, you need to install the `smt-client` package on the client. During the installation of the `smt-client` package, a cron job is created that runs the client executable **`/usr/sbin/smt-agent`** every three hours. The agent then asks the server if it has any jobs in the queue belonging to this client and executes these jobs. When there are no more jobs in the queue, the agent terminates completely. It is important to understand that jobs are not pushed directly to the clients when they get created. They are not executed until the client asks for them at the intervals specified in the cron job. Therefore, from the time a job is created on the server until it is executed on the client, a delay of several hours may occur.

Every job can have a parent job. This means that the child job only runs after the parent job has successfully finished. It is also possible to configure advanced timing and recurrence and persistence of jobs. You can find more details about SMT jobs in *Section 8.1.2.3, “smt-job”*.

When creating jobs, you need to specify the GUID of the target clients using the `-g` *`GUID`* parameter. Although the `-g` parameter can be specified multiple times in a single command, you cannot use wild cards to assign a job to all clients.

Currently, the following types of jobs are available:

Execute
: Run commands on the client

Eject
: Open, close, or toggle the CD tray of the client

Patchstatus

Report the status of installed patches

Reboot

Reboot the client

Softwarepush

Install packages

Update

Install available updates

Tip: Default Job Types

By default only `softwarepush`, `patchstatus`, and `update` jobs are allowed. To allow more types of jobs, append the job type to the `ALLOWED_AGENTS` list in `/etc/sysconfig/smt-client`.

All clients that register against the SMT server automatically get a persistent `patchstatus` job added to their job queue. This is also the case for clients without the `smt-clients` package (SUSE Linux Enterprise 10 and older, or non-SUSE based distributions). These clients appear with the `Unknown` patchstatus in the client lists. The `patchstatus` jobs for such clients are not required, and clients can safely be deleted to clean up the output of **`smt-job`**. Keep in mind that if you update a machine to SUSE Linux Enterprise 11 or later, you need to create the `patchstatus` job manually.

Whenever the client runs a `patchstatus` job, it compares the currently installed updates with what is available in the repositories on the SMT server. The job then reports back the number of missing patches that need to be installed in each of the four categories:

- Security
- Package Manager
- Recommended
- Optional

Tip: The `--agreelicense` Option

To install a package and its dependencies, the job type `softwarepush` is used. When creating this type of job, it is a good idea to use the `--agreelicense` option. If a package displays a license agreement and expects it to be accepted, the job will skip the package if `--agreelicense` is not specified. The **`smt-client`** command forwards the installation process to **`zypper`**, which does not consider a failed acceptance of a license agreement to be an error. This results in the job being completed successfully, even if the package is not installed. Using the `--agreelicense` option prevents this from happening.

5.4.1 Checking the Client Status with YaST

The *Clients Status* section of the *SMT Management* window provides the status information about all the clients that use the repositories on your SMT server. This information consists of two main parts: the list of the clients and the detailed information.

You can see the client's host name, the date and time of the last network contact with the SMT server, and its update status. The update status can be one of the following:

Up-to-date

The client packages are updated to their last version available in the production repository

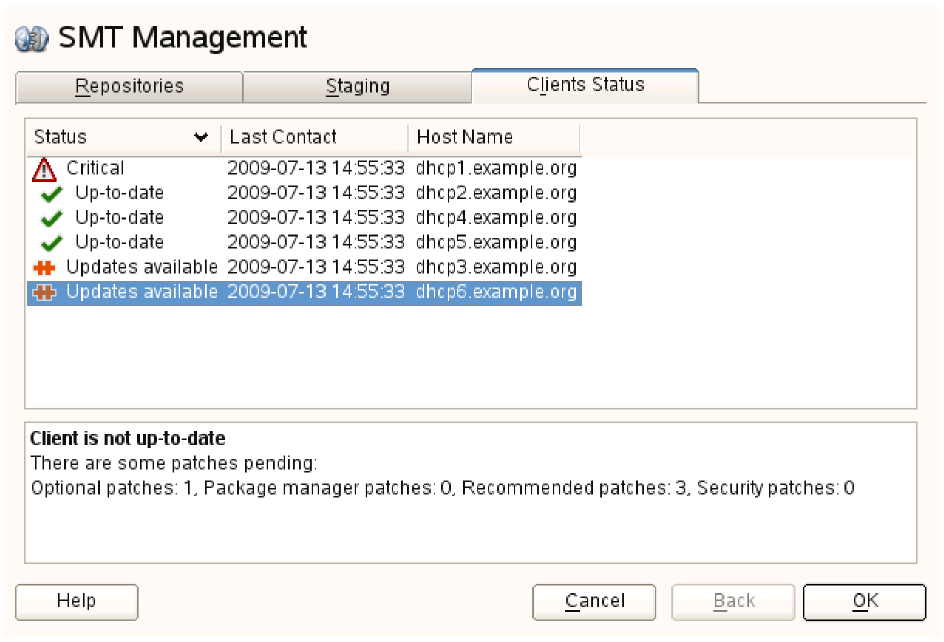
Updates available

This status means that there are updates available for the client that are either `optional` or `recommended`

Critical

Either `security` patches or `package manager` patches are available for the client

Detailed information about the selected client is available in the lower part of the window. This usually includes extended status information and detailed information about the number and types of available updates.


SMT Management

Status	Last Contact	Host Name
Critical	2009-07-13 14:55:33	dhcp1.example.org
Up-to-date	2009-07-13 14:55:33	dhcp2.example.org
Up-to-date	2009-07-13 14:55:33	dhcp4.example.org
Up-to-date	2009-07-13 14:55:33	dhcp5.example.org
Updates available	2009-07-13 14:55:33	dhcp3.example.org
Updates available	2009-07-13 14:55:33	dhcp6.example.org

Client is not up-to-date
There are some patches pending:
Optional patches: 1, Package manager patches: 0, Recommended patches: 3, Security patches: 0

FIGURE 5.5: CLIENTS STATUS

The date and time in the *Last Contact* column is the last time contact of the server—even if it only ran the regular registration update script. This date is not the date of the last 'patchstatus' report. The **`smt-client`** command-line tool prints the correct date and calls it `Patch Status Date`. The **`smt-client -v`** command prints both dates: the `patchstatus` date and the last contact of the client system.

Note: Hidden Patches

Some patches may not be visible if they are required by other patches that are only shown as available after the package manager patch or patches have been installed.

6 Managing Client Machines with SMT

SMT lets you register and manage client machines on SUSE Customer Center. Client machines must be configured to use SMT. For information about configuring clients to use SMT, see *Chapter 9, Configuring Clients to Use SMT*.

6.1 Listing Registered Clients

To list SMT-registered client machines, use the **`smt-list-registrations`** command. The following information is listed for each client: its *Unique ID*, *Hostname*, date and time of *Last Contact* with the SMT server, and the Software *Product* the client uses.

6.2 Deleting Registrations

To delete a registration from SMT and SUSE Customer Center, use the following command:

```
smt-delete-registration -g
Client_ID
```

To delete multiple registrations, the option `-g` can be used several times.

The ID of the client machine to be deleted can be determined from the output of the **`smt-list-registrations`** command.

6.3 Manual Registration of Clients at SUSE Customer Center

The **`smt-register`** command registers clients at SUSE Customer Center. This registers all unregistered clients and clients with data that changed since the last registration.

To register clients whose registration has failed, use the `--reseterror` option. This option resets the SCC registration error flag and tries to submit registrations again.

6.4 Scheduling Periodic Registrations of Clients at SUSE Customer Center

SMT module allows for the easy scheduling of client registrations. By default, registrations are scheduled to run every 15 minutes. To create or modify a new registration schedule, follow the steps below.

1. Start YaST *SMT Configuration* module (`yast2 smt-server`).
2. Go to the *Scheduled SMT Job*.
3. Select any *SCC Registration* job and click *Edit* to change its schedule.
   To create a new registration schedule, click *Add* and select *SCC Registration* as *Job to Run*.
4. Choose the *Frequency* of the scheduled SMT job. You can perform jobs *Daily*, *Weekly*, *Monthly*, or *Periodically* (every n-th hour or every m-th minute).
   Set the *Job Start Time* by entering the *Hour* and *Minute* or appropriate time periods. For weekly and monthly schedules, select the *Day of the Week* or the *Day of the Month* the mirroring should occur.

   ### Note: Lowest Registration Frequency

   Do not set the frequency lower than 10 minutes, because the maximum value of the `rndRegister` is 450 (7.5 minutes). If the frequency is lower, it may happen that the started process is still sleeping when the next process starts. This causes the second request to exit.

5. Click *OK* or *Add* and *Finish*.

Scheduling of SMT jobs in general is covered in *Section 3.5, "Setting the SMT Job Schedule with YaST"*.

YaST uses `cron` to schedule SUSE Customer Center registrations and other SMT jobs. If you prefer not to use YaST, you can use `cron` directly.

To disable automatic registration, change the `forwardRegistration` value in the `[LOCAL]` section of the `/etc/smt.conf` configuration file to `false`.

6.5 Compliance Monitoring

To assist customers in monitoring their license compliance, SMT generates a weekly report based on data from SMT and SUSE Customer Center. This report contains information about statistics of the registered machines, products used, and of the active, expiring or missing license subscriptions. If subscriptions are about to expire and/or more SUSE Linux Enterprise machines are registered than you have purchased licenses for, the report contains relevant warnings.

To calculate the compliance, the **`smt-report`** tool by default downloads information about the subscriptions and registrations (this can be disabled).

You can configure the recipient addresses for the reports in the *Database and Reporting* section of the YaST *Subscription Management Tool* configuration module. All of the e-mail configuration options are located in the `[REPORT]` section of `/etc/smt.conf` and explained in *Section 8.2.1.6, "[REPORT] Section of /etc/smt.conf"*.

The scheduling of the reports is configured in `/etc/cron.d/novell.com-smt`, while the parameters to use with the cron jobs are in the `REPORT_PARAMS` section of `/etc/smt.d/smt-cron.conf`.

Describing the content of the reports is beyond the scope of this section, but a set of reports can be split into five individual parts. By default, these reports are attached as individual files to the mail on the weekly report run. The alerts report is a normal text file while the others are in CSV format. The reports can also be created in PDF or XML by specifying `--pdf` or `--xml` as output format.

To generate a set of reports as CSV files based on local data and to display them in the standard output, run the following command:

```
smt-report --local --csv --file /root/smt-local-rep
```

Tip: Directory for Reports

The example stores the reports in the `/root` directory. You can change it to any other writable directory.

The command generates the following files:

```
/root/smt-local-rep-product_subscription_active.csv
/root/smt-local-rep-product_subscription_alerts.txt
/root/smt-local-rep-product_subscription_expired.csv
/root/smt-local-rep-product_subscription_expiresoon.csv
```

```
/root/smt-local-rep-product_subscription_summary.csv
```

Note: Multiple SMT Servers

If you have multiple SMT servers, the reports may not include all SMT servers or machines in your environment. For the complete statistics of all your registered machines, refer to the information provided by SUSE Customer Center.

For more information about types of reports, output formats, and targets refer to *Chapter 7, SMT Reports*.

7 SMT Reports

This chapter explains how to generate reports using the data from the SMT and SUSE Customer Center. These reports contain statistics of all the registered machines, products used and all active, expiring or missing subscriptions.

Note: Assignment of Reports

If you are using more than one SMT server, generated reports may not include all SMT servers or machines in your environment. For the complete statistics of all your registered machines, refer to the information in the SUSE Customer Center.

7.1 Report Schedule and Recipients

Generated SMT reports can be periodically sent to a list of specified e-mail addresses. To create or edit this list and to set the frequency of the reports, use the YaST SMT Configuration module. How to configure this list is described in *Section 3.4, "Setting E-mail Addresses to Receive Reports with YaST"*. Configuration of the report schedule is covered in *Section 3.5, "Setting the SMT Job Schedule with YaST"*.

The list can also be edited manually in the `reportEmail` part of the `/etc/smt.conf` configuration file. For more information about manually editing the list of addresses, see *Section 8.2.1.6, "[REPORT] Section of /etc/smt.conf"*. To set the frequency of reports manually, you can edit the `smt-gen-report` lines of the crontab in `/etc/cron.d/novell.com-smt`. For more information about the crontab format, see **`man 5 crontab`**.

Reports, including those generated as a scheduled SMT job, are created by the **`smt-report`** command. This command supports various parameters. To edit parameters used with scheduled commands, edit the `/etc/smt.d/smt-cron.conf` configuration file. For more information, see *Section 8.2.2, "/etc/smt.d/smt-cron.conf"*.

7.2 Report Output Formats and Targets

SMT reports can be printed to the standard output, exported to one or multiple files (in the CSV format), and mailed to a specified list of e-mail addresses. The following parameters can be used with the `smt-report` command:

`--quiet` or `-q`
: Suppress output to STDOUT and run `smt-report` in quiet mode.

`--file` or `-F`
: Export the report to one or several files. By default, the report is written to a single file, with the results formatted as tables. Optionally, the file name or whole path may be specified after the parameter: `--file FILENAME`. If no file name is specified, the default file name containing a time stamp is used. However, SMT will not check if the file or files already exist.
In the CSV (Comma-Separated Value) mode, the report is written to multiple files, therefore the specified file name expands to `[PATH/]FILENAME-reportname.extension` for every report.

`--csv` or `-c`
: The report is exported to multiple files in the CSV format. The first line of each *.csv file consists of the column names. It is recommended to use the `--csv` parameter together with the `--file` parameter. If the specified file name contains a `.csv` extension, the report format will be CSV (as if the `--csv` parameter was used).

`--mail` or `-m`
: Send the report to the addresses configured using the YaST SMT Configuration module and stored in `/etc/smt.conf`. The report is rendered as tables.

`--attach` or `-a`
: Attach the report to the mails in the CSV format. This option should only be used in combination with the `--mail` option.

`--pdf`
: The report is exported to multiple files in the PDF format.

`--xml`
: The report is exported to multiple files in the XML format.

Note: Disabling Sending Attachments

To disable sending CSV attachments with report mails, edit the `/etc/smt.d/smt-cron.conf` configuration file as follows: remove the `--attach` option from the `REPORT_PARAMS` value. The default line reads: `REPORT_PARAMS="--mail --attach -L /var/log/smt-report.log"`. To disable CSV attachments, change it to: `REPORT_PARAMS="--mail -L /var/log/smt-report.log"`.

If you have disabled CSV attachments but need them occasionally, you can send them manually with the **`smt-report --mail --attach -L /var/log/smt-report.log`** command.

8 SMT Tools and Configuration Files

This chapter describes the most important scripts, configuration files and certificates shipped with SMT.

8.1 Important Scripts and Tools

There are two important groups of SMT commands: The **`smt`** command and its sub-commands are used for managing the mirroring of updates, registration of clients, and reporting. The `systemd` `smt.target` is used for starting, stopping, restarting the SMT service and services that SMT depends on, and for checking their status.

8.1.1 SMT JobQueue

Since SUSE Linux Enterprise version 11, there is a new SMT service called SMT JobQueue for delegating *jobs* to the registered clients.

To enable JobQueue, the `smt-client` package needs to be installed on the SMT client. The client then pulls jobs from the server via a cron job (every 3 hours by default). The list of jobs is maintained on the server. Jobs are not pushed directly to the clients and processed immediately: instead, the client asks for them. Therefore, a delay of several hours may occur.

Every job can have its parent job, which sets a dependency. The child job only runs after the parent job successfully finished. Job timing is also possible: a job can have a start time and an expiration time to define its earliest execution time or the time the job expires. A job may also be persistent. It is run repeatedly with a delay. For example, a patch status job is a persistent job that runs once a day. For each client, a patch status job is automatically generated after it registers successfully against an SMT 11 server. The `patchstatus` information can be queried with the **`smt-client`** command. For already registered clients, you can add `patchstatus` jobs manually with the **`smt-job`** command.

You can edit, list, create and delete the jobs using the **`smt-job`** command-line tool. For more details on **`smt-job`**, see *Section 8.1.2.3, "smt-job"*.

Note: Overriding the Automatic Creation of Patch Status Jobs

When creating a software push or an update job, normally a non-persistent patch status job is added automatically. The parent ID is set to the ID of the new job. To disable this behavior, use the `--no-autopatchstatus` option.

SMT is not intended to be a system to directly access the clients or to immediately report the results back. It is a long-term maintenance and monitoring system rather than a live interaction tool.

Note: Job Time Lag Limitation

The client normally processes one job at a time, reports back the result, and then asks for the next job. If you create a persistent job with a time offset of only a few seconds, it is repeated forever and blocks other jobs. Therefore, adding jobs with a time offset shorter than one minute is not supported.

8.1.2 /usr/sbin/smt Commands

The key command to manage the SMT is **`smt`** (`/usr/sbin/smt`). The **`smt`** command should be used together with certain sub-commands described in this section. If the **`smt`** command is used alone, it prints a list of all available sub-commands. To get help for individual sub-commands, use **`smt`** *`SUBCOMMAND`* --help.

The following sub-commands are available:

- **`smt-client`**
- **`smt-delete-registration`**
- **`smt-job`**
- **`smt-list-products`**
- **`smt-list-registrations`**
- **`smt-mirror`**
- **`smt-scc-sync`**

- **`smt-register`**
- **`smt-report`**
- **`smt-repos`**
- **`smt-setup-custom-repos`**
- **`smt-staging`**
- **`smt-support`**
- **`smt-sync`**

There are two syntax types you can use with the **`smt`** command: **`smt`** followed by a sub-command or a single command consisting of **`smt`** followed by the dash and the desired sub-command. For example, it is possible to use either **`smt mirror`** or **`smt-mirror`**, as both have the same meaning.

Note: Conflicting Commands

Depending on your `$PATH` environment variable, the SMT **`smt`** command (`/usr/sbin/smt`) may collide with the **`smt`** command from the `star` package (`/usr/bin/smt`). Either use the absolute path `/usr/sbin/smt`, create an alias, or set your `$PATH` accordingly.

Another solution is to always use the **`smt-`** *`SUBCOMMAND`* syntax.

8.1.2.1 smt-client

The **`smt-client`** command shows information about registered clients. The information includes the following:

- guid
- host name
- patch status
- time stamps of the patch status
- last contact with the SMT server

The **`smt-client`** supports the following options:

`--verbose` or `-v`
: Shows detailed information about the client. The last contact date is shown as well.

`--debug` or `-d`
: Enables debugging mode.

`--logfile` or `-L` with the parameter *`LOGFILE`*
: Specifies the file to write the log messages to.

`--hostname` or `-h` with the parameter *`HOSTNAME`*
: Lists the entries whose host name begins with *`HOSTNAME`*.

`--guid` or `-g` with the parameter *`ID`*
: Lists the entries whose GUID is *`ID`*.

`--severity` or `-s` with the parameter *`LEVEL`*
: Filters the result by the patch status information. The value *`level`* can be one of `packagemanager`, `security`, `recommended` or `optional`.

8.1.2.2 smt-delete-registration

The **`smt-delete-registration`** command deletes one or more registrations from SMT and SUSE Customer Center. It unregisters machines from the system. The following options are available:

`--guid` or `-g` with the parameter *`ID`*
: Deletes the machine with the guid *`ID`* from the system. You can use this option multiple times.

`--debug` or `-d`
: Enables debugging mode.

8.1.2.3 smt-job

The **`smt-job`** script manages jobs for individual SMT clients. You can use this command to list, create, edit and delete jobs. The following options are available:

`--list` or `-l`
: Lists all client jobs. This is the default if the operation mode switch is omitted.

`--verbose` **or** `-v` **with the parameter** *`LEVEL`*

Shows detailed information about a job or jobs in a list mode. The *`level`* value can be a number from 0 to 3. The higher the value, the more verbose the command is.

`--create` **or** `-c`

Creates a new job.

`--edit` **or** `-e`

Edits an existing job.

`--delete` **or** `-d`

Deletes an existing job.

`--guid` **or** `-g` **with the parameter** *`ID`*

Specifies the client's guid. This parameter can be used multiple times to create a job for more than one client.

`--jobid` **or** `-j` **with the parameter** *`ID`*

Specifies the job ID. You need to specify job ID and client's guid when editing or deleting a job, as the same job for multiple clients has the same job ID.

`--deleteall` **or** `-A` *`id`*

Omit either the client's GUID or the job ID in the delete operation. The missing parameter matches all respective jobs.

`--type` **or** `-t` **with the parameter** *`TYPE`*

Specifies the job type. The type can be one of `patchstatus`, `softwarepush`, `update`, `execute`, `reboot`, `wait`, `eject`. On the client, only the following job types are enabled by default: `patchstatus`, `softwarepush` and `update`.

`--description` *`DESCRIPTION`*

Specifies a job description.

`--parent` *`ID`*

Specifies the job ID of the parent job. Use it to define a dependency. A job is not processed until its parent has successfully finished.

`--name` **or** `-n` **with the parameter** *`NAME`*

Specifies a job name.

`--persistent`
: Specifies if a job is persistent. Non-persistent jobs are processed only once, while persistent jobs are processed again and again. Use `--timelag` to define the time that elapses until the next run.

`--finished`
: Search option for finished jobs.

`--targeted` *`time`*
: Specifies the earliest execution time of a job. The job does not run exactly at that time, but a few minutes or hours after. The reason is that the client polls in a fixed interval for jobs.

`--expires` *`time`*
: Defines the time after which the job is no longer executed.

`--timelag` *`time`*
: Defines the time interval for persistent jobs.

For a complete list of available options and their explanations, see the manual page of the **`smt-job`** command (**`man smt-job`**).

8.1.2.3.1 Examples

List all finished jobs:

```
smt-job --list --finished
```

Create a `softwarepush` job that installs `xterm` and `bash` on client 12345 and 67890:

```
smt-job --create -t softwarepush -P xterm -P bash -g 12345 -g 67890
```

Change the timing for a persistent job with job ID 42 and guid 12345 to run every 6 hours:

```
smt-job --edit -j 42 -g 12345 --targeted 0000-00-00 --timelag 06:00:00
```

Delete all jobs with job ID 42:

```
smt-job --delete -jobid 42 --deleteall
```

8.1.2.4 smt-list-products

The **`smt-list-products`** script lists all software products in the SMT database. The following options are available:

`--used` or `-u`
: Shows only used products.

`--catstat` or `-c`
: Shows whether all repositories needed for a product are locally mirrored.

8.1.2.5 smt-list-registrations

The **`smt-list-registrations`** script lists all registrations. There are two options available for this command:

`--verbose` or `-v`
: Shows detailed information about the registered devices.

`--format` or `-f` with the parameter *`FORMAT`*
: Formats the output in the *`asciitable`* or *`csv`* formats.

8.1.2.6 smt-mirror

The **`smt-mirror`** command performs the mirroring procedure and downloads repositories that are set to be mirrored.

You can run the **`smt-mirror`** with the following options:

`--clean` or `-c`
: Removes all files no longer mentioned in the metadata from the mirror. No mirroring occurs before cleanup.

`--debug` or `-d`
: Enables the debugging mode.

`--deepverify`
: Turns on verifying of all package checksums.

`--hardlink` *`SIZE`*
: Searches for duplicate files with a size greater than the size specified in kilobytes. Creates hard links for them.

`--directory` *`PATH`*

Defines the directory to work on. When using this option, the default value configured in the `smt.conf` configuration file is ignored.

`--dbreplfile` *`FILE`*

Defines a path to the *.xml file to use as database replacement. You can create this file with the **`smt-scc`** command.

`--logfile` or `-L` with the parameter *`FILE`*

Specifies the path to a log file.

8.1.2.7 smt-sync

The **`smt-sync`** or **`smt sync`** command obtains data from SUSE Customer Center and updates the local SMT database. It can also save SUSE Customer Center data to a directory instead of the SMT database, or read the data from such a directory instead of downloading it from SUSE Customer Center.

For SUSE Linux Enterprise 11 clients, this script automatically determines whether Novell Customer Center or SUSE Customer Center should be used. Then **`smt-ncc-sync`** or **`smt-scc-sync`** is called. For SUSE Linux Enterprise 12 clients, only **`smt-scc-sync`** is supported.

8.1.2.8 smt-scc-sync

The **`smt scc-sync`** command obtains data from the SUSE Customer Center and updates the local SMT database. It can also save SUSE Customer Center data to a directory instead of the SMT database, or read SUSE Customer Center data from a directory instead of downloading it from SUSE Customer Center.

You can run the **`smt-scc-sync`** with the following options:

`--fromdir` *`DIRECTORY`*

Reads SUSE Customer Center data from a directory instead of downloading it from SUSE Customer Center.

`--todir` *`DIRECTORY`*

Writes SUSE Customer Center data to the specified directory without updating the SMT database.

Tip: SUSE Manager's Subscription Matching Feature

This data can be used by the subscription matching feature of SUSE Manager, which gives you a detailed overview of your subscription usage. For more information on the subscription matching feature, see https://documentation.suse.com/external-tree/en-us/suma/3.2/susemanager-reference/html/book.suma.reference.manual/ref.webui.audit.html#ref.webui.audit.subscription.

`--createdbreplacementfile`
: Creates a database replacement file for using **`smt-mirror`** without database.

`--logfile` or `-L` *`LOGFILE`*
: Specifies the path to a log file.

`--debug` or `-d`
: Enables debugging mode.

`--verboselevel` or `-v` *`BITMASK`*
: Specifies the verbosity level of log messages. Accepts a decimal number that represents a binary bitmask of one of seven verbosity levels. For example, the `--debug` option described above corresponds to the bitmask of `00111111`, which is `63` in decimal. To achieve even more verbosity, use `127`, which relates to the bitmask of `01111111`.

Important

The `--debug` and `--mail` options are mutually exclusive with the `--verboselevel` option.

8.1.2.9 smt-register

The **`smt-register`** or **`smt register`** command registers all currently unregistered clients at the SUSE Customer Center. It also registers all clients whose data has changed since the last registration.

The following options are available:

`--logfile` or `-L` with the parameter *`LOGFILE`*
: Specifies the path to a log file.

`--debug`
: Enables debugging mode.

8.1.2.10 smt-report

The **`smt-report`** or **`smt report`** command generates a subscription report based on local calculation or SUSE Customer Center registrations.

The following options are available:

`--mail` or `-m`
: Activates mailing the report to the addresses configured with the SMT Server and written in `/etc/smt.conf`. The report is formatted as tables.

`--attach` or `-a`
: Appends the report to the e-mails in CSV format. This option should only be used in combination with the `--mail` option.

`--quiet` or `-q`
: Suppresses output to STDOUT and runs **`smt-report`** in quiet mode.

`--csv` or `-c`
: Exports the report to multiple files in the CSV format. The first line of each *.csv file consists of the column names. The `--csv` parameter should only be used in combination with the `--file` parameter. If the specified file name has the `.csv` extension, the report is formatted as CSV (as if the `--csv` parameter was used).

`--pdf` or `-p`
: Exports the report in the PDF format. Use it only in combination with the `-file` option.

`--xml`
: Exports the report in the XML format. Use it only in combination with the `-file` option. For a detailed description of the XML format, see the manual page of the **`smt-report`** command.

`--file` or `-F`
: Exports the report to one or several files. By default, the report is written to a single file formatted as tables. Optionally, the file name or whole path may be specified after the `--file` *`filename`* parameter. If no file name is specified, a default file name containing a time stamp is used. However, SMT does not check if the file or files already exist.

In the CSV mode the report is written to multiple files, therefore the specified file name expands to `[PATH/]FILENAME-reportname.extension` for every report.

`--logfile` or `-L` with the parameter `LOGFILE`

Specifies path to a log file.

`--debug`

Enables debugging mode.

8.1.2.11 smt-repos

Use **`smt-repos`** (or **`smt repositories`**) to list all available repositories and for enabling, disabling and deleting repositories. The following options are available:

`--enable-mirror` or `-e`

Enables repository mirroring.

`--disable-mirror` or `-d`

Disables repository mirroring.

`--enable-by-prod` or `-p`

Enables repository mirroring by giving product data in the following format: `Product[,Version[,Architecture[,Release]]]`.

`--disable-by-prod` or `-P`

Disables repository mirroring by giving product data in the following format: `Product[,Version[,Architecture[,Release]]]`.

`--enable-staging` or `-s`

Enables repository staging.

`--disable-staging` or `-S`

Disables repository staging.

`--only-mirrorable` or `-m`

Lists only repositories that can be mirrored.

`--only-enabled` or `-o`

Lists only enabled repositories.

`--delete`

Lists repositories and deletes them from disk.

`--namespace` *`DIRNAME`*

Deletes the repository in the specified name space.

`--verbose` or `-v`

Shows detailed repository information.

8.1.2.12 smt-setup-custom-repos

The **`smt-setup-custom-repos`** and **`smt setup-custom-repos`** script are designed for setting up custom repositories (repositories not present in the download server) for use with SMT. You can use this script to add a new repository to the SMT database or to delete a repository from the database. The script supports the following options:

`--productid` *`PRODUCT_ID`*

ID of a product the repository belongs to. If a repository should belong to multiple products, use this option multiple times to assign the repository to all relevant products.

`--name` *`NAME`*

The name of the custom repository.

`--description` *`DESCRIPTION`*

The description of the custom repository.

`--exturl` *`URL`*

The URL for the repository to be mirrored from. Only HTTP and HTTPS protocols are supported.

`--delete` *`ID`*

Removes a custom repository with a given ID from the SMT database.

To set up a new repository, use the following command:

```
smt-setup-custom-repos --productid PRODUCT_ID \
--name NAME --exturl URL
```

For example:

```
smt-setup-custom-repos --productid 434 \
--name My_Catalog --exturl http://my.example.com/My_Catalog
```

To remove a configured repository, use the following command:

```
smt-setup-custom-repos --delete ID
```

For example:

```
smt-setup-custom-repos --delete 1cf336d819e8e5904f4d4b05ee081971a0cc8afc
```

8.1.2.13 smt-staging

A *patch* is an update of a package or group of packages. The term *update* and *patch* are often interchangeable. With the **`smt-staging`** script, you can set up patch filters for update repositories. It can also help you generate both testing repositories and repositories for the production environment.

The first argument of **`smt-staging`** is always the **`command`**. It must be followed by a *`repository`*. The repository can be specified by *`Name`* and *`Target`* from the table scheme returned by the **`smt-repos`** command. Alternatively, it can be specified by its `Repository ID` which can be obtained by running the command **`smt-repos -v`**. The **`smt-staging`** script supports the following commands:

`listupdates`
: Lists available patches and their allowed and forbidden status.

`allow/forbid`
: Allows or forbids specified patches.

`createrepo`
: Generates both testing and production repository with allowed patches.

`status`
: Gives information about both testing and production snapshots, and patch counts.

`listgroups`
: Lists staging groups.
: There is always one group available with the name “default”. The default group has the path `repo/full`, `repo/testing` and `repo`. New paths can be specified when creating a new group.

`creategroup`
: Creates a staging group. Required parameters are: group name, testing directory name, and production directory name.

`removegroup`
: Removes a staging group. The group name parameter is required.

The following options apply to any **`smt-staging`** command:

`--logfile` **or** `-L` *`file path`*
: Writes log information to the specified file. It is created if it does not already exist.

`--debug` **or** `-d`
: Turns on the debugging output and log.

`--verbose` **or** `-v`
: Turns more detailed output on.

The following options apply to specific **`smt-staging`** commands:

`--patch` *`PATCH_ID`*
: Specifies a patch by its ID. You can get a list of available patches with the **`listupates`** command. This option can be used multiple times. Use it with the **`allow`**, **`forbid`**, and **`listupdates`** commands. When used with **`listupdates`**, the command prints detailed information about the specified patches.

`--category` *`CATEGORY`*
: Specifies the patch category. The following categories are available: `security`, `recommended` and `optional`. Use it in combination with the **`allow`**, **`forbid`**, and **`listupdates`** commands.

`--all`
: Allows or forbids all patches in the **`allow`** or **`forbid`** commands.

`--individually`
: Allows or forbids multiple patches (for example, by category) one by one, similar to the `--patch` option used with each of the patches.

`--testing`
: Generates a repository for testing when used in combination with the **`createrepo`** command. The repository is generated from the full unfiltered local mirror of the remote repository. It is written into the `<MirrorTo>/repo/testing` directory, where `MirrorTo` is the value obtained from `smt.conf`.

`--production`

Generates a repository for production when used in combination with the **`createrepo`** command. The repository is generated from the testing repository. It is written into the `<MirrorTo>/repo` directory, where `MirrorTo` is the value obtained from `smt.conf`. If the testing repository does not exist, the production repository is generated from the full unfiltered local mirror of the remote repository.

`--group` *`GROUP`*

Specifies on which group the command should work. The default for `--group` is the name `default`.

`--nohardlink`

Prevents creating hard links instead of copying files when creating a repository with the **`createrepo`** command. If not specified, hard links are created instead.

`--nodesc`

Skips patch descriptions and summaries—to save certain screen space and make the output more readable.

`--sort-by-version`

Sorts the `listupdates` table by patch version. The higher the version, the newer the patch should be.

`--sort-by-category`

Sorts the `listupdates` table by patch category.

8.1.2.14 smt-support

The **`smt-support`** command manages uploaded support data often coming from the **`supportconfig`** tool. You can forward the data to SUSE, either selectively or in full. This command supports the following options:

`--incoming` or `-i` with the parameter *`DIRECTORY`*

Specifies the directory where the supportconfig archives are uploaded. You can also set this option with the `SMT_INCOMING` environment variable. The default `SMT_INCOMING` directory is `/var/spool/smt-support`.

`--list` or `-l`

Lists the uploaded supportconfig archives in the incoming directory.

`--remove` **or** `-r` **with the parameter** *`ARCHIVE`*

Deletes the specified archive.

`--empty` **or** `-R`

Deletes all archives in the incoming directory.

`--upload` **or** `-u` **with the parameter** *`ARCHIVE`*

Uploads the specified archive to SUSE. If you specify `-s`, `-n`, `-c`, `-p`, and `-e` options, the archive is repackaged with contact information.

`--uploadall` **or** `-U`

Uploads all archives in the incoming directory to SUSE.

`--srnum` **or** `-s` **with the parameter** *`SR_NUMBER`*

Specifies the Novell Service Request 12-digit number.

`--name` **or** `-n` **with the parameter** *`NAME`*

Specifies the first and last name of the contact, in quotes.

`--company` **or** `-c` **with the parameter** *`COMPANY`*

Specifies the company name.

`--storeid` **or** `-d` **with the parameter** *`ID`*

Specifies the store ID, if applicable.

`--terminalid` **or** `-t` **with the parameter** *`ID`*

Specifies the terminal ID, if applicable.

`--phone` **or** `-p` **with the parameter** *`PHONE`*

Specifies the phone number of the contact person.

`--email` **or** `-e` **with the parameter** *`E-MAIL_ADDRESS`*

Specifies the e-mail address of the contact.

8.1.3 SMT systemd Commands

You can manage SMT-related services with the standard `systemd` commands:

`systemctl start smt.target`

Starts the SMT services.

`systemctl stop smt.target`

Stops the SMT services.

`systemctl status smt.target`

Checks the status of the SMT services. Checks whether httpd, MariaDB, and cron are running.

`systemctl restart smt.target`

Restarts the SMT services.

`systemctl try-restart smt.target`

Checks whether the SMT services are enabled and if so, restarts them.

You can enable and disable SMT with the YaST SMT Server module.

8.2 SMT Configuration Files

The main SMT configuration file is `/etc/smt.conf`. You can set most of the options with the YaST SMT Server module. Another important configuration file is `/etc/smt.d/smt-cron.conf`, which contains parameters for commands launched as SMT scheduled jobs.

8.2.1 /etc/smt.conf

The `/etc/smt.conf` file has several sections. The `[NU]` section contains the update credentials and URL. The `[DB]` section contains the configuration of the MariaDB database for SMT. The `[LOCAL]` section includes other configuration data. The `[REPORT]` section contains the configuration of SMT reports.

Warning: Passwords in Clear Text

The `/etc/smt.conf` file contains passwords in clear text. Its default permissions (640, root, wwwrun) make its content easily accessible with scripts running on the Apache server. Be careful with running other software on the SMT Apache server. The best policy is to use this server only for SMT.

8.2.1.1 [NU] Section of /etc/smt.conf

The following options are available in the [NU] section:

`NUUrl`
: URL of the update service. Usually it should contain the `https://updates.suse.com/` URL.

`NURegUrl`
: URL of the update registration service. It is used by **`smt-sync`**. If this option is missing, the URL from `/etc/SUSEConnect` is used as a fallback.

`NUUser`
: `NUUser` should contain the user name for update service. For information about getting organization credentials, see *Section 4.1, “Mirroring Credentials”*. You can set this value with the SMT Server.

`NUPass`
: `NUPass` is the password for the user defined in `NUUser`. For information about getting organization credentials, see *Section 4.1, “Mirroring Credentials”*. You can set this value with the SMT Server.

`ApiType`
: `ApiType` is the type of service SMT uses; it can be either `NCC` for Novell Customer Center or `SCC` for SUSE Customer Center. The only supported value for SMT 12 is `SCC`.

8.2.1.2 [DB] Section of /etc/smt.conf

The three options defined in the [DB] section are used for configuring the database for SMT. Currently, only MariaDB is supported by SMT.

`config`
: The first parameter of the DBI->connect Perl method used for connection to the MariaDB database. The value should be in the form

```
dbi:mysql:database=SMT;host=LOCALHOST
```

where *`SMT`* is the name of the database and *`LOCALHOST`* is the host name of the database server.

`user`

The user for the database. The default value is `smt`.

`pass`

The password for the database user. You can set the password with the YaST SMT Server module.

8.2.1.3 [LOCAL] Section of /etc/smt.conf

The following options are available in the `[LOCAL]` section:

`url`

The base URL of the SMT server which is used to construct URLs of the repositories available on the server. This value should be set by YaST automatically during installation. The format of this option should be: `https://server.domain.tld/`.
You can change the URL manually. For example, the administrator may choose to use the `http://` scheme instead of `https://` for performance reasons. Another reason may be using an alias (configured with CNAME in DNS) instead of the host name of the server. For example, `http://smt.domain.tld/` instead of `http://server1.domain.tld/`.

`nccEmail`

E-mail address used for registration at the SUSE Customer Center. The SMT administrator can set this value with the YaST SMT Server module.

`MirrorTo`

Determines the path to mirror to.

`MirrorAll`

If the `MirrorAll` option is set to `true`, the **`smt-sync`** script will set all repositories that can be mirrored to be mirrored (DOMIRROR flag).

`MirrorSRC`

If the `MirrorSRC` option is set to `true`, source RPM packages are mirrored.

Note: Default Value Changed with SMT 11 SP2

With SMT 11 SP2, the preset default value was changed to `false`. If you also want SMT to mirror source RPM packages on new installations, set `MirrorSRC` to `true`.

Upgraded systems are not affected.

`forwardRegistration`

For SMT 11, this option determined whether the clients registered at SMT should be registered at Novell Customer Center, too. This option does not work with SUSE Customer Center yet.

`rndRegister`

Specify a delay in seconds before the clients are registered at SUSE Customer Center. The value is a random number between `0` and `450`, generated by the YaST SMT Server module. The purpose of this random delay is to prevent a high load on the SUSE Customer Center server that would occur if all smt-register cron jobs connected at the same time.

`mirror_preunlock_hook`

Specify the path to the script that will be run before the **`smt-mirror`** script removes its lock.

`mirror_postunlock_hook`

Specify the path to the script that will be run after the **`smt-mirror`** script removes its lock.

`HTTPProxy`

If you do not want to use global proxy settings, specify the proxy to be used for HTTP connection here. Use the following form: `http://`*`PROXY`*`.example.com:`*`3128`*.
If the proxy settings are not configured in `/etc/smt.conf`, the global proxy settings configured in `/etc/syconfig/proxy` are used. You can configure the global proxy settings with the YaST Proxy module.

`HTTPSProxy`

If you do not want to use global proxy settings, specify the proxy to be used for HTTPS connection here. Use the form: `https://`*`PROXY`*`.example.com:`*`3128`*.
If the proxy settings are not configured in `/etc/smt.conf`, the global proxy settings configured in `/etc/syconfig/proxy` are used. You can configure the global proxy settings with the YaST Proxy module.

`ProxyUser`

If your proxy requires authentication, specify a user name and password here, using the *`USERNAME`*`:`*`PASSWORD`* format.
If the proxy settings are not configured in `/etc/smt.conf`, the global proxy settings configured in `/etc/syconfig/proxy` are used. You can configure the global proxy settings with the YaST Proxy module.

Tip: Global User Authentication Setting

If you configure the global proxy settings with YaST, manually copy `/root/.curlrc` to the home directory of the `smt`. Adjust the permissions with the following commands as `root`:

```
cp /root/.curlrc /var/lib/smt/
chown smt:www /var/lib/smt/.curlrc
```

`requiredAuthType`

Specify an authentication type to access the repository. There are three possible types:

- `none` - no authentication is required. This is the default value.
- `lazy` - only user name and password are checked. A valid user can access all repositories.
- `strict` - checks also if the user has access to the repository.

`smtUser`

Specify a user name of a Unix user under which all smt commands will run.

`signingKeyID`

Specify the ID of the GPG key to sign modified repositories. The user specified under `smtUser` needs to have access to the key. If this option is not set, the modified repositories will be unsigned.

8.2.1.4 [REST] Section of /etc/smt.conf

The following options are available in the `[REST]` section:

`enableRESTAdminAccess`

If set to *`1`*, turns administrative access to the SMT RESTService on. Default value is *`0`*.

`RESTAdminUser`

Specify the user name that the REST-Admin uses to log in. Default value is *`RESTroot`*.

`RESTAdminPassword`

Specify the password for the REST-Admin user. The option has no default value. An empty password is invalid.

8.2.1.5 [JOBQUEUE] Section of /etc/smt.conf

The following options are available in the `[JOBQUEUE]` section:

`maxFinishedJobAge`
: Specify the maximum age of finished non-persistent jobs in days. Default value is 8.

`jobStatusIsSuccess`
: Specify a comma separated list of JobQueue status IDs that should be interpreted as successful. For more information about possible status IDs, see **`smt-job --help`**. Leaving this option empty is interpreted as default (1,4).

8.2.1.6 [REPORT] Section of /etc/smt.conf

The following options are available in the `[REPORT]` section:

`reportEmail`
: A comma separated list of e-mail addresses to send SMT status reports to. You can set this list with the YaST SMT Server module.

`reportEmailFrom`
: `From` field of report e-mails. If not set, the default `root@`*`HOSTNAME.DOMAINNAME`* will be used.

`mailServer`
: Relay mail server. If empty, e-mails are sent directly.

`mailServerPort`
: Port of the relay mail server set in `mailServer`.

`mailServerUser`
: User name for authentication to the mail server set in `mailServer`.

`mailServerPassword`
: Password for authentication to the mail server set in `mailServer`.

8.2.1.7 Example /etc/smt.conf

EXAMPLE 8.1: SMT.CONF

```
[NU]
NUUrl=https://updates.suse.com/
```

```
NURegUrl=https://scc.suse.com/connect
NUUser = exampleuser
NUPass = examplepassword
ApiType = SCC

[DB]
config = dbi:mysql:database=smt;host=localhost
user = smt
pass = smt

[LOCAL]
# Default should be http://server.domain.top/
url = http://smt.example.com/
# This email address is used for registration at SCC
nccEmail = exampleuser@example.com
MirrorTo = /srv/www/htdocs
MirrorAll = false
MirrorSRC = false
forwardRegistration = true
rndRegister = 127
# The hook script that should be called before the smt-mirror script removes its lock
mirror_preunlock_hook =
# The hook script that should be called after the smt-mirror script removed its lock
mirror_postunlock_hook =
# specify proxy settings here, if you do not want to use the global proxy settings
# If you leave these options empty the global options are used.
#
# specify which proxy you want to use for HTTP connection
# in the form http://proxy.example.com:3128
HTTPProxy =
# specify which proxy you want to use for HTTPS connection
# in the form http://proxy.example.com:3128
HTTPSProxy =
# specify username and password if your proxy requires authentication
# in the form username:password
ProxyUser =
#
# require authentication to access the repository?
# Three possible authtypes can be configured here
# 1) none   : no authentication required (default)
# 2) lazy   : check only username and password. A valid user has access to all
 repositories
# 3) strict : check also if this user has access to the repository.
#
requiredAuthType = none
#
# the smt commands should run with this unix user
```

```
#
smtUser = smt
#
# ID of the GPG key to be used to sign modified (filtered) repositories.
# The key must be accessible by the user who runs SMT, i.e. the user specified
# in the 'smtUser' configuration option.
#
# If empty, the modified repositories will be unsigned.
#
signingKeyID =
#
# This string is sent in HTTP requests as UserAgent.
# If the key UserAgent does not exist, a default is used.
# If UserAgent is empty, no UserAgent string is set.
#
#UserAgent=
# Organization credentials for this SMT server.
# These are currently only used to get list of all available repositories
# from https://your.smt.url/repo/repoindex.xml
# Note: if authenticated as a client machine instead of these mirrorUser,
# the above URL returns only repositories relevant for that client.
mirrorUser =
mirrorPassword =

[REST]
# Enable administrative access to the SMT RESTService by setting enableRESTAdminAccess=1
# default: 0
enableRESTAdminAccess = 0
# Define the username the REST-Admin uses for login
# default: RESTroot
RESTAdminUser = RESTroot
# Define the password for the REST-Admin (note: empty password is invalid)
# default: <empty>
RESTAdminPassword =

[JOBQUEUE]
# maximum age of finished (non-persistent) jobs in days
# default: 8
maxFinishedJobAge = 8
# comma separated list of JobQueue status IDs that should be interpreted as successful
# See smt-job --help for more information about possible Status IDs
# Please note: An empty string will be interpreted as default (1,4).
# default: 1,4
# useful:  1,4,6
jobStatusIsSuccess = 1,4

[REPORT]
```

```
# comma separated list of eMail addresses where the status reports will be sent to
reportEmail = exampleuser@example.com
# from field of report mails - if empty it defaults to "root@<hostname>.<domainname>"
reportEmailFrom =
# relay mail server - leave empty if mail should be sent directly
mailServer =
mailServerPort =
# mail server authentication - leave empty if not required
mailServerUser =
mailServerPassword =
```

8.2.2 /etc/smt.d/smt-cron.conf

The `/etc/smt.d/smt-cron.conf` configuration file contains options of the SMT commands launched as SMT scheduled jobs set with YaST (see *Section 3.5, "Setting the SMT Job Schedule with YaST"*). Cron is used to launch these scheduled jobs. The cron table is located in the `/etc/cron.d/novell.com-smt` file.

`SCC_SYNC_PARAMS`

Contains parameters of the **`smt scc-sync`** command, if called as part of an SMT scheduled job via cron. The default value is `"-L /var/log/smt/smt-sync.log --mail"`.

`MIRROR_PARAMS`

Contains parameters of the **`smt mirror`** command, if called as part of an SMT scheduled job via cron. The default value is `"-L /var/log/smt/smt-mirror.log --mail"` .

`REGISTER_PARAMS`

Contains parameters of the **`smt register`** command, if called as part of an SMT scheduled job via cron. The default value is `"-r -L /var/log/smt/smt-register.log --mail"` .

`REPORT_PARAMS`

Contains parameters of the **`smt report`** command, if called as part of an SMT scheduled job via cron. The default value is `"--mail --attach -L /var/log/smt/smt-report.log"` .

`JOBQUEUECLEANUP_PARAMS`

Contains parameters for smt jobqueue cleanup, if called as a part of an SMT scheduled job via cron. The default value is `"--mail -L /var/log/smt/smt-jobqueuecleanup.log"`.

8.3 Server Certificates

For communication between the SMT server and client machines, the encrypted HTTPS protocol is used, requiring a server certificate. If the certificate is not available, or if clients are not configured to use the certificate, the communication between server and clients will fail.

Every client must be able to verify the server certificate by trusting the CA (certificate authority) certificate that signed the server certificate. Therefore, the SMT server provides a copy of the CA at `/srv/www/htdocs/smt.crt`. This CA can be downloaded from every client via the URL `http://FQDN/smt.crt`. The copy is created by the `/usr/lib/SMT/bin/smt-maintenance` script. Whenever SMT is started with **`systemctl start smt.target`**, it checks the certificate. If a new CA certificate exists, it is copied again. Therefore, whenever the CA certificate is missing or changed, restart SMT using the **`systemctl restart smt.target`** command.

When the SMT Server module applies configuration changes, it checks for the existence of the common server certificate. If the certificate does not exist, YaST asks whether the certificate should be created. If the user confirms, the YaST CA Management module is started.

8.3.1 Certificate Expiration

The common server certificate SMT uses is valid for one year. After that time, a new certificate is needed. Either generate a new certificate using YaST CA Management module or import a new certificate using the YaST Common Server Certificate module. Both options are described in the following sections.

As long as the same CA certificate is used, there is no need to update certificates on the client machines. The generated CA certificate is valid for 10 years.

8.3.2 Creating a New Common Server Certificate

To create a new common server certificate with YaST, proceed as follows:

1. Start YaST and select *Security and Users* › *CA Management*. Alternatively, start the YaST CA Management module from a command line by entering **`yast2 ca_mgm`** as `root`.

2. Select the required CA and click *Enter CA*.

3. Enter the password if entering a CA for the first time. YaST displays the CA key information in the *Description* tab.

4. Click the *Certificates* tab (see *Figure 8.1, “Certificates of a CA”*) and select *Add* › *Add Server Certificate*.

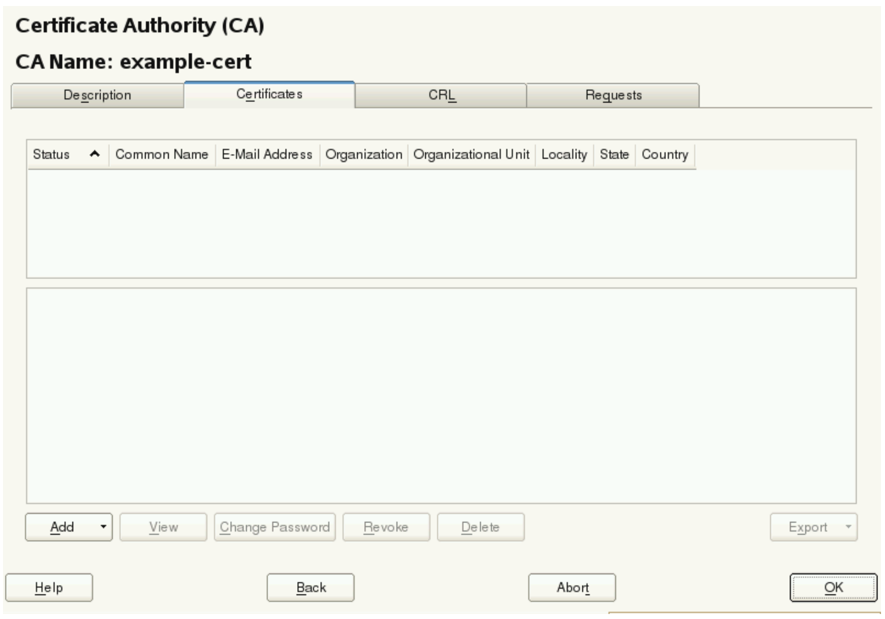


FIGURE 8.1: CERTIFICATES OF A CA

5. Enter the fully qualified domain name of the server as *Common Name*. Add a valid e-mail address of the server administrator. Other fields, such as *Organization*, *Organizational Unit*, *Locality*, and *State* are optional. Click *Next* to proceed.

Important: Host Name in Server Certificate

The server certificate must contain the correct host name. If the client requests server `https://some.hostname/`, then `some.hostname` must be part of the certificate. The host name must either be used as the *Common Name*, see *Step 5*, or as the *Subject Alternative Name*, see *Step 7*: `DNS:some.hostname` and `IP:<ipaddress>`.

6. Enter a *Password* for the private key of the certificate and re-enter it in the next field to verify it.

7. If you want to define a *Subject Alternative Name*, click *Advanced Options*, select *Subject Alternative Name* from the list and click *Add*.

Important: Subject Alternative Name

If *Subject Alternative Name* is in the server certificate, then it needs to contain the DNS entry. If *Subject Alternative Name* is present, the *Common Name* (CN) is not checked anymore.

8. If you want to keep the default values for the other options, like *Key Length* and *Valid Period*, click *Next*. An overview of the certificate to be created is shown.
9. Click *Create* to generate the certificate.
10. To export the new certificate as the common server certificate, select it in the *Certificates* tab and select *Export* › *Export as Common Server Certificate*.
11. After having created a new certificate, restart SMT using the **`systemctl restart smt.target`** command. Restarting SMT ensures that the new certificate is copied from `/etc/ssl/certs/YaST-CA.pem` to `/srv/www/htdocs/smt.crt`, the copy SMT uses. Restarting SMT also restarts the Web server.

For detailed information about managing certification and further usage of the YaST CA Management module and the Common Server Certificate module, refer to the *Security and Hardening Guide*. It is available from https://documentation.suse.com/sles/ .

8.3.3 Importing a Common Server Certificate

You can import an own common server certificate from a file. The certificate to be imported needs to be in the PKCS12 format with CA chain. Common server certificates can be imported with the YaST Common Server Certificate module.

To import an own certificate with YaST, proceed as follows:

1. Start YaST and select *Security and Users* › *Common Server Certificate*. Alternatively, start the YaST Common Server Certificate module from the command line by entering **`yast2 common_cert`** as `root`.
   The description of the currently used common server certificate is shown in the dialog that opens.
2. Click *Import* and select the file containing the certificate to be imported. Specify the certificate password in the *Password* field.

3. Click *Next*. If the certificate is successfully imported, close YaST with *Finish*.

4. After having created a new certificate, restart SMT using the **`systemctl restart smt.target`** command. Restarting SMT ensures that the new certificate is copied from `/etc/ssl/certs/YaST-CA.pem` to `/srv/www/htdocs/smt.crt`, the copy SMT uses. Restarting SMT also restarts the Web server.

8.3.4 Synchronizing Time between SMT Server and Clients

The synchronization of time between the SMT server and clients is highly recommended. Each server certificate has a validity period. If the client happens to be set to a time outside of this period, the certificate validation on the client side fails.

Therefore, it is advisable to keep the time on the server and clients synchronized. You can easily synchronize the time using NTP (network time protocol). Use **`yast2 ntp-client`** to configure an NTP client. Find detailed information about NTP in the Administration Guide.

9 Configuring Clients to Use SMT

Any machine running SUSE Linux Enterprise 10 SP4, 11 SP1 or later, or any version of SUSE Linux Enterprise 12 can be configured to register against SMT and download software updates from there, instead of communicating directly with SUSE Customer Center or Novell Customer Center.

If your network includes an SMT server to provide a local update source, you need to equip the client with the server's URL. As client and server communicate via the HTTPS protocol during registration, you also need to make sure the client trusts the server's certificate. In case you set up your SMT server to use the default server certificate, the CA certificate will be available on the SMT server at `http://FQDN/smt.crt` .

If the certificate is not issued by a well-trusted authority, the registration process will import the certificate from the URL specified as `regcert` parameter (SUSE Linux Enterprise Server 10 and 11). For SLE 12, the certificate will be downloaded automatically from SMT. In this case, the client displays the new certificate details (its fingerprint), and you need to accept the certificate.

There are several ways to provide the registration information and to configure the client machine to use SMT:

1. Provide the required information via kernel parameters at boot time (*Section 9.1, "Using Kernel Parameters to Access an SMT Server"*).

2. Configure the clients using an AutoYaST profile (*Section 9.2, "Configuring Clients with AutoYaST Profile"*).

3. Use the **`clientSetup4SMT.sh`** script (*Section 9.3, "Configuring Clients with the clientSetup4SMT.sh Script in SLE 11 and 12"*). This script can be run on a client to make it register against a specified SMT server.

4. In SUSE Linux Enterprise 11 and 12, you can set the SMT server URL with the YaST registration module during installation (*Section 9.4, "Configuring Clients with YaST"*).

These methods are described in the following sections.

9.1 Using Kernel Parameters to Access an SMT Server

Important: `regcert` Parameter Support

Note that the `regcert` kernel boot parameter is supported for SLE 10 and 11. It is not supported from SLE 12.

Any client can be configured to use SMT by providing the following kernel parameters during machine boot: `regurl` and `regcert`. The first parameter is mandatory, the latter is optional.

Warning: Beware of Typing Errors

Make sure the values you enter are correct. If `regurl` has not been specified correctly, the registration of the update source will fail.

If an invalid value for `regcert` has been entered, you will be prompted for a local path to the certificate. In case `regcert` is not specified, it will default to `http://FQDN/smt.crt` with `FQDN` being the name of the SMT server.

regurl

URL of the SMT server.

For SLE 11 and older clients, the URL needs to be in the following format: `https://FQDN/center/regsvc/` with *`FQDN`* being the fully qualified host name of the SMT server. It must be identical to the FQDN of the server certificate used on the SMT server. Example:

```
regurl=https://smt.example.com/center/regsvc/
```

For SLE 12 clients, the URL needs to be in the following format: `https://FQDN` with *`FQDN`* being the fully qualified host name of the SMT server. It must be identical to the FQDN of the server certificate used on the SMT server. Example:

```
regurl=https://smt.example.com/
```

regcert

Location of the SMT server's CA certificate. Specify one of the following locations:

URL

Remote location (HTTP, HTTPS, or FTP) from which the certificate can be downloaded. Example:

```
regcert=http://smt.example.com/smt.crt
```

Floppy

Specifies a location on a floppy. The floppy needs to be inserted at boot time—you will not be prompted to insert it if it is missing. The value needs to start with the string `floppy`, followed by the path to the certificate. Example:

```
regcert=floppy/smt/smt-ca.crt
```

Local Path

Absolute path to the certificate on the local machine. Example:

```
regcert=/data/inst/smt/smt-ca.cert
```

Interactive

Use `ask` to open a pop-up menu during installation where you can specify the path to the certificate. Do not use this option with AutoYaST. Example:

```
regcert=ask
```

Deactivate Certificate Installation

Use `done` if either the certificate will be installed by an add-on product, or if you are using a certificate issued by an official certificate authority. Example:

```
regcert=done
```

Warning: Change of SMT Server Certificate

If the SMT server gets a new certificate from an untrusted CA, the clients need to retrieve the new CA certificate file.

On SLE 10 and 11, this is done automatically with the registration process in the following cases:

- If a URL was used at installation time to retrieve the certificate.
- If the `regcert` parameter was omitted and thus the default URL is used.

If the certificate was loaded using any other method, such as floppy or local path, the CA certificate will not be updated.

On SUSE Linux Enterprise Server 12, after the certificate has changed, YaST displays a dialog for importing a new certificate. If you confirm importing the new certificate, the old one is replaced with the new one.

9.2 Configuring Clients with AutoYaST Profile

Clients can be configured to register with SMT server via AutoYaST profile. For general information about creating AutoYaST profiles and preparing automatic installation, refer to the *AutoYaST Guide*. In this section, only SMT specific configuration is described.

To configure SMT specific data using AutoYaST, follow the steps for the relevant version of SMT client.

9.2.1 Configuring SUSE Linux Enterprise 11 Clients

1. As `root`, start YaST and select *Miscellaneous* › *Autoinstallation* to start the graphical AutoYaST front-end.

   From a command line, you can start the graphical AutoYaST front-end with the **`yast2 autoyast`** command.

2. Open an existing profile using *File* › *Open*, create a profile based on the current system's configuration using *Tools* › *Create Reference Profile*, or work with an empty profile.

3. Select *Software* › *Novell Customer Center Configuration*. An overview of the current configuration is shown.

4. Click *Configure*.

5. Set the URL of the *SMT Server* and, optionally, the location of the *SMT Certificate*. The possible values are the same as for the kernel parameters `regurl` and `regcert` (see *Section 9.1, "Using Kernel Parameters to Access an SMT Server"*). The only exception is that the `ask` value for `regcert` does not work in AutoYaST, because it requires user interaction. If using it, the registration process will be skipped.

6. Perform all other configuration needed for the systems to be deployed.

7. Select *File* › *Save As* and enter a file name for the profile, such as `autoinst.xml`.

9.2.2 Configuring SUSE Linux Enterprise 12 Clients

1. As `root`, start YaST and select *Miscellaneous* › *Autoinstallation* to start the graphical AutoYaST front-end.

From a command line, you can start the graphical AutoYaST front-end with the **`yast2 autoyast`** command.

2. Open an existing profile using *File* › *Open*, create a profile based on the current system's configuration using *Tools* › *Create Reference Profile*, or work with an empty profile.
3. Select *Software* › *Product Registration*. An overview of the current configuration is shown.
4. Click *Edit*.
5. Check *Register the Product*, set the URL of the SMT server in *Use Specific Server URL Instead of the Default*, and you can set the *Optional SSL Server Certificate URL*. The possible values for the server URL are the same as for the kernel parameter `regurl`. For the SSL certificate location, you can use either HTTP or HTTPS based URLs.
6. Perform all other configuration needed for the systems to be deployed, then click *Finish* to return to the main screen.
7. Select *File* › *Save As* and enter a file name for the profile, such as `autoinst.xml`.

9.3 Configuring Clients with the clientSetup4SMT.sh Script in SLE 11 and 12

In SLE 11 and 12, the **`/usr/share/doc/packages/smt/clientSetup4SMT.sh`** script is provided together with SMT. This script allows you to configure a client machine to use an SMT server. It can also be used to reconfigure an existing client to use a different SMT server.

Note: Installation of **`wget`**

The script `clientSetup4SMT.sh` itself uses **`wget`**, so **`wget`** must be installed on the client.

Important: Upgrade **`clientSetup4SMT.sh`**

If you migrated your client OS from an older SUSE Linux Enterprise, check if the version of the **`clientSetup4SMT.sh`** script on your host is up to date. **`clientSetup4SMT.sh`** from older versions of SMT cannot manage SMT 12 clients. If you apply software patches regularly on your SMT server, you can always find the latest version of **`clientSetup4SMT.sh`** at `<SMT_HOSTNAME>/repo/tools/clientSetup4SMT.sh`.

To configure a client machine to use SMT with the **`clientSetup4SMT.sh`** script, follow these steps:

1. Copy the `clientSetup4SMT.sh` script from your SMT server to the client machine. The script is available at `<SMT_HOSTNAME>/repo/tools/clientSetup4SMT.sh` and `/srv/www/htdocs/repo/tools/clientSetup4SMT.sh`. You can download it with a browser, using **`wget`**, or by another means, such as with **`scp`**.

2. As `root`, execute the script on the client machine. The script can be executed in two ways. In the first case, the script name is followed by the registration URL. For example:

   ```
   ./clientSetup4SMT.sh https://smt.example.com/center/regsvc/
   ```

   In the second case, the script uses the `--host` option followed by the host name of the SMT server, and `--regcert` followed by the URL of the SSL certificate; for example:

   ```
   ./clientSetup4SMT.sh --host smt.example.com \
     --regcert http://smt.example.com/smt.crt
   ```

   In this case, without any “namespace” specified, the client will be configured to use the default production repositories. If `--namespace` *`GROUPNAME`* is specified, the client will use that staging group.

3. The script downloads the server's CA certificate. Accept it by pressing `Y`.

4. The script performs all necessary modifications on the client. However, the registration itself is not performed by the script.

5. The script downloads and asks to accept additional GPG keys to sign repositories with.

6. On SLE 11, perform the registration by executing **`suse_register`** or running the **`yast2 inst_suse_register`** module on the client.
   On SLE 12, perform the registration by executing

   ```
   SUSEConnect -p PRODUCT_NAME --url https://smt.example.org
   ```

   or running the **`yast2 registration`** (SUSE Linux Enterprise Server 12 SP1 and newer) or **`yast2 scc`** (SUSE Linux Enterprise Server 12) module on the client.

The **`clientSetup4SMT.sh`** script works with SUSE Linux Enterprise 10 SP2 and later Service Packs, SLE 11, and SLE 12 systems.

This script is also provided for download. You can get it by running

```
wget http://smt.example.com/repo/tools/clientSetup4SMT.sh
```

Important: Extension and Module Registration in SUSE Linux Enterprise 12

When registering an existing system against SMT 12—both on the command line and using YaST—you need to register additional extensions and modules separately, one by one. This applies both to already installed extensions and to extensions that you plan to install.

9.3.1 Problems Downloading GPG Keys from the Server

The `apache2-example-pages` package includes a `robots.txt` file. The file is installed into the Apache2 document root directory, and controls how clients can access files from the Web server. If this package is installed on the server, `clientSetup4SMT.sh` fails to download the keys stored under `/repo/keys`.

You can solve this problem by either editing `robots.txt`, or uninstalling the `apache2-example-pages` package.

If you choose to edit the `robots.txt` file, add before the `Disallow: /` statement:

```
Allow: /repo/keys
```

9.4 Configuring Clients with YaST

9.4.1 Configuring Clients with YaST in SLE 11

To configure a client to perform the registration against an SMT server use the YaST registration module (**`yast2 inst_suse_register`**).

Click *Advanced* › *Local Registration Server* and enter the name of the SMT server plus the path to the registration internals (`/center/regsvc/`), for example:

```
https://smt.example.com/center/regsvc/
```

After confirmation the certificate is loaded and the user is asked to accept it. Then continue.

Warning: Staging Groups Registration

If a staging group is used, make sure that settings in `/etc/suseRegister.conf` are done accordingly. If not already done, modify the `register=` parameter and append `&namespace=NAMESPACE`. For more information about staging groups, see *Section 5.3, “Staging Repositories”*.

Alternatively, use the **`clientSetup4SMT.sh`** script (see *Section 9.3, “Configuring Clients with the clientSetup4SMT.sh Script in SLE 11 and 12”*).

9.4.2 Configuring Clients with YaST in SLE 12

To configure a client to perform the registration against an SMT server use the YaST *Product Registration* module **`yast2 registration`** (SUSE Linux Enterprise Server 12 SP1 or newer) or **`yast2 scc`** (SUSE Linux Enterprise Server 12).

On the client, the credentials are not necessary and you may leave the relevant fields empty. Click *Local Registration Server* and enter its URL. Then click *Next* until the exit from the module.

9.5 Registering SLE11 Clients against SMT Test Environment

To configure a client to register against the test environment instead of the production environment, modify `/etc/suseRegister.conf` on the client machine by setting:

```
register = command=register&namespace=testing
```

For more information about using SMT with a test environment, see *Section 4.5, “Using the Test Environment”*.

9.6 Registering SLE12 Clients against SMT Test Environment

To register a client in the testing environment, follow these steps:

1. De-register the client from the SMT server by running **`SUSEConnect --de-register`** on the client host.

2. Modify `/etc/SUSEConnect` on the client machine as follows:

   ```
   namespace: testing
   ```

3. Re-register the client host against SMT in order for the new namespace setting to take effect. See general information about registering SMT clients in *Chapter 9, Configuring Clients to Use SMT*.

For more information about using SMT with a test environment, see *Section 4.5, “Using the Test Environment”*.

9.7 Listing Accessible Repositories

To retrieve the accessible repositories for a client, download `repo/repoindex.xml` from the SMT server with the client's credentials. The credentials are stored in `/etc/zypp/credentials.d/SCCcredentials` (SUSE Linux Enterprise Server 12) or `/etc/zypp/credentials.d/NCCcredentials` (SUSE Linux Enterprise Server 11) on the client machine. Using **`wget`**, the command for testing could be as follows:

```
wget https://USER:PASS@smt.example.com/repo/repoindex.xml
```

`repoindex.xml` returns the complete repository list as they come from the vendor. If a repository is marked for staging, `repoindex.xml` lists the repository in the `full` namespace (`repos/full/$RCE`).

To get a list of all repositories available on the SMT server, use the credentials specified in the `[LOCAL]` section of `/etc/smt.conf` on the server as `mirrorUser` and `mirrorPassword`.

9.8 Online Migration of SUSE Linux Enterprise Clients

SUSE Linux Enterprise clients registered against SMT can be migrated online to the latest service pack of the same major release the same way as clients registered against SUSE Customer Center or Novell Customer Center. Before starting the migration, make sure that SMT is configured to provide the correct version of repositories to which you need the clients to migrate.

For detailed information on online migration, see https://documentation.suse.com/sles-11/html/SLES-all/cha-update-sle.html ↗ for SUSE Linux Enterprise 11 clients, or *Book “Deployment Guide”, Chapter 20 “Upgrading SUSE Linux Enterprise”* for SUSE Linux Enterprise 12 clients.

9.9 How to Update Red Hat Enterprise Linux with SMT

SMT enables customers that possess the required entitlements to mirror updates for Red Hat Enterprise Linux (RHEL). Refer to http://www.suse.com/products/expandedsupport/ for details on SUSE Linux Enterprise Server Subscription with Expanded Support. This section discusses the actions required to configure the SMT server and clients (RHEL servers) for this solution.

Note: SUSE Linux Enterprise Server 10

Configuring RHEL client with Subscription Management Tool for SUSE Linux Enterprise (SMT 1.0) running SUSE Linux Enterprise Server 10 is slightly different. For more information, see How to update Red Hat Enterprise Linux with SMT. (http://www.novell.com/support/search.do?usemicrosite=true&searchString=7001751)

9.9.1 How to Prepare SMT Server for Mirroring and Publishing Updates for RHEL

1. Install SUSE Linux Enterprise Server (SLES) with the SMT packages as per the documentation on the respective products.
2. During SMT setup, use organization credentials that have access to Novell-provided RHEL update repositories.
3. Verify that the organization credentials have access to download updates for the Red Hat products with

   ```
   smt-repos -m | grep RES
   ```

4. Enable mirroring of the RHEL update repositories for the desired architecture(s):

   ```
   smt-repos -e REPO-NAME ARCHITECTURE
   ```

5. Mirror the updates and log verbose output:

   ```
   smt-mirror -d -L /var/log/smt/smt-mirror.log
   ```

The updates for RHEL will also be mirrored automatically as part of the default nightly SMT mirroring cron job. When the mirror process of the repositories for your RHEL products has completed, the updates are available via

```
http://smt-server.your-domain.top/repo/$RCE/REPOSITORY_NAME/ARCHITECTURE/
```

6. To enable GPG checking of the repositories, the key used to sign the repositories needs to be made available to the RHEL clients. This key is now available in the res-signingkeys package, which is included in the SMT 11 installation source.

   - Install the `res-signingkeys` package with the command

     ```
     zypper in -y res-signingkeys
     ```

   - The installation of the package stores the key file as `/srv/www/htdocs/repo/keys/res-signingkeys.key`.

   - Now the key is available to the clients and can be imported into their RPM database as described later.

9.9.2 How to Configure the YUM Client on RHEL 5.2 to Receive Updates from SMT

1. Import the repository signing key downloaded above into the local RPM database with

   ```
   rpm --import http://smt.example.com/repo/keys/res-signingkeys.key
   ```

2. Create a file in `/etc/yum.repos.d/` and name it `RES5.repo`.

3. Edit the file and enter the repository data, and point to the repository on the SMT server as follows:

   ```
   [smt]
   name=SMT repository
   baseurl=http://smt.example.com/repo/$RCE/REPOSITORY_NAME/ARCHITECTURE/
   enabled=1
   gpgcheck=1
   ```

   Example of base URL:

   ```
   http://smt.mycompany.com/repo/$RCE/RES5/i386/
   ```

4. Save the file.
5. Disable standard Red Hat repositories by setting

   ```
   enabled=0
   ```

   in the repository entries in other files in `/etc/yum.repos.d/` (if any are enabled).
   Both YUM and the update notification applet should work correctly now and notify of available updates when applicable. You may need to restart the applet.

9.9.3 How to Configure the UP2DATE Client on RHEL 3.9 and 4.7 to Receive Updates from SMT

1. Import the repository signing key downloaded above into the local RPM database with

   ```
   rpm --import http://smt.example.com/repo/keys/res-signingkeys.key
   ```

2. Edit the file `/etc/sysconfig/rhn/sources` and make the following changes:
3. Comment out any lines starting with `up2date`.
   Normally, there will be a line that says "up2date default".
4. Add an entry pointing to the SMT repository (all in one line):

   ```
   yum REPO_NAME http://smt.example.com/repo/$RCE/REPOSITORY_NAME/ARCHITECTURE/
   ```

   where `repo-name` should be set to RES3 for 3.9 and RES4 for 4.7.
5. Save the file.

Both up2date and the update notification applet should work correctly now, pointing to the SMT repository and indicating updates when available. In case of trouble, try to restart the applet.

To ensure correct reporting of the Red Hat Enterprise systems in SUSE Customer Center, they need to be registered against your SMT server. For this a special suseRegisterRES package is provided through the RES* repositories and it should be installed, configured and executed as described below.

9.9.4 How to Register RHEL 5.2 against SMT

1. Install the suseRegisterRES package.

```
yum install suseRegisterRES
```

Note: Additional Packages

You may need to install the `perl-Crypt-SSLeay` and `perl-XML-Parser` packages from the original RHEL media.

2. Copy the SMT certificate to the system:

   ```
   wget http://smt.example.com/smt.crt
   ```

   ```
   cat smt.crt >> /etc/pki/tls/cert.pem
   ```

3. Edit `/etc/suseRegister.conf` to point to SMT by changing the URL value to

   ```
   url: https://smt.example.com/center/regsvc/
   ```

4. Register the system:

   ```
   suse_register
   ```

9.9.5 How to Register RHEL 4.7 and RHEL 3.9 against SMT

1. Install the `suseRegisterRES` package:

   ```
   up2date --get suseRegisterRES
   up2date --get perl-XML-Writer
   rpm -ivh /var/spool/up2date/suseRegisterRES*.rpm /var/spool/up2date/perl-XML-
   Writer-0*.rpm
   ```

   ### Note: Additional Packages

   You may need to install the `perl-Crypt-SSLeay` and `perl-XML-Parser` packages from the original RHEL media.

2. Copy the SMT certificate to the system:

   ```
   wget http://smt.example.com/smt.crt
   ```

   ```
   cat smt.crt >> /usr/share/ssl/cert.pem
   ```

3. Edit `/etc/suseRegister.conf` to point to SMT by changing the URL value to

   ```
   url = https://smt.example.com/center/regsvc/
   ```

   or (for SUSE Customer Center)

   ```
   url = https://smt.example.com
   ```

4. Register the system:

   ```
   suse_register
   ```

10 Advanced Topics

This chapter covers usage scenarios beyond the regular workflow to give you more control over your SMT server.

10.1 Backup of the SMT Server

Creating backups of the SMT server regularly can help restore it quickly and reliably if the server fails.

There are three main parts on the SMT server to back up:

- Configuration files
- Package repositories
- The database

10.1.1 Configuration Files and Repositories

The SMT server configuration is stored in the `/etc/smt.conf` file and files in the `/etc/smt.d` directory.

As SMT depends on the services provided by the Apache Web server and MariaDB database engine, you need to back up their configuration files as well. Apache configuration files are located in the `/etc/apache2` directory, while configuration of MariaDB is stored in `/etc/my.cnf`, `/etc/mysqlaccess.conf`, and files in the `/etc/my.cnf.d` directory.

Package repositories are stored in the `/srv/www/htdocs/repo` directory. While you can normally mirror the repositories on the restored server from the update server as well, the download can take a long time. Therefore backing up the repositories can save you time and bandwidth. Moreover, backing up the repositories is necessary if you are using repository staging and want to restore the snapshots of the repositories (see *Section 4.6, "Testing and Filtering Update Repositories with Staging"*).

Warning: Size of the Repositories

The software repositories can be significant in size, and you will need to transfer them from the update server.

Use your preferred tool to back up the configuration and repository files.

10.1.2 The Database

SMT uses the MariaDB database to store information about clients, registrations, machine data, which repositories are enabled for mirroring, and custom repositories. Unlike the configuration files and repositories, the database information cannot be recovered without a valid backup.

To back up the SMT database, you can for example create a cron job that performs an SQL dump into a plain text file:

```
mysqldump -u root -p SMT_DB_PASSWORD smt > /BACKUP_DIR/smt-db-backup.sql
```

Then add the resulting file to your regular backup jobs.

10.2 Disconnected SMT Servers

In some restricted environments it is not possible for SMT servers to access the Internet because they are located on disconnected or isolated networks. In this case, you can back up the relevant data on an external storage device using special parameters with the SMT commands.

You need an external SMT server that mirrors the repositories from SUSE Customer Center. Then you can transfer these repositories to the SMT servers on the isolated network using the external storage device.

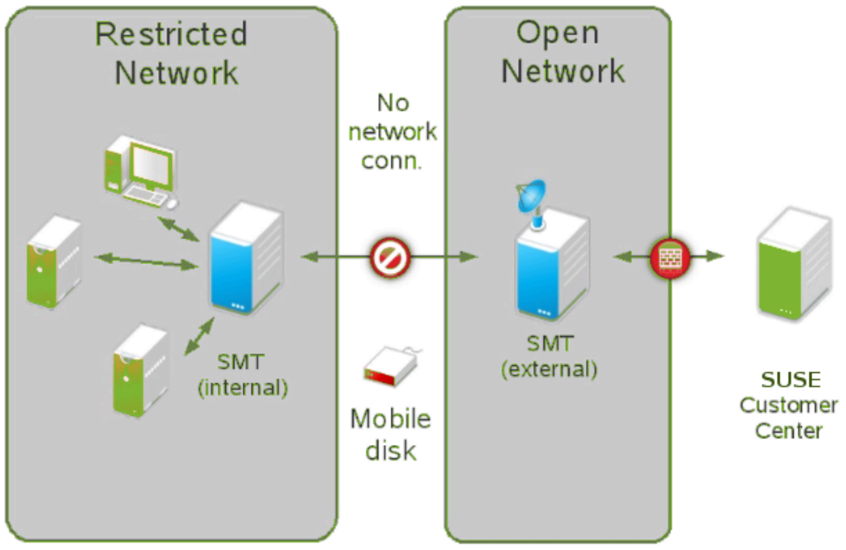


FIGURE 10.1: SMT DISCONNECTED SETUP

Although the initial setup of this solution requires additional configuration, the regular update synchronization with SUSE Customer Center and distribution to isolated servers is simple. The steps required during the initial setup are as follows:

- Installing and configuring the external SMT server
- Installing the internal server
- Editing `/etc/smt.conf` and setting up a cron job on the internal SMT server
- Transferring the SUSE Customer Center data from the external SMT server to the internal server
- Enabling and disabling repositories on the internal server
- Creating an SMT database replacement file on the internal server—when performing mirror jobs, this file can be used instead of the normal MariaDB database

Day-to-day operation requires the following actions:

- Running the **`smt-mirror`** job on the external server
- Synchronizing the mirrored repositories from the external storage device to the internal SMT server

Below is a detailed description of the individual steps.

PROCEDURE 10.1: EXTERNAL SMT SERVER CONFIGURATION FOR THE DISCONNECTED SETUP

1. Install and configure SMT as described in *Chapter 2, SMT Installation*.
2. Enable the repositories for use by the internal SMT servers.
3. Perform a standard repository mirroring from SUSE Customer Center with **`smt-mirror`**.
4. Attach a removable storage device to the server and mount it.
5. Export the required SUSE Customer Center data to a directory on the mounted storage device:

    a. Create a directory with correct permissions for storing the data. Because the **`smt`** commands run as the `smt` user (whose numeric UID can differ between the servers), you need to make permissions for the directories on the external storage device less restrictive:

    ```
    chmod o+w /path/to/scc/dir/on/storage/device
    ```

   b. Export the SUSE Customer Center data:

      ```
      smt-sync --todir /path/to/scc/dir/on/storage/device
      ```

6. Create a directory with correct permissions:

   ```
   mkdir /path/to/repository/on/storage/device
   chmod o+w /path/to/repository/on/storage/device
   ```

7. Unmount and detach the storage device.

PROCEDURE 10.2: INTERNAL SMT SERVER CONFIGURATION FOR THE DISCONNECTED SETUP

1. Ensure you have a working SUSE Linux Enterprise Server installation source.

2. Install SMT the same way as on the external server with the following exceptions:

   a. Start the *SMT Wizard*:

      ```
      tux > sudo yast2 smt-wizard
      ```

      The first step of the wizard shows the *Customer Center Configuration*.

   b. In the *User* and *Password* text boxes, enter random strings (the boxes must not be left empty).

   c. Set up the database, SSL certificate and everything else as you would normally do.

   d. Finish the *SUSE Customer Center Configuration* wizard.

   e. In the final step of the wizard, *Writing SMT Configuration*, ignore the following error message:

      ```
      Running the synchronization script failed
      ```

3. Re-launch the YaST Subscription Management Tool Server Configuration module (**`yast2 smt-server`**) and go to the *Scheduled SMT Jobs* tab.

4. Delete `SCC Registration` and `Synchronization of Updates` jobs.

5. Click *OK* to finish the wizard, provide the SMT user password, and acknowledge the synchronization error again.

6. Prevent registration data upstream synchronization to SUSE Customer Center by setting

```
forwardRegistration = false
```

in `/etc/smt.conf`.

7. Connect an external storage device and mount it.
8. Populate the SMT database with the previously created SUSE Customer Center data:

   ```
   smt-sync --fromdir /path/to/scc/dir/on/mobile/disk
   ```

9. Enable mirroring of the desired repositories using the **`smt-repos -e`** command.
10. Create a database replacement file on the external storage device:

    ```
    smt-sync --createdbreplacementfile /path/to/dbrepl/file/on/mobile/disk
    ```

11. Unmount and detach the storage device.

Now the configuration of both the external and internal SMT servers is complete. However, the update repository is still empty. After you run the following daily operation routines for the first time, the repository will be synchronized, and the internal SMT server will be ready to serve clients.

PROCEDURE 10.3: DAILY EXTERNAL SMT SERVER OPERATION

1. Connect an external storage device and mount it.
2. Perform a mirror to a directory on the storage device based on the file stored on it:

   ```
   smt-mirror --dbreplfile /path/to/dbrepl/file/on/storage/device/filename.xml \
    --fromlocalsmt --directory /path/to/repository/on/storage/device \
    -L /var/log/smt/smt-mirror-example.log
   ```

3. Update the database on the storage device with the product and subscription info from SUSE Customer Center:

   ```
   smt-sync --todir /path/to/scc/dir/on/storage/device
   ```

4. Optionally, scan the storage device for viruses and other unwanted content.
5. Unmount and disconnect the storage device.

PROCEDURE 10.4: DAILY INTERNAL SMT SERVER OPERATION

1. Connect a storage device and mount it.

2. Update the SUSE Customer Center data on the server:

```
smt-sync --fromdir /path/to/scc/dir/on/storage/device
```

3. Mirror from the storage device to the server:

```
smt-mirror --fromdir /path/to/repository/on/storage/device
```

4. Update the SUSE Customer Center data on the storage device with local changes in the mirror status since the last synchronization:

```
smt-sync --createdbreplacementfile /path/to/dbrepl/file/on/storage/device/
filename.xml
```

5. Unmount and disconnect the storage device.

A SMT REST API

The SMT REST interface is meant for communication with SMT clients and integration into other Web services. The base URI for all the following REST calls is `https://YOURSMTSERVER/=/1`. The SMT server responds with XML data described for each call by an RNC snippet with comments.

Quick Reference

Note: API for authenticating SMT clients.

Used internally in the `smt-client` package. Not intended for general administrative use!

```
GET /jobs                          get list of all jobs for client
GET /job/@next                     get the next job for client
GET /job/<jobid>                   get job with jobid for client.
                                   Note: this marks the job as retrieved
PUT /job/<jobid>                   update job having <jobid> using XML data.
                                   Note: updates only retrieved jobs
```

For backward compatibility reasons, the following are also available:

```
GET /jobs/@next                    same as GET /job/@next
GET /jobs/<jobid>                  same as GET /job/<jobid>
PUT /jobs/<jobid>                  same as PUT /job/<jobid>
```

API for general access (this needs authentication using credentials from the `[REST]` section of `smt.conf`).

```
GET /client                        get data of all clients
GET /client/<GUID>                 get data of client with specified GUID
GET /client/<GUID>/jobs            get client's job data
GET /client/<GUID>/patchstatus     get client's patch status
GET /client/<GUID>/job/@next       get client's next job
GET /client/<GUID>/job/<jobid>     get specified client job data
GET /client/@all/jobs              get job data of all clients
GET /client/@all/patchstatus       get patch status of all clients
GET /repo                          get all repositories known to SMT
GET /repo/<repoid>                 get details of repository with <repoid>
GET /repo/<repoid>/patches         get repository's patches
GET /patch/<patchid>                   get patch <patchid> details
GET /product                           get list of all products known to SMT
GET /product/<productid>               get details of product with <productid>
GET /product/<productid>/repos         get list of product's repositories
```

For backward compatibility reasons, plural forms are also available; for example:

```
GET /clients                           same as GET /client
GET /repos                             same as GET /repo
GET /product                           same as GET /product
```

Detailed Description

API for authenticating clients:

`GET /jobs`

Get list of all jobs for an authenticating client. When getting the jobs via this path they will not be set to the status retrieved.

Example:

```
<jobs>
  <job name="Patchstatus Job" created="2010-06-18 16:34:38"
 description="Patchstatus Job for Client 456" exitcode="" expires=""
 finished="" guid="456" guid_id="30" id="31" message="" parent_id=""
 persistent="1" retrieved="" status="0" stderr="" stdout="" targeted=""
 timelag="23:00:00" type="1" verbose="0">
    <arguments></arguments>
  </job>
  <job name="Software Push" created="2010-06-18 16:37:59" description="Software
 Push: mmv, whois" exitcode="" expires="" finished="" guid="456" guid_id="30"
 id="32" message="" parent_id="" persistent="0" retrieved="" status="0"
 stderr="" stdout="" targeted="" timelag="" type="2" verbose="0">
    <arguments>
      <packages>
        <package>mmv</package>
        <package>whois</package>
      </packages>
    </arguments>
  </job>
  <job name="Update Job" created="2010-06-18 16:38:39" description="Update Job"
 exitcode="" expires="" finished="" guid="456" guid_id="30" id="34" message=""
 parent_id="" persistent="0" retrieved="" status="0" stderr="" stdout=""
 targeted="" timelag="" type="3" verbose="0">
    <arguments></arguments>
  </job>
  <job name="Execute" created="2010-06-18 17:40:10" description="Execute
 custom command" exitcode="0" expires="" finished="2010-06-18 17:40:14"
 guid="456" guid_id="30" id="41" message="execute successfully finished"
 parent_id="" persistent="0" retrieved="2010-06-18 17:40:14" status="1"
 stderr="man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash" stdout=""
 targeted="" timelag="" type="4" verbose="1">
   <arguments command="grep man /etc/passwd" />
  </job>
  <job name="Reboot" created="2010-06-18 16:40:28" description="Reboot now"
 exitcode="" expires="2011-06-12 15:15:15" finished="" guid="456" guid_id="30"
 id="37" message="" parent_id="" persistent="0" retrieved="" status="0"
 stderr="" stdout="" targeted="2010-06-12 15:15:15" timelag="" type="5"
 verbose="0">
    <arguments></arguments>
  </job>
  <job name="Wait 5 sec. for exit 0." created="2010-06-18 16:40:59"
 description="Wait for 5 seconds and return with value 0." exitcode=""
 expires="" finished="" guid="456" guid_id="30" id="38" message="" parent_id=""
 persistent="0" retrieved="" status="0" stderr="" stdout="" targeted=""
 timelag="" type="7" verbose="0">
    <arguments exitcode="0" waittime="5" />
  </job>
  <job name="Eject job" created="2010-06-18 16:42:00" description="Job to eject
 the CD/DVD drawer" exitcode="" expires="" finished="" guid="456" guid_id="30"
 id="39" message="" parent_id="" persistent="0" retrieved="" status="0"
 stderr="" stdout="" targeted="" timelag="" type="8" verbose="0">
    <arguments action="toggle" />
  </job>
</jobs>
```

`GET /jobs/@next`

Get the next job for an authenticating client. The job will not be set to the retrieved status.

Example:

```
<job id="31" guid="456" type="patchstatus" verbose="false">
  <arguments></arguments>
</job>
```

`GET /jobs/<jobid>`

Get a job with the specified jobid for an authenticating client. The job will be set to the retrieved status.

When the client retrieves a job, not all the metadata is part of the XML response. However, it can be the full set of metadata, as **`smt-client`** only picks the data that is relevant. But a job retrieval should only contain the minimal set of data that is required to fulfill it.

RNC:

```
start = element job {
  attribute id {xsd:integer},         # the job ID. A job id alone is not
 unique.
                                      # A job is only uniquely identified with
                                      # guid and id. The same jobs for multiple
                                      # clients have the same job id.
  attribute parent_id {xsd:integer}?, # ID of the job on which this job depends
  attribute guid {xsd:string},
  attribute guid_id {xsd:integer}?,   # internal database ID of the client
                                      # (for compatibility reasons, if third
                                      # party application talks to SMT REST
                                      # service).
  attribute type {                    # job type ID string. Must be unique and
                                      # equal to the name of the Perl module on
                                      # the client.
    "softwarepush",
    "patchstatus",
    "<custom>"                        # add your own job types
  },
  attribute name {xsd:string},        # short custom name of the job, user-
defined
  attribute description {xsd:string}, # custom description of what the job does
  attribute created {xsd:string},     # time stamp of creation
  attribute expires {xsd:string},     # expiration time stamp; the job expires
                                      # if not retrieved by then
  attribute finished {xsd:string},    # time stamp of job completion
  attribute retrieved {xsd:string},   # time stamp of retrieval of the job
  attribute persistent {xsd:boolean}?, # defines whether the job is a
 persistent
                                      # (repetitive) job
  attribute verbose {xsd:boolean},    # if true, output of job commands is
                                      # attached to the result
  attribute exitcode {xsd:integer},   # the last exit code of the system
 command
                                      # executed to complete the job
  attribute message {xsd:string},     # custom human-readable message the
 client
                                      # sends back as a result
  attribute status {                  # logical status of the job
    0,     # not yet worked on: The job may be already retrieved but no
           # result was sent back yet.
    1,     # success: The job was retrieved, processed and the client sent
           # back a success response.
    2,     # failed: The job was retrieved, processed and the client sent
           # back a failure response.
    3},    # denied by client: The job was retrieved but could not be
           # processed as the client denied to process this job type
           # (a client needs to allow all job types that should be processed,
           # any other will be denied).
  attribute stderr {text},            # standard error output of jobs's system
                                      # commands (filled if verbose)
  attribute stdout {text},            # standard output of jobs's system
                                      # commands (filled if verbose)
  attribute targeted {xsd:string},    # time stamp when this job will be
                                      # delivered at the earliest
  attribute timelag {xsd:string}?,    # interval time of a persistent job in
                                      # the format "HH:MM:SS" (HH can be
                                      # bigger than 23)
  element-arguments                   # job-type-specific XML data
}
```

Example (minimal job definition for a 'softwarepush' job):

```
<job id="32" guid="456" type="softwarepush" verbose="false">
  <arguments>
    <packages>
      <package>mmv</package>
      <package>whois</package>
    </packages>
  </arguments>
</job>
```

`PUT /job/<jobid>`

Update a job for an authenticating client using XML data.

A client can only send job results for jobs properly retrieved previously. The jobs will be set to status done (except for persistent jobs, in which case a new target time will be computed).

Examples:

- Example for a successful patchstatus job:

  ```
  <job id="31" guid="abc123" exitcode="0" message="0:0:0:0 #
   PackageManager=0 Security=0 Recommended=0 Optional=0" status="1"
   stderr="" stdout="" />
  ```

- Example for a failed softwarepush:

  ```
  <job id="32" guid="abc123" exitcode="104" message="softwarepush failed"
   status="2" stderr="" stdout="" />
  ```

- Example for a successful update:

```
<job id="34" guid="abc123" exitcode="0" message="update successfully
 finished" status="1" stderr="" stdout="" />
```

- Example for a successful reboot job:

```
<job id="37" guid="abc123" exitcode="0" message="reboot triggered"
 status="1" stderr="" stdout="" />
```

- Execute for a successful wait job:

```
<job id="38" guid="abc123" exitcode="0" message="wait successfully
 finished" status="1" stderr="" stdout="" />
```

- Example for a successful eject job:

```
<job id="39" guid="abc123" exitcode="0" message="eject successfully
 finished" status="1" stderr="" stdout="" />
```

- Example for a successful execute job:

```
<job id="41" guid="abc123" exitcode="0" message="execute successfully
 finished" status="1" stderr="man:x:13:62:Manual pages viewer:/var/cache/
man:/bin/bash" stdout="" />
```

API for general access:

`GET /repo/<repoid>`

Returns detailed information about the specified repository. The `<repoid>` can be obtained using the `/repos` or `/products/<productid>/repos/` call.

RNC:

```
start = element repo {                       # repository
  attribute id {xsd:integer},                # SMT ID of the repository
  attribute name {xsd:string},               # repository's Unix name
  attribute target {xsd:string},             # repository's target product
  attribute type {"nu" | "yum" | "zypp" | "pum"}, # type of repository
  element description {xsd:string},          # description of the repository
  element localpath {xsd:string},            # path to local SMT mirror of the
                                             # repository
  element url {xsd:anyURI},                  # original URL of the repository
  element mirrored {
    attribute date {xsd:integer}             # timestamp of the last successful
                                             # mirror (empty if not mirrored yet)
  }
}
```

Example:

```
<repo name="SLES10-SP2-Updates" id="226" target="sles-10-i586" type="nu">
  <description>SLES10-SP2-Updates for sles-10-i586</description>
  <localpath>/local/htdocs/repo/$RCE/SLES10-SP2-Updates/sles-10-i586</
localpath>
  <mirrored date="1283523440"/>
  <url>https://nu.novell.com/repo/$RCE/SLES10-SP2-Updates/sles-10-i586/</url>
</repo>
```

`GET /repo/<repoid>/patches`

Returns a list of all patches in the specified software repository. The repoid can be obtained using the `/repos` or `/products/<productid>/repos/` call.

RNC:

```
start = element patches {
  element patch {
    attribute id {xsd:integer},                # SMT ID of the patch
    attribute name {xsd:string},               # patch's Unix name
    attribute version {xsd:integer}            # patch's version number
    attribute category {                       # patch importance category
      "security",
      "recommended",
      "optional",
      "mandatory"}
  }*
}
```

Example:

```
<patches>
  <patch name="slesp2-krb5" category="security" id="1471" version="6775"/>
  <patch name="slesp2-heartbeat" category="recommended" id="1524"
 version="5857"/>
  <patch name="slesp2-curl" category="security" id="1409" version="6402"/>
  ...
</patches>
```

`GET /repos`

Returns a list of all software repositories known to SMT. Those which are currently mirrored on SMT have non-empty mirror time stamp in the mirrored attribute.

RNC:

```
start = element repos {
  element repo {
    attribute id {xsd:integer},        # SMT ID of the repository
    attribute name {xsd:string},       # repository's Unix name
    attribute target {xsd:string},     # repository's target product
    attribute mirrored {xsd:integer}   # time stamp of the last successful
 mirror
                                       # (empty if not mirrored yet)
  }*
}
```

Example:

```
<repos>
  <repo name="SLE10-SDK-Updates" id="1" mirrored="" target="sles-10-x86_64"/>
  <repo name="SLE10-SDK-SP3-Pool" id="2" mirrored="" target="sles-10-ppc"/>
  <repo name="SLES10-SP2-Updates" id="226" mirrored="1283523440"
 target="sles-10-i586"/>
  ...
</repo>
```

`GET /patch/<patchid>`

Returns detailed information about the specified patch. The patchid can be obtained via the `/repo/<repoid>/patches` call.

RNC:

```
start = element patch {
  attribute id {xsd:integer},            # SMT ID of the patch
  attribute name {xsd:string},           # patch's Unix name
  attribute version {xsd:integer},       # patch's version number
  attribute category {                   # patch importance category
    "security",
    "recommended",
    "optional",
    "mandatory"},
  element title {xsd:string},            # title of the patch
  element description {text},            # description of issues fixed by the
 patch
  element issued {
    attribute date {xsd:integer}         # patch release time stamp
  },
  element packages {                     # packages which need update as part
                                         # of this patch
    element package {                    # individual RPM package data
      attribute name {xsd:string},       # package name
      attribute epoch {xsd:integer},     # epoch number
      attribute version {xsd:string},    # version string
      attribute release {xsd:string},    # release string
      attribute arch {xsd:string},       # architecture string
      element origlocation {xsd:anyURI}, # URL of the RPM package in the
                                         # original repository
      element smtlocation {xsd:anyURI}   # URL of the RPM package at the SMT
 server
    }*
  },
  element references {                   # references to issues fixed by this
                                         # patch
    element reference {                  # individual reference details
      attribute id,                      # ID number of the issue (bugzilla
                                         # or CVE number)
      attribute title {xsd:string},      # issue title
      attribute type {"bugzilla","cve"}, # type of the issue
      attribute href {xsd:anyURI}        # URL of the issue in its issue
                                         # tracking system
    }*
  }
}
```

Example:

```
<patch name="slesp2-krb5" category="security" id="1471" version="6775">
  <description>
    Specially crafted AES and RC4 packets could allow unauthenticated
    remote attackers to trigger an integer overflow leads to heap memory
    corruption (CVE-2009-4212). This has been fixed.
    Specially crafted AES and RC4 packets could allow
    unauthenticated remote attackers to trigger an integer
    overflow leads to heap memory corruption (CVE-2009-4212).
  </description>
  <issued date="1263343020"/>
  <packages>
    <package name="krb5" arch="i586" epoch="" release="19.43.2"
 version="1.4.3">
      <origlocation>https://nu.novell.com/repo/$RCE/SLES10-SP2-Updates/sles-10-
i586/rpm/i586/krb5-1.4.3-19.43.2.i586.rpm</origlocation>
      <smtlocation>http://kompost.suse.cz/repo/$RCE/SLES10-SP2-Updates/sles-10-
i586/rpm/i586/krb5-1.4.3-19.43.2.i586.rpm</smtlocation>
    </package>
    <package name="krb5-apps-servers" arch="i586" epoch="" release="19.43.2"
 version="1.4.3">
      <origlocation>https://nu.novell.com/repo/$RCE/SLES10-SP2-Updates/sles-10-
i586/rpm/i586/krb5-apps-servers-1.4.3-19.43.2.i586.rpm</origlocation>
      <smtlocation>http://kompost.suse.cz/repo/$RCE/SLES10-SP2-Updates/sles-10-
i586/rpm/i586/krb5-apps-servers-1.4.3-19.43.2.i586.rpm</smtlocation>
    </package>
    ...
  </packages>
  <references>
    <reference id="535943" href="https://bugzilla.suse.com/show_bug.cgi?
id=535943" title="bug number 535943" type="bugzilla"/>
    <reference id="CVE-2009-4212" href="http://cve.mitre.org/cgi-bin/
cvename.cgi?name=CVE-2009-4212" title="CVE-2009-4212" type="cve"/>
  </references>
  <title>Security update for Kerberos 5</title>
</patch>
```

`GET /products`

Returns list of all products known to SMT.

RNC:

```
start element products {
  element product {
    attribute id {xsd:integer},      # SMT ID of the product
    attribute name {xsd:string},     # Unix name of the product
    attribute version {xsd:string},  # version string
    attribute rel {xsd:string},      # release string
    attribute arch {xsd:string},     # target machine architecture string
    attribute uiname {xsd:string}    # name of the product to be
                                     # displayed to users
  }*
}
```

Example:

```
<products>
  <product name="SUSE_SLED" arch="x86_64" id="1824" rel="" uiname="SUSE Linux
 Enterprise Desktop 11 SP1" version="11.1"/>
  <product name="SUSE_SLES" arch="i686" id="1825" rel="" uiname="SUSE Linux
 Enterprise Server 11 SP1" version="11.1"/>
  <product name="sle-hae" arch="i686" id="1880" rel="" uiname="SUSE Linux
 Enterprise High Availability Extension 11 SP1" version="11.1"/>
  <product name="SUSE-Linux-Enterprise-Thin-Client" arch="" id="940" rel="SP1"
 uiname="SUSE Linux Enterprise 10 Thin Client SP1" version="10"/>
  ...
</products>
```

`GET /product/<productid>`

Returns information about the specified product. The productid can be obtained from data returned by the `/products` call.

RNC:

```
start = element product {
  attribute id {xsd:integer},       # SMT ID of the product
  attribute name {xsd:string},      # Unix name of the product
  attribute version {xsd:string},   # version string
  attribute rel {xsd:string},       # release string
  attribute arch {xsd:string},      # target machine architecture string
  attribute uiname {xsd:string}     # name of the product to be displayed
                                    # to users
}
```

Example:

```
<product name="SUSE_SLED" arch="x86_64" id="1824" rel="" uiname="SUSE Linux
 Enterprise Server 11 SP1" version="11.1"/>
```

`GET /product/<productid>/repos`

Returns the list of all software repositories for the specified product. The productid can be obtained from the data returned by the `/products` call.

RNC:

See the `/repos` call.

Example:

```
<repos>
  <repo name="SLED11-SP1-Updates" id="143" mirrored="" target="sle-11-x86_64"/>
  <repo name="SLE11-SP1-Debuginfo-Pool" id="400" mirrored="" target="sle-11-
x86_64"/>
  <repo name="SLED11-Extras" id="417" mirrored="" target="sle-11-x86_64"/>
  <repo name="SLED11-SP1-Pool" id="215" mirrored="" target="sle-11-x86_64"/>
  <repo name="nVidia-Driver-SLE11-SP1" id="469" mirrored="" target=""/>
  <repo name="ATI-Driver-SLE11-SP1" id="411" mirrored="" target=""/>
  <repo name="SLE11-SP1-Debuginfo-Updates" id="6" mirrored="" target="sle-11-
x86_64"/>
</repos>
```