SUSE Linux Enterprise Desktop 15 SP7

Administration Guide

Administration Guide

SUSE Linux Enterprise Desktop 15 SP7

This guide covers system administration tasks like maintaining, monitoring and customizing an initially installed system.

Publication Date: December 18, 2025

https://documentation.suse.com

Contents

Preface xx

1 Available documentation xx

2 Improving the documentation xxi

3 Documentation conventions xxii

4 Support xxiii

Support statement for SUSE Linux Enterprise Desktop xxiv • Technology previews xxv

I COMMON TASKS 1

1 Bash and Bash scripts 2

1.1 What is “the shell”? 2

Bash configuration files 2 • The directory structure 5

1.2 Writing shell scripts 9

1.3 Redirecting command events 11

1.4 Using aliases 12

1.5 Using variables in Bash 12

Using argument variables 13 • Using variable substitution 14

1.6 Grouping and combining commands 15

1.7 Working with common flow constructs 16

The if control command 16 • Creating loops with the **for** command 16

1.8 More information 17

2 sudo basics 18

2.1 Basic **sudo** usage 18

Running a single command 18 • Starting a shell 19

2.2 Configuring **sudo** 20

sudo configuration best practices 21 • Create a user-specific configuration file 21 • Create custom configurations by grouping items 23 • Simplify configurations by applying aliases 23 • Basic sudoers configuration syntax 25 • Basic sudoers rules 26

2.3 Using **sudo** with X.Org applications 27

2.4 More information 28

3 Using YaST 29

3.1 YaST interface overview 29

3.2 Useful key combinations 29

4 YaST in text mode 31

4.1 Navigation in modules 32

4.2 Advanced key combinations 33

4.3 Restriction of key combinations 34

4.4 YaST command line options 34

Installing packages from the command line 35 • Working with individual modules 35 • Command line parameters of YaST modules 35

5 Changing language and country settings with YaST 60

5.1 Changing the system language 60

Modifying system languages with YaST 61 • Switching the default system language 63 • Switching languages for standard X and GNOME applications 64

5.2 Changing the country and time settings 64

6 Managing users with YaST 67

6.1 User and group administration dialog 67

6.2 Managing user accounts 69

6.3 Additional options for user accounts 71
Automatic login and passwordless login 71 • Enforcing password policies 72 • Managing quotas 73

6.4 Changing default settings for local users 75

6.5 Assigning users to groups 76

6.6 Managing groups 76

6.7 Changing the user authentication method 78

6.8 Default system users 79

6.9 Default system groups 81

6.10 Reserved user and group IDs 82

7 YaST online update 84

7.1 The online update dialog 85

7.2 Installing patches 86

7.3 Viewing retracted patches 88

7.4 Automatic online update 88

8 Installing or removing software 91

8.1 Definition of terms 91

8.2 Registering an installed system 93
Registering with YaST 93 • Registering with SUSEConnect 93

8.3 Using the YaST software manager 93
Searching software 94 • Installing and removing packages or patterns 96 • Updating packages 97 • Package dependencies 99 • Handling package recommendations 100

8.4 Managing software repositories and services 101
Adding software repositories 101 • Managing repository properties 103 • Managing repository keys 104

8.5 The GNOME package updater 104

8.6 Updating packages with *GNOME Software* 107

9 Managing software with command line tools 109

9.1 Using Zypper 109

General usage 109 • Using Zypper subcommands 111 • Installing and removing software with Zypper 111 • Updating software with Zypper 116 • Identifying processes and services using deleted files 121 • Managing repositories with Zypper 123 • Querying repositories and packages with Zypper 125 • Showing lifecycle information 127 • Configuring Zypper 128 • Troubleshooting 128 • Zypper rollback feature on Btrfs file system 128 • More information 129

9.2 RPM—the package manager 129

Verifying package authenticity 130 • Managing packages: install, update, and uninstall 130 • Delta RPM packages 131 • RPM queries 132 • Installing and compiling source packages 135 • Compiling RPM packages with build 137 • Tools for RPM archives and the RPM database 137

10 System recovery and snapshot management with Snapper 138

10.1 Default setup 139

Default settings 140 • Types of snapshots 140 • Directories that are excluded from snapshots 141 • Customizing the setup 141

10.2 Using Snapper to undo changes 145

Undoing YaST and Zypper changes 146 • Using Snapper to restore files 151

10.3 System rollback by booting from snapshots 152

Snapshots after rollback 155 • Accessing and identifying snapshot boot entries 156 • Limitations 157

10.4 Enabling Snapper in user home directories 159

Installing pam_snapper and creating users 159 • Removing users 160 • Manually enabling snapshots in home directories 160

10.5 Creating and modifying Snapper configurations 160

Managing existing configurations 162

10.6 Manually creating and managing snapshots 165
Snapshot metadata 166 • Creating snapshots 167 • Modifying snapshot metadata 169 • Deleting snapshots 169

10.7 Automatic snapshot clean-up 170
Cleaning up numbered snapshots 171 • Cleaning up timeline snapshots 173 • Cleaning up snapshot pairs that do not differ 174 • Cleaning up manually created snapshots 175 • Adding disk quota support 175

10.8 Showing exclusive disk space used by snapshots 176

10.9 Frequently asked questions 178

11 Live kernel patching with KLP 180

11.1 Advantages of Kernel Live Patching 180

11.2 Kernel Live Patching overview 180
Kernel Live Patching scope 182 • Kernel Live Patching limitations 182

11.3 Activating Kernel Live Patching using YaST 182

11.4 Activating Kernel Live Patching from the command line 183

11.5 Performing Kernel Live Patching 184
Checking expiration date of the live patch 185

11.6 Troubleshooting Kernel Live Patching issues 185
Manual patch downgrade 185

12 User space live patching 187

12.1 About user space live patching 187
Prerequisites 187 • Supported libraries 188 • Using `libpulp` 188

12.2 ppc64le specific behavior of user space live patching 190

12.3 More information 191

13 Transactional updates 192

13.1 Limitations 192

13.2 Enabling `transactional-update` 194

13.3 Managing automatic updates 194

13.4 The **`transactional-update`** command 195

13.5 Troubleshooting 197

14 Remote graphical sessions with VNC 198

14.1 The **`vncviewer`** client 198

Connecting using the vncviewer CLI 198 • Connecting using the vncviewer GUI 199 • Notification of unencrypted connections 199

14.2 Remmina: the remote desktop client 199

Installation 199 • Main window 200 • Adding remote sessions 200 • Starting remote sessions 202 • Editing, copying, and deleting saved sessions 203 • Running remote sessions from the command line 203

14.3 Configuring one-time sessions on the VNC server 204

Available configurations 205 • Initiating a one-time VNC session 206 • Configuring one-time VNC sessions 206

14.4 Configuring persistent VNC server sessions 207

VNC session initiated using `vncmanager` 208

14.5 Configuring encryption on the VNC server 211

14.6 Compatibility with Wayland 212

15 File copying with RSync 213

15.1 Conceptual overview 213

15.2 Basic syntax 213

15.3 Copying files and directories locally 214

15.4 Copying files and directories remotely 215

15.5 Configuring and using an rsync server 215

15.6 More information 218

II BOOTING A LINUX SYSTEM 219

16 Introduction to the boot process 220

16.1 Terminology 220

16.2 The Linux boot process 221

The initialization and boot loader phase 221 • The kernel phase 223 • The init on initramfs phase 226 • The systemd phase 228

17 UEFI (Unified Extensible Firmware Interface) 229

17.1 Secure boot 229

Implementation on SUSE Linux Enterprise Desktop 230 • MOK (Machine Owner Key) 232 • Booting a custom kernel 233 • Using non-inbox drivers 235 • Features and limitations 236

17.2 The Secure Boot Revocation List 237

How to apply an online Revocation List update 237 • How to apply an online Revocation List update 238

17.3 More information 239

18 The boot loader GRUB 2 240

18.1 Main differences between GRUB legacy and GRUB 2 240

18.2 Configuration file structure 241

The file `/boot/grub2/grub.cfg` 242 • The file `/etc/default/grub` 242 • Scripts in `/etc/grub.d` 245 • Mapping between BIOS drives and Linux devices 247 • Editing menu entries during the boot procedure 247 • Setting a boot password 249 • Authorized access to boot menu entries 250

18.3 Configuring the boot loader with YaST 251

Boot loader location and boot code options 252 • Adjusting the disk order 254 • Configuring advanced options 254

18.4 Helpful GRUB 2 commands 257

18.5 Rescue mode 258

18.6 More information 259

19 The systemd daemon 260

19.1 The systemd concept 260

Unit file 261

19.2 Basic usage 262

Managing services in a running system 262 • Permanently enabling/disabling services 264

19.3 System start and target management 266

Targets compared to runlevels 266 • Debugging system start-up 269 • System V compatibility 273

19.4 Managing services with YaST 274

19.5 Customizing systemd 275

Where are unit files stored? 275 • Override with drop-in files 275 • Creating drop-in files manually 276 • Converting xinetd services to systemd 278 • Creating custom targets 279

19.6 Advanced usage 280

Cleaning temporary directories 280 • System log 281 • Snapshots 281 • Loading kernel modules 281 • Performing actions before loading a service 282 • Kernel control groups (cgroups) 283 • Terminating services (sending signals) 284 • Important notes on the D-Bus service 284 • Debugging services 285

19.7 systemd timer units 286

systemd timer types 286 • systemd timers and service units 287 • Practical example 287 • Managing systemd timers 289

19.8 More information 289

III SYSTEM 290

20 32-bit and 64-bit applications in a 64-bit system environment 291

20.1 Runtime support 291

20.2 Kernel specifications 292

21 **journalctl: query the systemd journal** 293
21.1 Making the journal persistent 293
21.2 **journalctl**: useful switches 294
21.3 Filtering the journal output 294
Filtering based on a boot number 295 • Filtering based on time interval 295 • Filtering based on fields 296
21.4 Investigating systemd errors 297
21.5 Journald configuration 298
Changing the journal size limit 298 • Forwarding the journal to /dev/ttyX 298 • Forwarding the journal to syslog facility 298
21.6 Using YaST to filter the systemd journal 299
21.7 Viewing logs in GNOME 300
22 **update-alternatives: managing multiple versions of commands and files** 301
22.1 Overview 301
22.2 Use cases 303
22.3 Getting an overview of alternatives 303
22.4 Viewing details on specific alternatives 303
22.5 Setting the default version of alternatives 304
22.6 Installing custom alternatives 305
22.7 Defining dependent alternatives 307
23 **Basic networking** 309
23.1 IP addresses and routing 312
IP addresses 312 • Netmasks and routing 313
23.2 IPv6—the next generation Internet 315
Advantages 315 • Address types and structure 317 • Coexistence of IPv4 and IPv6 321 • Configuring IPv6 322 • More information 323

23.3 Name resolution 323

23.4 Configuring a network connection with YaST 325

Configuring the network card with YaST 325

23.5 NetworkManager 336

NetworkManager and **wicked** 336 • NetworkManager functionality and configuration files 337 • Controlling and locking down NetworkManager features 338

23.6 Configuring a network connection manually 338

The **wicked** network configuration 338 • Configuration files 345 • Testing the configuration 356 • Unit files and start-up scripts 360

23.7 Setting up bonding devices 361

Hotplugging of bond ports 364 • Predictable naming scheme 365

23.8 Setting up team devices for Network Teaming 366

Use case: load balancing with Network Teaming 370 • Use case: failover with Network Teaming 371 • Use case: VLAN over team device 372

24 Printer operation 374

24.1 The CUPS workflow 375

24.2 Methods and protocols for connecting printers 376

24.3 Installing the software 376

24.4 Network printers 377

24.5 Configuring CUPS with command line tools 378

24.6 Printing from the command line 379

24.7 Special features in SUSE Linux Enterprise Desktop 380

CUPS and firewall 380 • Browsing for network printers 381 • PPD files in multiple packages 381

24.8 Troubleshooting 382

Printers without standard printer language support 382 • No suitable PPD file available for a PostScript printer 383 • Network printer connections 383 • Defective printouts without error message 385 • Disabled queues 386 • CUPS browsing: deleting print jobs 386 • Defective print jobs and data transfer errors 386 • Debugging CUPS 387 • More information 387

25 Graphical user interface 388

25.1 X Window System 388

25.2 Installing and configuring fonts 389
Showing installed fonts 390 • Viewing fonts 390 • Querying fonts 391 • Installing fonts 392 • Configuring the appearance of fonts 392

25.3 GNOME configuration for administrators 401
The `dconf` system 401 • System-wide configuration 401 • Managing tiling windows on Wayland 402 • More information 404

25.4 Switching between Intel and NVIDIA Optimus GPUs with SUSE Prime 404
Prerequisites 405 • Installing and using SUSE Prime 405 • Installing NVIDIA drivers 406

26 Accessing file systems with FUSE 407

26.1 Configuring FUSE 407

26.2 Mounting an NTFS partition 407

26.3 More information 408

27 Installing multiple kernel versions 409

27.1 Enabling and configuring multiversion support 409
Automatically deleting unused kernels 410 • Use case: deleting an old kernel after reboot only 411 • Use case: keeping older kernels as fallback 412 • Use case: keeping a specific kernel version 412

27.2 Installing/removing multiple kernel versions with YaST 412

27.3 Installing/removing multiple kernel versions with Zypper 413

28 Managing kernel modules 416

28.1 Listing loaded modules with lsmod and modinfo 416

28.2 Adding and removing kernel modules 417

Loading kernel modules automatically on boot 417 • Blacklisting kernel modules with modprobe 418

29 Dynamic kernel device management with udev 420

29.1 The /dev directory 420

29.2 Kernel uevents and udev 420

29.3 Drivers, kernel modules and devices 421

29.4 Booting and initial device setup 421

29.5 Monitoring the running udev daemon 422

29.6 Influencing kernel device event handling with udev rules 423

Using operators in udev rules 425 • Using substitutions in udev rules 426 • Using udev match keys 427 • Using udev assign keys 428

29.7 Persistent device naming 429

29.8 Files used by udev 430

29.9 More information 431

30 Special system features 432

30.1 Information about special software packages 432

The bash package and /etc/profile 432 • The cron package 433 • Stopping cron status messages 434 • Log files: package logrotate 434 • The **locate** command 434 • The **ulimit** command 435 • The **free** command 436 • Man pages and info pages 436 • Selecting man pages using the **man** command 436 • Settings for GNU Emacs 437

30.2 Virtual consoles 438

30.3 Keyboard mapping 438

30.4 Language and country-specific settings 439
System-wide locale settings 440 • Some examples 441 • Locale settings in `~/.i18n` 442 • Settings for language support 442 • More information 443

31 Using NetworkManager 444

31.1 Use cases for NetworkManager 444

31.2 Enabling or disabling NetworkManager 445

31.3 Configuring network connections 445
Managing wired network connections 447 • Managing wireless network connections 447 • Enabling wireless captive portal detection 448 • Configuring your Wi-Fi/Bluetooth card as an access point 448 • NetworkManager and VPN 449

31.4 NetworkManager and security 450
User and system connections 451 • Storing passwords and credentials 451 • Firewall zones 452

31.5 Frequently asked questions 452

31.6 Troubleshooting 454

31.7 More information 455

IV HARDWARE CONFIGURATION 456

32 Setting up your system keyboard layout 457

33 Setting up sound cards 458

34 Setting up a printer 461

34.1 Configuring printers 461
Adding drivers with YaST 462 • Editing a local printer configuration 464

34.2 Configuring printing via the network with YaST 464
Using CUPS 464 • Using print servers other than CUPS 466

34.3 Sharing printers over the network 466

35 Setting up a scanner 467
35.1 Configuring an HP all-in-one device 467
35.2 Sharing a scanner over the network 468
35.3 Scanning over the network 468
36 Power management 469
36.1 Power saving functions 469
36.2 Advanced configuration and power interface (ACPI) 470
Controlling the CPU performance 471 • Troubleshooting 471
36.3 Rest for the hard disk 473
36.4 Troubleshooting 474
CPU frequency does not work 474
37 Persistent memory 475
37.1 Introduction 475
37.2 Terms 477
37.3 Use cases 479
PMEM with DAX 479 • PMEM with BTT 480
37.4 Tools for managing persistent memory 480
37.5 Setting up persistent memory 481
Viewing available NVDIMM storage 481 • Configuring the storage as a single PMEM namespace with DAX 483 • Creating a PMEM namespace with BTT 485 • Placing the file system journal on PMEM/BTT 486
37.6 More information 487

V SERVICES 488

38 Service management with YaST 489

39 Time synchronization with NTP 491

39.1 Configuring an NTP client with YaST 492

NTP daemon start 492 • Type of the configuration source 493 • Configure time servers 493

39.2 Manually configuring NTP in the network 494

39.3 Configuring NTS 495

39.4 Configure `chronyd` at runtime using **`chronyc`** 498

39.5 Dynamic time synchronization at runtime 499

39.6 Setting up a local reference clock 499

39.7 Clock synchronization to an external time reference (ETR) 500

VI TROUBLESHOOTING 501

40 Help and documentation 502

40.1 Documentation directory 502

Release notes 503 • Package documentation 503

40.2 Man pages 504

40.3 Info pages 505

40.4 Online resources 506

41 Gathering system information for support 507

41.1 Displaying current system information 507

41.2 Collecting system information with supportconfig 508

Creating a service request number 508 • Upload targets 509 • Creating a supportconfig archive with YaST 509 • Creating a supportconfig archive from command line 512 • Understanding the output of

supportconfig 512 • Common supportconfig options 514 • Overview of the archive content 515

41.3 Submitting information to Global Technical Support 518

41.4 Analyzing system information 520

SCA command line tool 520 • SCA appliance 522 • Developing custom analysis patterns 533

41.5 Gathering information during the installation 534

41.6 Support of kernel modules 534

Technical background 535 • Working with unsupported modules 535

41.7 More information 536

42 Common problems and their solutions 537

42.1 Finding and gathering information 537

42.2 Boot problems 540

The GRUB 2 boot loader fails to load 540 • No login or prompt appears 541 • No graphical login 541 • Root Btrfs partition cannot be mounted 542 • Force checking root partitions 542 • Disable swap to enable booting 542 • GRUB 2 fails during reboot on a dual-boot system 542

42.3 Login problems 543

Valid user name and password combinations fail 543 • Valid user name and password not accepted 544 • Login to encrypted home partition fails 546 • GNOME desktop has issues 547

42.4 Network problems 547

NetworkManager problems 551

42.5 Data problems 552

Managing partition images 552 • Using the rescue system 553

A An example network 561

B GNU licenses 562

Preface

Revision History

2023-02-03

1 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 run into an issue, 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/.

On 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`**.

2 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.

3 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
- *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:

```
# command
> sudo command
```

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

```
> 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 end of the line:

```
> 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:

```
> command
```

```
output
```

- Notices

Warning: Warning notice

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

Important: Important notice

Important information you should know 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.

4 Support

Find the support statement for SUSE Linux Enterprise Desktop 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.

4.1 Support statement for SUSE Linux Enterprise Desktop

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 Desktop 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.
- 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.

4.2 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.

I Common tasks

1 Bash and Bash scripts 2

2 **sudo** basics 18

3 Using YaST 29

4 YaST in text mode 31

5 Changing language and country settings with YaST 60

6 Managing users with YaST 67

7 YaST online update 84

8 Installing or removing software 91

9 Managing software with command line tools 109

10 System recovery and snapshot management with Snapper 138

11 Live kernel patching with KLP 180

12 User space live patching 187

13 Transactional updates 192

14 Remote graphical sessions with VNC 198

15 File copying with RSync 213

1 Bash and Bash scripts

Revision History

2024-06-21

Today, many people use computers with a graphical user interface (GUI) like GNOME. Although GUIs offer many features, they are limited when performing automated task execution. Shells complement GUIs well, and this chapter gives an overview of several aspects of shells, in this case the Bash shell.

1.1 What is “the shell”?

Traditionally, *the* Linux shell is Bash (Bourne again Shell). When this chapter speaks about “the shell” it means Bash. There are more shells available (ash, csh, ksh, zsh, …), each employing different features and characteristics.

1.1.1 Bash configuration files

A shell can be invoked as an:

1. **Interactive login shell.** This is used when logging in to a machine, invoking Bash with the `--login` option or when logging in to a remote machine with SSH.
2. **Interactive non-login shell.** This is normally the case when starting xterm, konsole, gnome-terminal, or similar command line interface (CLI) tools.
3. **Non-interactive non-login shell.** This is invoked when invoking a shell script at the command line.

Each shell reads different configuration files. The following tables show the login and non-login shell configuration files.

> **Tip**
>
> Bash looks for its configuration files in a specific order depending on the type of shell where it is run. Find more details on the Bash man page (**`man 1 bash`**). Search for the headline `INVOCATION`.

TABLE 1.1: BASH CONFIGURATION FILES FOR LOGIN SHELLS

File	Description
`/etc/profile`	Do not modify this file, otherwise your modifications may be destroyed during your next update.
`/etc/profile.local`	Use this file if you extend `/etc/profile`
`/etc/profile.d/`	Contains system-wide configuration files for specific programs
`~/.profile`	Insert user specific configuration for login shells here

The login shell also sources the configuration files listed under *Table 1.2, "Bash configuration files for non-login shells"*.

TABLE 1.2: BASH CONFIGURATION FILES FOR NON-LOGIN SHELLS

`/etc/bash.bashrc`	Do not modify this file, otherwise your modifications may be destroyed during your next update.
`/etc/bash.bashrc.local`	Use this file to insert your system-wide modifications for Bash only
`~/.bashrc`	Insert user specific configuration here

Additionally, Bash uses multiple files:

TABLE 1.3: SPECIAL FILES FOR BASH

File	Description
`~/.bash_history`	Contains a list of all commands you have typed
`~/.bash_logout`	Executed when logging out

File	Description
`~/.alias`	User defined aliases of frequently used commands. See **`man 1 alias`** for more details about defining aliases.

No-Login Shells

There are special shells that block users from logging into the system: `/bin/false` and `/sbin/nologin`. Both fail silently when the user attempts to log into the system. This was intended as a security measure for system users, though modern Linux operating systems have more effective tools for controlling system access, such as PAM and AppArmor.

The default on SUSE Linux Enterprise Desktop is to assign `/bin/bash` to human users, and `/bin/false` or `/sbin/nologin` to system users. The `nobody` user has `/bin/bash` for historical reasons, as it is a user with minimum privileges that used to be the default for system users. However, whatever little bit of security gained by using `nobody` is lost when multiple system users use it. It should be possible to change it to `/sbin/nologin`; the fastest way to test it is change it and see if it breaks any services or applications.

Use the following command to list which shells are assigned to all users, system and human users, in `/etc/passwd`. The output varies according to the services and users on your system:

```
> sort -t: -k 7 /etc/passwd | awk -F: '{print $1"\t" $7}' | column -t
tux                  /bin/bash
nobody               /bin/bash
root                 /bin/bash
avahi                /bin/false
chrony               /bin/false
dhcpd                /bin/false
dnsmasq              /bin/false
ftpsecure            /bin/false
lightdm              /bin/false
mysql                /bin/false
postfix              /bin/false
rtkit                /bin/false
sshd                 /bin/false
tftp                 /bin/false
unbound              /bin/false
bin                  /sbin/nologin
daemon               /sbin/nologin
ftp                  /sbin/nologin
```

```
lp                 /sbin/nologin
mail               /sbin/nologin
man                /sbin/nologin
nscd               /sbin/nologin
polkitd            /sbin/nologin
pulse              /sbin/nologin
qemu               /sbin/nologin
radvd              /sbin/nologin
rpc                /sbin/nologin
statd              /sbin/nologin
svn                /sbin/nologin
systemd-coredump   /sbin/nologin
systemd-network    /sbin/nologin
systemd-timesync   /sbin/nologin
usbmux             /sbin/nologin
vnc                /sbin/nologin
wwwrun             /sbin/nologin
messagebus         /usr/bin/false
scard              /usr/sbin/nologin
```

1.1.2 The directory structure

The following table provides a short overview of the most important higher-level directories that you find on a Linux system. Find more detailed information about the directories and important subdirectories in the following list.

TABLE 1.4: OVERVIEW OF A STANDARD DIRECTORY TREE

Directory	Contents
`/`	Root directory—the starting point of the directory tree.
`/bin`	Essential binary files, such as commands that are needed by both the system administrator and normal users. Usually also contains the shells, such as Bash.
`/boot`	Static files of the boot loader.
`/dev`	Files needed to access host-specific devices.
`/etc`	Host-specific system configuration files.

Directory	Contents
/home	Holds the home directories of all users who have accounts on the system. However, root's home directory is not located in /home but in /root.
/lib	Essential shared libraries and kernel modules.
/media	Mount points for removable media.
/mnt	Mount point for temporarily mounting a file system.
/opt	Add-on application software packages.
/root	Home directory for the superuser root.
/sbin	Essential system binaries.
/srv	Data for services provided by the system.
/tmp	Temporary files.
/usr	Secondary hierarchy with read-only data.
/var	Variable data such as log files.
/windows	Only available if you have both Microsoft Windows* and Linux installed on your system. Contains the Windows data.

The following list provides more detailed information and gives some examples of which files and subdirectories can be found in the directories:

/bin

Contains the basic shell commands that may be used both by root and by other users. These commands include **ls**, **mkdir**, **cp**, **mv**, **rm** and **rmdir**. /bin also contains Bash, the default shell in SUSE Linux Enterprise Desktop.

/boot

Contains data required for booting, such as the boot loader, the kernel, and other data that is used before the kernel begins executing user-mode programs.

`/dev`

Holds device files that represent hardware components.

`/etc`

Contains local configuration files that control the operation of programs like the X Window System. The `/etc/init.d` subdirectory contains LSB init scripts that can be executed during the boot process.

`/home/USERNAME`

Holds the private data of every user who has an account on the system. The files located here can only be modified by their owner or by the system administrator. By default, your e-mail directory and personal desktop configuration are located here in the form of hidden files and directories, such as `.gconf/` and `.config`.

Note: Home directory in a network environment

If you are working in a network environment, your home directory may be mapped to a directory in the file system other than `/home`.

`/lib`

Contains the essential shared libraries needed to boot the system and to run the commands in the root file system. The Windows equivalent for shared libraries are DLL files.

`/media`

Contains mount points for removable media, such as CD-ROMs, flash disks, and digital cameras (if they use USB). `/media` generally holds any type of drive except the hard disk of your system. When your removable medium has been inserted or connected to the system and has been mounted, you can access it from here.

`/mnt`

This directory provides a mount point for a temporarily mounted file system. `root` may mount file systems here.

`/opt`

Reserved for the installation of third-party software. Optional software and larger add-on program packages can be found here.

`/root`

Home directory for the `root` user. The personal data of `root` is located here.

`/run`

A tmpfs directory used by `systemd` and various components. `/var/run` is a symbolic link to `/run`.

`/sbin`

As the `s` indicates, this directory holds utilities for the superuser. `/sbin` contains the binaries essential for booting, restoring and recovering the system in addition to the binaries in `/bin`.

`/srv`

Holds data for services provided by the system, such as FTP and HTTP.

`/tmp`

This directory is used by programs that require temporary storage of files.

Important: Cleaning up `/tmp` at boot time

Data stored in `/tmp` is not guaranteed to survive a system reboot. It depends, for example, on settings made in `/etc/tmpfiles.d/tmp.conf`.

`/usr`

`/usr` has nothing to do with users, but is the acronym for Unix system resources. The data in `/usr` is static, read-only data that can be shared among various hosts compliant with the `Filesystem Hierarchy Standard` (FHS). This directory contains all application programs including the graphical desktops such as GNOME and establishes a secondary hierarchy in the file system. `/usr` holds several subdirectories, such as `/usr/bin`, `/usr/sbin`, `/usr/local`, and `/usr/share/doc`.

`/usr/bin`

Contains generally accessible programs.

`/usr/sbin`

Contains programs reserved for the system administrator, such as repair functions.

`/usr/local`

In this directory the system administrator can install local, distribution-independent extensions.

`/usr/share/doc`

Holds various documentation files and the release notes for your system. In the `manual` subdirectory find an online version of this manual. If more than one language is installed, this directory may contain versions of the manuals for different languages.

Under `packages` find the documentation included in the software packages installed on your system. For every package, a subdirectory `/usr/share/doc/packages/PACKAGENAME` is created that often holds README files for the package and sometimes examples, configuration files or additional scripts.

If HOWTOs are installed on your system `/usr/share/doc` also holds the `howto` subdirectory in which to find additional documentation on many tasks related to the setup and operation of Linux software.

`/var`

Whereas `/usr` holds static, read-only data, `/var` is for data which is written during system operation and thus is variable data, such as log files or spooling data. For an overview of the most important log files you can find under `/var/log/`, refer to *Table 42.1, "Log files"*.

`/windows`

Only available if you have both Microsoft Windows and Linux installed on your system. Contains the Windows data available on the Windows partition of your system. Whether you can edit the data in this directory depends on the file system your Windows partition uses. If it is FAT32, you can open and edit the files in this directory. For NTFS, SUSE Linux Enterprise Desktop also includes write access support. However, the driver for the NTFS-3g file system has limited functionality.

1.2 Writing shell scripts

Shell scripts provide a convenient way to perform a wide range of tasks: collecting data, searching for a word or phrase in a text and other useful things. The following example shows a small shell script that prints a text:

EXAMPLE 1.1: A SHELL SCRIPT PRINTING A TEXT

```
#!/bin/sh ❶
# Output the following line: ❷
echo "Hello World" ❸
```

1. The first line begins with the *Shebang* characters (`#!`) which indicate that this file is a script. The interpreter, specified after the *Shebang*, executes the script. In this case, the specified interpreter is **`/bin/sh`**.
2. The second line is a comment beginning with the hash sign. We recommend that you comment difficult lines. With proper commenting, you can remember the purpose and function of the line. Also, other readers can better understand your script. Commenting is considered good practice in the development community.
3. The third line uses the built-in command **`echo`** to print the corresponding text.

Before you can run this script, there are a few prerequisites:

1. Every script should contain a Shebang line (as in the example above). If the line is missing, you need to call the interpreter manually.

2. You can save the script wherever you want. However, it is a good idea to save it in a directory where the shell can find it. The search path in a shell is determined by the environment variable `PATH`. A normal user does not have write access to `/usr/bin`. Therefore it is recommended to save your scripts in the users' directory `~/bin/`. The above example gets the name `hello.sh`.

3. The script needs executable permissions. Set the permissions with the following command:

```
> chmod +x ~/bin/hello.sh
```

If you have fulfilled all the above prerequisites, you can execute the script in the following ways:

1. **As absolute path.** The script can be executed with an absolute path. In our case, it is **`~/bin/hello.sh`**.

2. **Everywhere.** If the `PATH` environment variable contains the directory where the script is located, you can execute the script with **`hello.sh`**.

1.3 Redirecting command events

Each command can use three channels, either for input or output:

- **Standard output.** This is the default output channel. Whenever a command prints something, it uses the standard output channel.
- **Standard input.** If a command needs input from users or other commands, it uses this channel.
- **Standard error.** Commands use this channel for error reporting.

To redirect these channels, there are the following possibilities:

`Command > File`
: Saves the output of the command into a file, the existing file is deleted. For example, the **`ls`** command writes its output into the file `listing.txt`:

```
> ls > listing.txt
```

`Command >> File`
: Appends the output of the command to a file. For example, the **`ls`** command appends its output to the file `listing.txt`:

```
> ls >> listing.txt
```

`Command < File`
: Reads the file as input for the given command. For example, the **`read`** command reads in the content of the file into the variable:

```
> read a < foo
```

`Command1 | Command2`
: Redirects the output of the left command as input for the right command. For example, the **`cat`** command outputs the content of the `/proc/cpuinfo` file. This output is used by **`grep`** to filter only those lines which contain `cpu`:

```
> cat /proc/cpuinfo | grep cpu
```

Every channel has a *file descriptor*: 0 (zero) for standard input, 1 for standard output and 2 for standard error. It is allowed to insert this file descriptor before a < or > character. For example, the following line searches for a file starting with `foo`, but suppresses its errors by redirecting it to `/dev/null`:

```
> find / -name "foo*" 2>/dev/null
```

1.4 Using aliases

An alias is a shortcut definition of one or more commands. The syntax for an alias is:

```
alias NAME=DEFINITION
```

For example, the following line defines an alias **`lt`** that outputs a long listing (option `-l`), sorts it by modification time (`-t`), and prints it in reverse sorted order (`-r`):

```
> alias lt='ls -ltr'
```

To view all alias definitions, use **`alias`**. Remove your alias with **`unalias`** and the corresponding alias name.

1.5 Using variables in Bash

A shell variable can be global or local. Global variables, or environment variables, can be accessed in all shells. In contrast, local variables are visible in the current shell only.

To view all environment variables, use the **`printenv`** command. If you need to know the value of a variable, insert the name of your variable as an argument:

```
> printenv PATH
```

A variable, be it global or local, can also be viewed with **`echo`**:

```
> echo $PATH
```

To set a local variable, use a variable name followed by the equal sign, followed by the value:

```
> PROJECT="SLED"
```

Do not insert spaces around the equal sign, otherwise you get an error. To set an environment variable, use **`export`**:

```
> export NAME="tux"
```

To remove a variable, use **unset**:

```
> unset NAME
```

The following table contains common environment variables that you can use in you shell scripts:

TABLE 1.5: USEFUL ENVIRONMENT VARIABLES

`HOME`	the home directory of the current user
`HOST`	the current host name
`LANG`	when a tool is localized, it uses the language from this environment variable. English can also be set to `C`
`PATH`	the search path of the shell, a list of directories separated by colon
`PS1`	specifies the normal prompt printed before each command
`PS2`	specifies the secondary prompt printed when you execute a multi-line command
`PWD`	current working directory
`USER`	the current user

1.5.1 Using argument variables

For example, if you have the script **foo.sh** you can execute it like this:

```
> foo.sh "Tux Penguin" 2000
```

To access all the arguments which are passed to your script, you need positional parameters. These are `$1` for the first argument, `$2` for the second, and so on. You can have up to nine parameters. To get the script name, use `$0`.

The following script **foo.sh** prints all arguments from 1 to 4:

```
#!/bin/sh
```

```
echo \"$1\" \"$2\" \"$3\" \"$4\"
```

If you execute this script with the above arguments, you get:

```
"Tux Penguin" "2000" "" ""
```

1.5.2 Using variable substitution

Variable substitutions apply a pattern to the content of a variable either from the left or right side. The following list contains the possible syntax forms:

`${VAR#pattern}`

removes the shortest possible match from the left:

```
> file=/home/tux/book/book.tar.bz2
> echo ${file#*/}
home/tux/book/book.tar.bz2
```

`${VAR##pattern}`

removes the longest possible match from the left:

```
> file=/home/tux/book/book.tar.bz2
> echo ${file##*/}
book.tar.bz2
```

`${VAR%pattern}`

removes the shortest possible match from the right:

```
> file=/home/tux/book/book.tar.bz2
> echo ${file%.*}
/home/tux/book/book.tar
```

`${VAR%%pattern}`

removes the longest possible match from the right:

```
> file=/home/tux/book/book.tar.bz2
> echo ${file%%.*}
/home/tux/book/book
```

`${VAR/pattern_1/pattern_2}`

substitutes the content of *VAR* from the *PATTERN_1* with *PATTERN_2*:

```
> file=/home/tux/book/book.tar.bz2
```

```
> echo ${file/tux/wilber}
/home/wilber/book/book.tar.bz2
```

1.6 Grouping and combining commands

Shells allow you to concatenate and group commands for conditional execution. Each command returns an exit code which determines the success or failure of its operation. If it is 0 (zero) the command was successful, everything else marks an error which is specific to the command.

The following list shows, how commands can be grouped:

`Command1 ; Command2`

executes the commands in sequential order. The exit code is not checked. The following line displays the content of the file with **`cat`** and then prints its file properties with **`ls`** regardless of their exit codes:

```
> cat filelist.txt ; ls -l filelist.txt
```

`Command1 && Command2`

runs the right command, if the left command was successful (logical AND). The following line displays the content of the file and prints its file properties only, when the previous command was successful (compare it with the previous entry in this list):

```
> cat filelist.txt && ls -l filelist.txt
```

`Command1 || Command2`

runs the right command, when the left command has failed (logical OR). The following line creates only a directory in `/home/wilber/bar` when the creation of the directory in `/home/tux/foo` has failed:

```
> mkdir /home/tux/foo || mkdir /home/wilber/bar
```

`funcname(){ ... }`

creates a shell function. You can use the positional parameters to access its arguments. The following line defines the function `hello` to print a short message:

```
> hello() { echo "Hello $1"; }
```

You can call this function like this:

```
> hello Tux
```

which prints:

```
Hello Tux
```

1.7 Working with common flow constructs

To control the flow of your script, a shell has **`while`**, **`if`**, **`for`** and **`case`** constructs.

1.7.1 The if control command

The **`if`** command is used to check expressions. For example, the following code tests whether the current user is Tux:

```
if test $USER = "tux"; then
  echo "Hello Tux."
else
  echo "You are not Tux."
fi
```

The test expression can be as complex or simple as possible. The following expression checks if the file `foo.txt` exists:

```
if test -e /tmp/foo.txt ; then
  echo "Found foo.txt"
fi
```

The test expression can also be abbreviated in square brackets:

```
if [ -e /tmp/foo.txt ] ; then
  echo "Found foo.txt"
fi
```

Find more useful expressions at https://bash.cyberciti.biz/guide/If..else..fi ↗.

1.7.2 Creating loops with the `for` command

The **`for`** loop allows you to execute commands to a list of entries. For example, the following code prints certain information about PNG files in the current directory:

```
for i in *.png; do
```

```
 ls -l $i
done
```

1.8 More information

Important information about Bash is provided in the man pages **man bash**. More about this topic can be found in the following list:

- https://tldp.org/LDP/Bash-Beginners-Guide/html/index.html—Bash Guide for Beginners
- https://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html—BASH Programming - Introduction HOW-TO
- https://tldp.org/LDP/abs/html/index.html—Advanced Bash-Scripting Guide
- https://www.grymoire.com/Unix/Sh.html—Sh - the Bourne Shell

2 sudo basics

Revision History

2024-05-13

Running certain commands requires root privileges. However, for security reasons and to avoid mistakes, it is not recommended to log in as `root`. A safer approach is to log in as a regular user, and then use **`sudo`** to run commands with elevated privileges.

On SUSE Linux Enterprise Desktop, **`sudo`** is configured to work similarly to **`su`**. However, **`sudo`** provides a flexible mechanism that allows users to run commands with privileges of any other user. This can be used to assign roles with specific privileges to certain users and groups. For example, it is possible to allow members of the group `users` to run a command with the privileges of user `wilber`. Access to the command can be further restricted by disallowing any command options. While su always requires the `root` password for authentication with PAM, **`sudo`** can be configured to authenticate with your own credentials. This means that the users do not need to share the `root` password, which improves security.

2.1 Basic sudo usage

The following chapter provides an introduction to basic usage of **`sudo`**.

2.1.1 Running a single command

As a regular user, you can run any command as `root` by adding **`sudo`** before it. This prompts you to provide the root password. If authenticated successfully, this runs the command as `root`:

```
> id -un ❶
tux
> sudo id -un
root's password: ❷
root
> id -un
tux ❸
> sudo id -un
❹
root
```

❶ The **`id -un`** command prints the login name of the current user.

2. The password is not shown during input, neither as clear text nor as masking characters.
3. Only commands that start with **sudo** run with elevated privileges.
4. The elevated privileges persist for a certain period of time, so you do not need to provide the `root` password again.

Tip: I/O redirection

When using **sudo**, I/O redirection does not work:

```
> sudo echo s > /proc/sysrq-trigger
bash: /proc/sysrq-trigger: Permission denied
> sudo cat < /proc/1/maps
bash: /proc/1/maps: Permission denied
```

In the example above, only the **echo** and **cat** commands run with elevated privileges. The redirection is done by the user's shell with user privileges. To perform redirection with elevated privileges, either start a shell as in *Section 2.1.2, "Starting a shell"* or use the **dd** utility:

```
echo s | sudo dd of=/proc/sysrq-trigger
sudo dd if=/proc/1/maps | cat
```

2.1.2 Starting a shell

Using **sudo** every time to run a command with elevated privileges is not always practical. While you can use the **sudo bash** command, it is recommended to use one of the built-in mechanisms to start a shell:

`sudo -s (<command>)`

Starts a shell specified by the `SHELL` environment variable or the target user's default shell. If a command is specified, it is passed to the shell (with the `-c` option). Otherwise the shell runs in interactive mode.

```
tux:~ > sudo -s
root's password:
root:/home/tux # exit
tux:~ >
```

`sudo -i (<command>)`

Similar to `-s`, but starts the shell as a login shell. This means that the shell's start-up files (`.profile` etc.) are processed, and the current working directory is set to the target user's home directory.

```
tux:~ > sudo -i
root's password:
root:~ # exit
tux:~ >
```

Tip: Environment variables

By default, **`sudo`** does not propagate environment variables. This behavior can be changed using the `env_reset` option (see *Useful flags and options*).

2.2 Configuring **sudo**

`sudo` provides a wide range on configurable options.

Note: Locked yourself out of sudo

If you accidentally locked yourself out of **`sudo`**, use **`su -`** and the `root` password to start a root shell. To fix the error, run **`visudo`**.

Warning: Example configurations are for demonstration purposes only

The example rules outlined below are purely for demonstration purposes. Use them to understand the general syntax of **`sudo`** configuration files. Do not use them in real-world setups, because they do not reflect the complexity of these environments.

2.2.1 **sudo** configuration best practices

Before you start, here are a few ground rules for maintaining **sudo** configurations:

Always use `visudo` to edit `sudo` configuration files

Any changes to the **sudo** configuration should be done using the **visudo** command. **visudo** is a tailor-made tool that allows you to edit the **sudo** configuration files and runs basic syntax checks, making sure that the configuration remains intact and functional. A faulty **sudo** configuration can result in a user being locked out of their own system.

Always create custom configurations under `/etc/sudoers.d/`

Custom configurations must reside under `/etc/sudoers.d/` to be pulled in by **sudo**. Settings in the custom configuration files take precedence over the ones in the default configuration in `/etc/sudoers`.

Always mind the order in which configurations are read

To make sure the custom configurations are read in the correct order, prefix them with numbers. Use leading zeroes to establish the order in which the files are read. For example, `01_myfirstconfig` is parsed before `10_myotherconfig`. If a directive has been set in a file that is read before another file that contains conflicting information, the last-read directive is applied.

Always use descriptive file names

Use file names that hint at what the configuration file does. This helps you keep track of what your **sudo** setup is supposed to do.

2.2.2 Create a user-specific configuration file

Create a **sudo** configuration file that allows a normal user (`tux`) to use the **useradd** command with their own password instead of the `root` password.

EXAMPLE 2.1: CREATE A USER-SPECIFIC CONFIGURATION FILE

1. As system administrator (`root`), create a custom configuration file that holds the new user-specific directives by starting **visudo**. Use both numbering and a descriptive name:

```
# visudo -f /etc/sudoers.d/02_usermanagement
```

2. Create a rule that allows `tux` to execute the **`/usr/sbin/useradd`** binary in the entire environment that this **`sudo`** configuration is applied to:

```
tux❶ ALL❷ = /usr/sbin/useradd❸
```

❶ Specify the user or group. List users by name or `#UID`, and groups by `%GROUPNAME`. Separate multiple items with commas. To negate entries, use `!`.

❷ Specify one or several (separated by commas) hosts. Use (fully qualified) host names or IP addresses. Add `ALL` to enforce this setting globally across all hosts. Use `!` for negations.

❸ Specify one or several executables (separated by commas). When specifying them, make sure to mind the following rules:

`/usr/sbin/useradd`
: Without any additional options added, this allows the execution of every possible **`useradd`** command.

`/usr/sbin/useradd -c`
: If you explicitly specify an option, then that option is the only one that is allowed. Nothing else would be available to the user you specified above.

`/usr/sbin/useradd ""`
: This would just let the user invoke a mere **`useradd`** without any option at all.

In the example above, you would want to either allow all options and subcommands or limit them to a few for security reasons, but forbidding a user from specifying any option at all would be pointless in this context.

3. To let the user use their own password instead of the `root` password, add the following line:

```
Defaults:tux !targetpw
```

When active, this flag requires the user to enter the password of the target user, that is, `root`. This flag is enabled by default on any SUSE Linux Enterprise Desktop system. Negate it using `!` to require the user to just enter their own password instead of the `root` password.

4. Save the configuration, leave the editor and open a second shell to test whether **`sudo`** honors your new configuration.

2.2.3 Create custom configurations by grouping items

Modify the configuration from *Example 2.1, "Create a user-specific configuration file"* so that a group of named users can run the **useradd** command without the need for the `root` password. Also, add the **usermod** and **userdel** to the list of commands available to this group.

EXAMPLE 2.2: CREATE CUSTOM CONFIGURATIONS BY GROUPING ITEMS

1. To modify the example configuration, open it as system administrator with **visudo**:

```
# visudo /etc/sudoers.d/02_usermanagement
```

2. Add more users to the rule in a comma-separated list:

```
tux, wilber ALL = /usr/sbin/useradd
```

3. To allow the listed users to execute a list of commands, specify the commands as a comma-separated list:

```
tux, wilber ALL = /usr/sbin/useradd, /usr/sbin/usermod, /usr/sbin/userdel
```

4. To let the listed users use their own password instead of the `root` password, add the following line:

```
Defaults:tux, wilber !targetpw
```

When active, this flag requires the listed users to enter the password of the target user, that is, `root`. This flag is enabled by default on any SUSE Linux Enterprise Desktop system. Negate it using `!` to require the listed users to just enter their own password instead of the `root` password.

5. Save the configuration, leave the editor and open a second shell to test whether **sudo** honors your new configuration.

2.2.4 Simplify configurations by applying aliases

Use aliases to simplify your custom configuration from *Example 2.2, "Create custom configurations by grouping items"* even further. Grouping items helps to a certain extent, but using global aliases for users, commands and hosts is the most efficient way to keep a clean and lean **sudo** configuration.

Using aliases and groups instead of lists is a much better way to address changes in your setup. Should a user leave, just remove them from the global `User_Alias` declaration in your alias declaration file instead of scouring all the separate custom configuration files. The same procedure applies for any other type of alias (`Host_Alias`, `Cmnd_Alias` and `Runas_Alias`).

EXAMPLE 2.3: SIMPLIFY CONFIGURATIONS BY APPLYING ALIASES

1. Create a new file to hold your global alias definitions:

```
# visudo /etc/sudoers.d/01_aliases
```

2. Add the following line to create the `TEAMLEADERS` alias:

```
User_Alias    TEAMLEADERS = tux, wilber
```

3. Add the following line to create the `USERMANAGEMENT` alias:

```
Cmnd_Alias    USERMANAGEMENT = /usr/sbin/useradd, /usr/sbin/usermod, /usr/sbin/
userdel
```

4. Save your changes and exit **visudo**.

5. As system administrator, start **visudo** to edit the example configuration file:

```
# visudo -f /etc/sudoers.d/02_usermanagement
```

6. Delete the previous rule and replace it with the following rule that uses the aliases you have just defined above:

```
TEAMLEADERS ALL = USERMANAGEMENT
```

7. To let all the users defined by `User_Alias` use their own password instead of the `root` password, add the following line:

```
Defaults:TEAMLEADERS !targetpw
```

8. Save the configuration, leave the editor and open a second shell to test whether **sudo** honors your new configuration.

2.2.5 Basic sudoers configuration syntax

The sudoers configuration files contain two types of options: strings and flags. While strings can contain any value, flags can be turned either ON or OFF. The most important syntax constructs for sudoers configuration files are as follows:

```
# Everything on a line after # is ignored ❶
Defaults !insults # Disable the insults flag ❷
Defaults env_keep += "DISPLAY HOME" # Add DISPLAY and HOME to env_keep
tux ALL = NOPASSWD: /usr/bin/frobnicate, PASSWD: /usr/bin/journalctl ❸
```

❶ There are two exceptions: `#include` and `#includedir` are regular commands.

❷ Remove the `!` character to set the desired flag to ON.

❸ See *Section 2.2.6, "Basic sudoers rules"*.

USEFUL FLAGS AND OPTIONS

`targetpw`

This flag controls whether the invoking user is required to enter the password of the target user (ON) (for example `root`) or the invoking user (OFF).

```
Defaults targetpw # Turn targetpw flag ON
```

`rootpw`

If set, **sudo** prompts for the `root` password. The default is OFF.

```
Defaults !rootpw # Turn rootpw flag OFF
```

`env_reset`

If set, **sudo** constructs a minimal environment with `TERM`, `PATH`, `HOME`, `MAIL`, `SHELL`, `LOGNAME`, `USER`, `USERNAME`, and `SUDO_*`. Additionally, variables listed in `env_keep` are imported from the calling environment. The default is ON.

```
Defaults env_reset # Turn env_reset flag ON
```

`env_keep`

List of environment variables to keep when the `env_reset` flag is ON.

```
# Set env_keep to contain EDITOR and PROMPT
Defaults env_keep = "EDITOR PROMPT"
Defaults env_keep += "JRE_HOME" # Add JRE_HOME
Defaults env_keep -= "JRE_HOME" # Remove JRE_HOME
```

`env_delete`

List of environment variables to remove when the `env_reset` flag is OFF.

```
# Set env_delete to contain EDITOR and PROMPT
Defaults env_delete = "EDITOR PROMPT"
Defaults env_delete += "JRE_HOME" # Add JRE_HOME
Defaults env_delete -= "JRE_HOME" # Remove JRE_HOME
```

The `Defaults` token can also be used to create aliases for a collection of users, hosts, and commands. Furthermore, it is possible to apply an option only to a specific set of users.

For detailed information about the sudoers configuration files, consult **`man 5 sudoers`**.

2.2.6 Basic sudoers rules

Each rule follows the following scheme (`[]` marks optional parts):

```
#Who      Where         As whom      Tag                  What
User_List Host_List = [(User_List)] [NOPASSWD:|PASSWD:] Cmnd_List
```

SUDOERS RULE SYNTAX

`User_List`

One or several (separated by comma) identifiers: either a user name, a group in the format `%GROUPNAME`, or a user ID in the format `#UID`. Negation can be specified with the `!` prefix.

`Host_List`

One or several (separated by comma) identifiers: either a (fully qualified) host name or an IP address. Negation can be specified with the `!` prefix. `ALL` is a common choice for `Host_List`.

`NOPASSWD:|PASSWD:`

The user is not prompted for a password when running commands matching `Cmd_List` after `NOPASSWD:`.

`PASSWD` is the default. It only needs to be specified when both `PASSWD` and `NOPASSWD` are on the same line:

```
tux ALL = PASSWD: /usr/bin/foo, NOPASSWD: /usr/bin/bar
```

`Cmnd_List`

One or several (separated by comma) specifiers: a path to an executable, followed by an optional allowed argument.

```
/usr/bin/foo     # Anything allowed
/usr/bin/foo bar # Only "/usr/bin/foo bar" allowed
/usr/bin/foo ""  # No arguments allowed
```

`ALL` can be used as `User_List`, `Host_List`, and `Cmnd_List`.

A rule that allows `tux` to run all commands as root without entering a password:

```
tux ALL = NOPASSWD: ALL
```

A rule that allows `tux` to run **`systemctl restart apache2`**:

```
tux ALL = /usr/bin/systemctl restart apache2
```

A rule that allows `tux` to run **`wall`** as `admin` with no arguments:

```
tux ALL = (admin) /usr/bin/wall ""
```

Warning: Unsafe rules

Do not use rules like `ALL ALL = ALL` without `Defaults targetpw`. Otherwise anyone can run commands as `root`.

Important: Winbind and sudo

When specifying the group name in the `sudoers` file, make sure that you use the NetBIOS domain name instead of the realm, for example:

```
%DOMAIN\\GROUP_NAME ALL = (ALL) ALL
```

Keep in mind that when using winbindd, the format also depends on the `winbind separator` option in the `smb.conf` file. By default, it is `\`. If it is changed, for example, to `+`, then the account format in the `sudoers` file must be `DOMAIN+GROUP_NAME`.

2.3 Using **sudo** with X.Org applications

Starting graphical applications with **`sudo`** normally results in the following error:

```
> sudo xterm
xterm: Xt error: Can't open display: %s
```

```
xterm: DISPLAY is not set
```

A simple workaround is to use xhost to temporarily allow the root user to access the local user's X session. This is done using the following command:

```
xhost si:localuser:root
```

The command below removes the granted access:

```
xhost -si:localuser:root
```

Warning: Potential security issue

Running graphical applications with root privileges has security implications. It is recommended to enable root access for a graphical application only as an exception. It is also recommended to revoke the granted root access as soon as the graphical application is closed.

2.4 More information

The **`sudo --help`** command offers a brief overview of the available command line options, while the **`man sudoers`** command provides detailed information about `sudoers` and its configuration.

3 Using YaST

Revision History

2023-08-21

YaST is a SUSE Linux Enterprise Desktop tool that provides a graphical interface for all essential installation and system configuration tasks. Whether you need to update packages, configure a printer, modify firewall settings, set up an FTP server, or partition a hard disk—you can do it using YaST. Written in Ruby, YaST features an extensible architecture that makes it possible to add new functionality via modules.

Additional information about YaST is available on the project's official Web site at https://yast.opensuse.org/.

3.1 YaST interface overview

YaST has two graphical interfaces: one for use with graphical desktop environments like KDE and GNOME, and an ncurses-based pseudo-graphical interface for use on systems without an X server (see *Chapter 4, YaST in text mode*).

In the graphical version of YaST, all modules in YaST are grouped by category, and the navigation sidebar allows you to quickly access modules in the desired category. The search field at the top makes it possible to find modules by their names. To find a specific module, enter its name into the search field, and you should see the modules that match the entered string as you type.

Important: List of installed YaST modules

The list of installed modules for the ncurses-based and GUI version of YaST may differ. Before starting any YaST module, verify that it is installed for the version of YaST that you are using.

3.2 Useful key combinations

The graphical version of YaST supports keyboard shortcuts

`Print Screen`

Take and save a screenshot. It may not work on certain desktop environments.

`Shift`–`F4`

Enable and disable the color palette optimized for visually impaired users.

`Shift`–`F7`

Enable/disable logging of debug messages.

`Shift`–`F8`

Open a file dialog to save log files to a user-defined location.

`Ctrl`–`Shift`–`Alt`–`D`

Send a DebugEvent. YaST modules can react to this by executing special debugging actions. The result depends on the specific YaST module.

`Ctrl`–`Shift`–`Alt`–`M`

Start and stop macro recorder.

`Ctrl`–`Shift`–`Alt`–`P`

Replay macro.

`Ctrl`–`Shift`–`Alt`–`S`

Show stylesheet editor.

`Ctrl`–`Shift`–`Alt`–`T`

Dump widget tree to the log file.

`Ctrl`–`Shift`–`Alt`–`X`

Open a terminal window (xterm). Useful for installation process via VNC.

`Ctrl`–`Shift`–`Alt`–`Y`

Show widget tree browser.

4 YaST in text mode

Revision History

2024-05-13

The ncurses-based pseudo-graphical YaST interface is designed primarily to help system administrators to manage systems without an X server. The interface offers several advantages compared to the conventional GUI. You can navigate the ncurses interface using the keyboard, and there are keyboard shortcuts for practically all interface elements. The ncurses interface is light on resources, and runs fast even on modest hardware. You can run the ncurses-based version of YaST via an SSH connection, so you can administer remote systems. Keep in mind that the minimum supported size of the terminal emulator in which to run YaST is 80x25 characters.

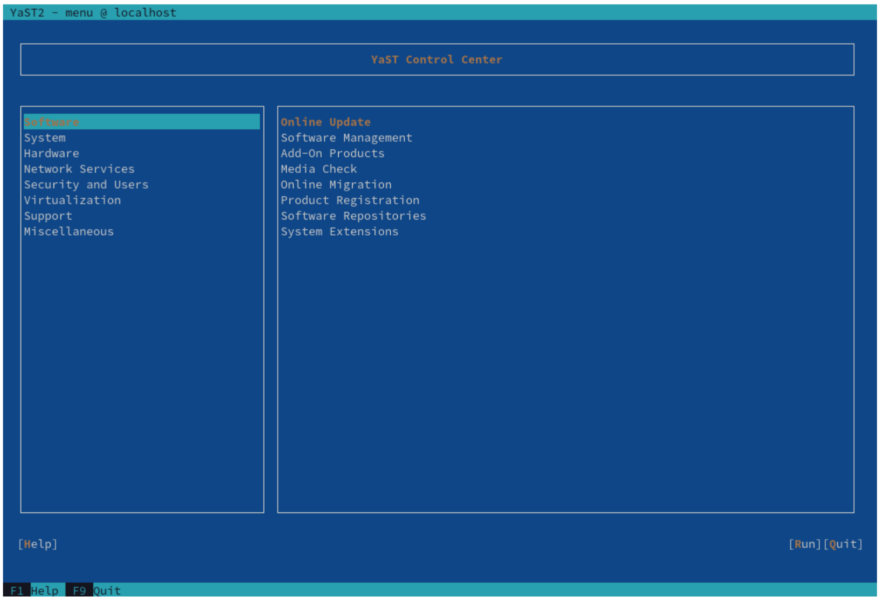


FIGURE 4.1: MAIN WINDOW OF YAST IN TEXT MODE

To launch the ncurses-based version of YaST, open the terminal and run the **`sudo yast2`** command. Use the `→|` or arrow keys to navigate between interface elements like menu items, fields and buttons. All menu items and buttons in YaST can be accessed using the appropriate function keys or keyboard shortcuts. For example, you can cancel the current operation by pressing `F9`, while the `F10` key can be used to accept the changes. Each menu item and button in YaST's ncurses-based interface has a highlighted letter in its label. This letter is part of the keyboard shortcut assigned to the interface element. For example, the letter `Q` is highlighted in the *Quit* button. This means that you can activate the button by pressing `Alt`–`Alt+Q`.

Tip: Refreshing YaST dialogs

If a YaST dialog gets corrupted or distorted, for example, while resizing the window, press `Ctrl`–`L` to refresh and restore its contents.

4.1 Navigation in modules

The following description of the control elements in the YaST modules assumes that all function keys and `Alt` key combinations work and are not assigned to different global functions. Read *Section 4.3, "Restriction of key combinations"* for information about possible exceptions.

Moving between buttons and selection lists

Use `→|` to move between the buttons and frames containing selection lists. To navigate in the opposite direction, use `Alt`–`→|` or `Shift`–`→|` combinations.

Navigating in selection lists

Use the arrow keys (`↑` and `↓`) to move through the individual elements in an active frame containing a selection list. If individual entries are longer than the frame's width, use `Shift`–`→` or `Shift`–`←` to scroll horizontally. If the arrow key causes the selection to move to another frame, use `Ctrl`–`E` or `Ctrl`–`A` instead.

Working with buttons, radio buttons, and check boxes

To select items with empty square brackets (check boxes) or empty parentheses (radio buttons), press `Space` or `Enter`. Alternatively, radio buttons and check boxes can be selected directly with `Alt`–`highlighted_letter`. In this case, you do not need to confirm with `Enter`. If you navigate to an item with `→|`, press `Enter` to execute the selected action or activate the respective menu item.

Function keys

The function keys (from `F1` to `F12`) enable quick access to the specific buttons. Available function key combinations (FX) are shown in the bottom line of the YaST screen. Which function keys are really mapped to which buttons depend on the active YaST module, because the different modules offer different buttons (*Details*, *Info*, *Add*, *Delete*, etc.). Use `F10` for *Accept*, *OK*, *Next*, and *Finish*. Press `F1` to access the YaST help.

Using the navigation tree

Certain YaST modules use a navigation tree in the left part of the window to select configuration dialogs. Use the arrow keys (`↑` and `↓`) to navigate in the tree. Use `Space` to open or close tree items. In the ncurses mode, `Enter` must be pressed after a selection in the navigation tree to show the selected dialog. This is an intentional behavior to save time-consuming redraws when browsing through the navigation tree.

Selecting software in the software installation module

Use the filters on the left side to list packages matching the specified string. Installed packages are marked with the letter `i`. To change the status of a package, press `Space` or `Enter` . Alternatively, use the *Actions* menu to select the needed status change (install, delete, update, taboo or lock).

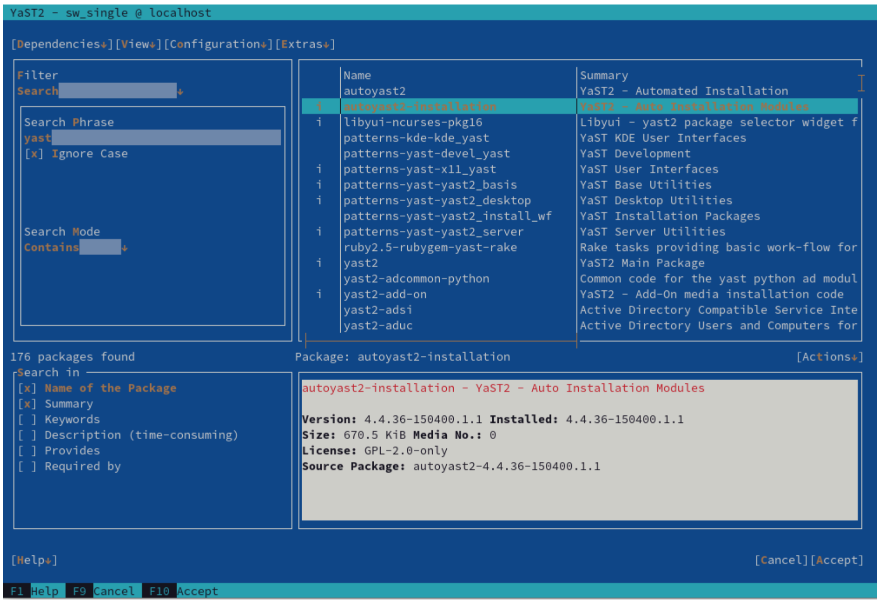


FIGURE 4.2: THE SOFTWARE INSTALLATION MODULE

4.2 Advanced key combinations

The ncurses-based version of YaST offers several advanced key combinations.

`Shift` – `F1`

List advanced hotkeys.

`Shift` – `F4`

Change color schema.

`Ctrl` – `Q`

: Quit the application.

`Ctrl` – `L`

: Refresh screen.

`Ctrl` – `D` `F1`

: List advanced hotkeys.

`Ctrl` – `D` `Shift` – `D`

: Dump dialog to the log file as a screenshot.

`Ctrl` – `D` `Shift` – `Y`

: Open YDialogSpy to see the widget hierarchy.

4.3 Restriction of key combinations

If your window manager uses global `Alt` combinations, the `Alt` combinations in YaST may not work. Keys like `Alt` or `Shift` can also be occupied by the settings of the terminal.

Using `Alt` instead of `Esc`

`Alt` shortcuts can be executed with `Esc` instead of `Alt`. For example, `Esc` – `H` replaces `Alt` – `H`. (Press `Esc`, *then* press `H`.)

Backward and forward navigation with `Ctrl` – `F` and `Ctrl` – `B`

If the `Alt` and `Shift` combinations are taken over by the window manager or the terminal, use the combinations `Ctrl` – `F` (forward) and `Ctrl` – `B` (backward) instead.

Restriction of function keys

The function keys (`F1` ... `F12`) are also used for functions. Certain function keys may be taken over by the terminal and may not be available for YaST. However, the `Alt` key combinations and function keys should always be fully available on a text-only console.

4.4 YaST command line options

Besides the text mode interface, YaST provides a command line interface. To get a list of YaST command line options, use the following command:

```
> sudo yast -h
```

4.4.1 Installing packages from the command line

If you know the package name, and the package is provided by an active installation repository, you can use the command line option `-i` to install the package:

```
> sudo yast -i package_name
```

or

```
> sudo yast --install -i package_name
```

package_name can be a single short package name, for example, `gvim`, installed with dependency checking, or the full path to an RPM package, which is installed without dependency checking.

While YaST offers basic functionality for managing software from the command line, consider using Zypper for more advanced package management tasks. Find more information on using Zypper in *Section 9.1, "Using Zypper"*.

4.4.2 Working with individual modules

To save time, you can start individual YaST modules using the following command:

```
> sudo yast module_name
```

View a list of all modules available on your system with **`yast -l`** or **`yast --list`**.

4.4.3 Command line parameters of YaST modules

To use YaST functionality in scripts, YaST provides command line support for individual modules. However, not all modules have command line support. To display the available options of a module, use the following command:

```
> sudo yast module_name help
```

If a module does not provide command line support, it is started in a text mode with the following message:

```
This YaST module does not support the command line interface.
```

The following sections describe all YaST modules with command line support, along with a brief explanation of all their commands and available options.

4.4.3.1 Common YaST module commands

All YaST modules support the following commands:

help

Lists all the module's supported commands with their description:

```
> sudo yast lan help
```

longhelp

Same as `help`, but adds a detailed list of all command's options and their descriptions:

```
> sudo yast lan longhelp
```

xmlhelp

Same as `longhelp`, but the output is structured as an XML document and redirected to a file:

```
> sudo yast lan xmlhelp xmlfile=/tmp/yast_lan.xml
```

interactive

Enters the *interactive* mode. This lets you run the module's commands without prefixing them with `sudo yast`. Use `exit` to leave the interactive mode.

4.4.3.2 yast add-on

Adds a new add-on product from the specified path:

```
> sudo yast add-on http://server.name/directory/Lang-AddOn-CD1/
```

You can use the following protocols to specify the source path: http:// ftp:// nfs:// disk:// cd:// or dvd://.

4.4.3.3 yast audit-laf

Displays and configures the Linux Audit Framework. Refer to the *Book "Security and Hardening Guide"* for more details. `yast audit-laf` accepts the following commands:

set

Sets an option:

```
> sudo yast audit-laf set log_file=/tmp/audit.log
```

For a complete list of options, run **yast audit-laf set help**.

show

Displays settings of an option:

```
> sudo yast audit-laf show diskspace
space_left: 75
space_left_action: SYSLOG
admin_space_left: 50
admin_space_left_action: SUSPEND
action_mail_acct: root
disk_full_action: SUSPEND
disk_error_action: SUSPEND
```

For a complete list of options, run **yast audit-laf show help**.

4.4.3.4 yast dhcp-server

Manages the DHCP server and configures its settings. **yast dhcp-server** accepts the following commands:

disable

Disables the DHCP server service.

enable

Enables the DHCP server service.

host

Configures settings for individual hosts.

interface

Specifies to which network interface to listen to:

```
> sudo yast dhcp-server interface current
Selected Interfaces: eth0
Other Interfaces: bond0, pbu, eth1
```

For a complete list of options, run **yast dhcp-server interface help**.

options

Manages global DHCP options. For a complete list of options, run **yast dhcp-server options help**.

status

Prints the status of the DHCP service.

subnet

Manages the DHCP subnet options. For a complete list of options, run **yast dhcp-server subnet help**.

4.4.3.5 yast dns-server

Manages the DNS server configuration. **yast dns-server** accepts the following commands:

acls

Displays access control list settings:

```
> sudo yast dns-server acls show
ACLs:
-----
 Name       Type        Value
 ----------------------------
 any        Predefined
 localips   Predefined
 localnets  Predefined
 none       Predefined
```

dnsrecord

Configures zone resource records:

```
> sudo yast dnsrecord add zone=example.org query=office.example.org type=NS
 value=ns3
```

For a complete list of options, run **yast dns-server dnsrecord help**.

forwarders

Configures DNS forwarders:

```
> sudo yast dns-server forwarders add ip=10.0.0.100
> sudo yast dns-server forwarders show
[...]
Forwarder IP
------------
10.0.0.100
```

For a complete list of options, run **yast dns-server forwarders help**.

host

Handles “A” and its related “PTR” record at once:

```
> sudo yast dns-server host show zone=example.org
```

For a complete list of options, run **`yast dns-server host help`**.

logging

Configures logging settings:

```
> sudo yast dns-server logging set updates=no transfers=yes
```

For a complete list of options, run **`yast dns-server logging help`**.

mailserver

Configures zone mail servers:

```
> sudo yast dns-server mailserver add zone=example.org mx=mx1 priority=100
```

For a complete list of options, run **`yast dns-server mailserver help`**.

nameserver

Configures zone name servers:

```
> sudo yast dns-server nameserver add zone=example.com ns=ns1
```

For a complete list of options, run **`yast dns-server nameserver help`**.

soa

Configures the start of authority (SOA) record:

```
> sudo yast dns-server soa set zone=example.org serial=2006081623 ttl=2D3H20S
```

For a complete list of options, run **`yast dns-server soa help`**.

startup

Manages the DNS server service:

```
> sudo yast dns-server startup atboot
```

For a complete list of options, run **`yast dns-server startup help`**.

transport

Configures zone transport rules. For a complete list of options, run **`yast dns-server transport help`**.

zones

Manages DNS zones:

```
> sudo yast dns-server zones add name=example.org zonetype=master
```

For a complete list of options, run **`yast dns-server zones help`**.

4.4.3.6 yast disk

Prints information about all disks or partitions. The only supported command is **`list`** followed by either of the following options:

disks

Lists all configured disks in the system:

```
> sudo yast disk list disks
Device   | Size       | FS Type | Mount Point | Label | Model
---------+------------+---------+-------------+-------+-------------
/dev/sda | 119.24 GiB |         |             |       | SSD 840
/dev/sdb |  60.84 GiB |         |             |       | WD1003FBYX-0
```

partitions

Lists all partitions in the system:

```
> sudo yast disk list partitions
Device          | Size       | FS Type | Mount Point | Label | Model
---------------+------------+---------+-------------+-------+------
/dev/sda1       |   1.00 GiB | Ext2    | /boot       |       |
/dev/sdb1       |   1.00 GiB | Swap    | swap        |       |
/dev/sdc1       | 698.64 GiB | XFS     | /mnt/extra  |       |
/dev/vg00/home | 580.50 GiB | Ext3    | /home       |       |
/dev/vg00/root | 100.00 GiB | Ext3    | /           |       |
[...]
```

4.4.3.7 yast ftp-server

Configures FTP server settings. **`yast ftp-server`** accepts the following options:

SSL, TLS

Controls secure connections via SSL and TLS. SSL options are valid for the `vsftpd` only.

```
> sudo yast ftp-server SSL enable
> sudo yast ftp-server TLS disable
```

access

Configures access permissions:

```
> sudo yast ftp-server access authen_only
```

For a complete list of options, run **yast ftp-server access help**.

anon_access

Configures access permissions for anonymous users:

```
> sudo yast ftp-server anon_access can_upload
```

For a complete list of options, run **yast ftp-server anon_access help**.

anon_dir

Specifies the directory for anonymous users. The directory must already exist on the server:

```
> sudo yast ftp-server anon_dir set_anon_dir=/srv/ftp
```

For a complete list of options, run **yast ftp-server anon_dir help**.

chroot

Controls *change root* environment (chroot):

```
> sudo yast ftp-server chroot enable
> sudo yast ftp-server chroot disable
```

idle-time

Sets the maximum idle time in minutes before FTP server terminates the current connection:

```
> sudo yast ftp-server idle-time set_idle_time=15
```

logging

Determines whether to save the log messages into a log file:

```
> sudo yast ftp-server logging enable
> sudo yast ftp-server logging disable
```

max_clients

Specifies the maximum number of concurrently connected clients:

```
> sudo yast ftp-server max_clients set_max_clients=1500
```

max_clients_ip

Specifies the maximum number of concurrently connected clients via IP:

```
> sudo yast ftp-server max_clients_ip set_max_clients=20
```

max_rate_anon

Specifies the maximum data transfer rate permitted for anonymous clients (KB/s):

```
> sudo yast ftp-server max_rate_anon set_max_rate=10000
```

max_rate_authen

Specifies the maximum data transfer rate permitted for locally authenticated users (KB/s):

```
> sudo yast ftp-server max_rate_authen set_max_rate=10000
```

port_range

Specifies the port range for passive connection replies:

```
> sudo yast ftp-server port_range set_min_port=20000 set_max_port=30000
```

For a complete list of options, run **`yast ftp-server port_range help`**.

show

Displays FTP server settings.

startup

Controls the FTP start-up method:

```
> sudo yast ftp-server startup atboot
```

For a complete list of options, run **`yast ftp-server startup help`**.

umask

Specifies the file umask for `authenticated:anonymous` users:

```
> sudo yast ftp-server umask set_umask=177:077
```

welcome_message

Specifies the text to display when someone connects to the FTP server:

```
> sudo yast ftp-server welcome_message set_message="hello everybody"
```

4.4.3.8 yast http-server

Configures the HTTP server (Apache2). **`yast http-server`** accepts the following commands:

configure

Configures the HTTP server host settings:

```
> sudo yast http-server configure host=main servername=www.example.com \
```

```
serveradmin=admin@example.com
```

For a complete list of options, run **yast http-server configure help**.

hosts

Configures virtual hosts:

```
> sudo yast http-server hosts create servername=www.example.com \
 serveradmin=admin@example.com documentroot=/var/www
```

For a complete list of options, run **yast http-server hosts help**.

listen

Specifies the ports and network addresses where the HTTP server should listen:

```
> sudo yast http-server listen add=81
> sudo yast http-server listen list
Listen Statements:
==================
:80
:81
> sudo yast http-server delete=80
```

For a complete list of options, run **yast http-server listen help**.

mode

Enables or disables the wizard mode:

```
> sudo yast http-server mode wizard=on
```

modules

Controls the Apache2 server modules:

```
> sudo yast http-server modules enable=php5,rewrite
> sudo yast http-server modules disable=ssl
> sudo http-server modules list
[...]
Enabled rewrite
Disabled ssl
Enabled php5
[...]
```

4.4.3.9 yast kdump

Configures kdump settings. For more information on kdump, refer to the *Book "System Analysis and Tuning Guide", Chapter 18 "Kexec and Kdump", Section 18.7 "Basic Kdump configuration"*. **yast kdump** accepts the following commands:

copykernel

Copies the kernel into the dump directory.

customkernel

Specifies the *kernel_string* part of the name of the custom kernel. The naming scheme is /boot/vmlinu[zx]-*kernel_string*[.gz].

```
> sudo yast kdump customkernel kernel=kdump
```

For a complete list of options, run **yast kdump customkernel help**.

dumpformat

Specifies the (compression) format of the dump kernel image. Available formats are "none", "ELF", "compressed" or "lzo":

```
> sudo yast kdump dumpformat dump_format=ELF
```

dumplevel

Specifies the dump level number in the range from 0 to 31:

```
> sudo yast kdump dumplevel dump_level=24
```

dumptarget

Specifies the destination for saving dump images:

```
> sudo kdump dumptarget target=ssh server=name_server port=22 \
 dir=/var/log/dump user=user_name
```

For a complete list of options, run **yast kdump dumptarget help**.

immediatereboot

Controls whether the system should reboot immediately after saving the core in the Kdump kernel:

```
> sudo yast kdump immediatereboot enable
> sudo yast kdump immediatereboot disable
```

keepolddumps

Specifies how many old dump images are kept. Specify zero to keep them all:

```
> sudo yast kdump keepolddumps no=5
```

kernelcommandline

Specifies the command line that needs to be passed off to the Kdump kernel:

```
> sudo yast kdump kernelcommandline command="ro root=LABEL=/"
```

kernelcommandlineappend

Specifies the command line that you need to *append* to the default command line string:

```
> sudo yast kdump kernelcommandlineappend command="ro root=LABEL=/"
```

notificationcc

Specifies an e-mail address for sending copies of notification messages:

```
> sudo yast kdump notificationcc email="user1@example.com user2@example.com"
```

notificationto

Specifies an e-mail address for sending notification messages:

```
> sudo yast kdump notificationto email="user1@example.com user2@example.com"
```

show

Displays kdump settings:

```
> sudo yast kdump show
Kdump is disabled
Dump Level: 31
Dump Format: compressed
Dump Target Settings
target: file
file directory: /var/crash
Kdump immediate reboots: Enabled
Numbers of old dumps: 5
```

smtppass

Specifies the file with the plain text SMTP password used for sending notification messages:

```
> sudo yast kdump smtppass pass=/path/to/file
```

smtpserver

Specifies the SMTP server host name used for sending notification messages:

```
> sudo yast kdump smtpserver server=smtp.server.com
```

smtpuser

Specifies the SMTP user name used for sending notification messages:

```
> sudo yast kdump smtpuser user=smtp_user
```

startup

Enables or disables start-up options:

```
> sudo yast kdump startup enable alloc_mem=128,256
> sudo yast kdump startup disable
```

4.4.3.10 yast keyboard

Configures the system keyboard for virtual consoles. It does not affect the keyboard settings in graphical desktop environments, such as GNOME or KDE. **`yast keyboard`** accepts the following commands:

list

Lists all available keyboard layouts.

set

Activates new keyboard layout setting:

```
> sudo yast keyboard set layout=czech
```

summary

Displays the current keyboard configuration.

4.4.3.11 yast lan

Configures network cards. **`yast lan`** accepts the following commands:

add

Configures a new network card:

```
> sudo yast lan add name=vlan50 ethdevice=eth0 bootproto=dhcp
```

For a complete list of options, run **`yast lan add help`**.

delete

Deletes an existing network card:

```
> sudo yast lan delete id=0
```

edit

Changes the configuration of an existing network card:

```
> sudo yast lan edit id=0 bootproto=dhcp
```

list

Displays a summary of network card configuration:

```
> sudo yast lan list
id name,           bootproto
0 Ethernet Card 0, NONE
1 Network Bridge,  DHCP
```

4.4.3.12 yast language

Configures system languages. **yast language** accepts the following commands:

list

Lists all available languages.

set

Specifies the main system languages and secondary languages:

```
> sudo yast language set lang=cs_CZ languages=en_US,es_ES no_packages
```

4.4.3.13 yast mail

Displays the configuration of the mail system:

```
> sudo yast mail summary
```

4.4.3.14 yast nfs

Controls the NFS client. **yast nfs** accepts the following commands:

add

Adds a new NFS mount:

```
> sudo yast nfs add spec=remote_host:/path/to/nfs/share file=/local/mount/point
```

For a complete list of options, run `yast nfs add help`.

delete

Deletes an existing NFS mount:

```
> sudo yast nfs delete spec=remote_host:/path/to/nfs/share file=/local/mount/point
```

For a complete list of options, run `yast nfs delete help`.

edit

Changes an existing NFS mount:

```
> sudo yast nfs edit spec=remote_host:/path/to/nfs/share \
 file=/local/mount/point type=nfs4
```

For a complete list of options, run `yast nfs edit help`.

list

Lists existing NFS mounts:

```
> sudo yast nfs list
Server            Remote File System    Mount Point    Options
----------------------------------------------------------------
nfs.example.com   /mnt                  /nfs/mnt       nfs
nfs.example.com   /home/tux/nfs_share   /nfs/tux       nfs
```

4.4.3.15 yast nfs-server

Configures the NFS server. `yast nfs-server` accepts the following commands:

add

Adds a directory to export:

```
> sudo yast nfs-server add mountpoint=/nfs/export hosts=*.allowed_hosts.com
```

For a complete list of options, run `yast nfs-server add help`.

delete

Deletes a directory from the NFS export:

```
> sudo yast nfs-server delete mountpoint=/nfs/export
```

set

Specifies additional parameters for the NFS server:

```
> sudo yast nfs-server set enablev4=yes security=yes
```

For a complete list of options, run **`yast nfs-server set help`**.

start

Starts the NFS server service:

```
> sudo yast nfs-server start
```

stop

Stops the NFS server service:

```
> sudo yast nfs-server stop
```

summary

Displays a summary of the NFS server configuration:

```
> sudo yast nfs-server summary
NFS server is enabled
NFS Exports
* /mnt
* /home

NFSv4 support is enabled.
The NFSv4 domain for idmapping is localdomain.
NFS Security using GSS is enabled.
```

4.4.3.16 yast nis

Configures the NIS client. **`yast nis`** accepts the following commands:

configure

Changes global settings of a NIS client:

```
> sudo yast nis configure server=nis.example.com broadcast=yes
```

For a complete list of options, run **`yast nis configure help`**.

disable

Disables the NIS client:

```
> sudo yast nis disable
```

enable

Enables your machine as NIS client:

```
> sudo yast nis enable server=nis.example.com broadcast=yes automounter=yes
```

For a complete list of options, run **`yast nis enable help`**.

find

Shows available NIS servers for a given domain:

```
> sudo yast nis find domain=nisdomain.com
```

summary

Displays a configuration summary of a NIS client.

4.4.3.17 yast nis-server

Configures a NIS server. **`yast nis-server`** accepts the following commands:

master

Configures a NIS master server:

```
> sudo yast nis-server master domain=nisdomain.com yppasswd=yes
```

For a complete list of options, run **`yast nis-server master help`**.

slave

Configures a NIS worker server:

```
> sudo yast nis-server slave domain=nisdomain.com master_ip=10.100.51.65
```

For a complete list of options, run **`yast nis-server slave help`**.

stop

Stops a NIS server:

```
> sudo yast nis-server stop
```

summary

Displays a configuration summary of a NIS server:

```
> sudo yast nis-server summary
```

4.4.3.18 yast proxy

Configures proxy settings. **`yast proxy`** accepts the following commands:

authentication

Specifies the authentication options for proxy:

```
> sudo yast proxy authentication username=tux password=secret
```

For a complete list of options, run **`yast proxy authentication help`**.

enable, disable

Enables or disables proxy settings.

set

Changes the current proxy settings:

```
> sudo yast proxy set https=proxy.example.com
```

For a complete list of options, run **`yast proxy set help`**.

summary

Displays proxy settings.

4.4.3.19 yast rdp

Controls remote desktop settings. **`yast rdp`** accepts the following commands:

allow

Allows remote access to the server's desktop:

```
> sudo yast rdp allow set=yes
```

list

Displays the remote desktop configuration summary.

4.4.3.20 yast samba-client

Configures the Samba client settings. **`yast samba-client`** accepts the following commands:

configure

Changes global settings of Samba:

```
> sudo yast samba-client configure workgroup=FAMILY
```

isdomainmember

Checks whether the machine is a member of a domain:

```
> sudo yast samba-client isdomainmember domain=SMB_DOMAIN
```

joindomain

Makes the machine a member of a domain:

```
> sudo yast samba-client joindomain domain=SMB_DOMAIN user=username password=pwd
```

winbind

Enables or disables Winbind services (the `winbindd` daemon):

```
> sudo yast samba-client winbind enable
> sudo yast samba-client winbind disable
```

4.4.3.21 yast samba-server

Configures Samba server settings. **`yast samba-server`** accepts the following commands:

backend

Specifies the back-end for storing user information:

```
> sudo yast samba-server backend smbpasswd
```

For a complete list of options, run **`yast samba-server backend help`**.

configure

Configures global settings of the Samba server:

```
> sudo yast samba-server configure workgroup=FAMILY description='Home server'
```

For a complete list of options, run **`yast samba-server configure help`**.

list

Displays a list of available shares:

```
> sudo yast samba-server list
Status     Type Name
==============================
Disabled   Disk profiles
Enabled    Disk print$
```

```
Enabled    Disk homes
Disabled   Disk groups
Enabled    Disk movies
Enabled    Printer printers
```

role

Specifies the role of the Samba server:

```
> sudo yast samba-server role standalone
```

For a complete list of options, run **`yast samba-server role help`**.

service

Enables or disables the Samba services (`smb` and `nmb`):

```
> sudo yast samba-server service enable
> sudo yast samba-server service disable
```

share

Manipulates a single Samba share:

```
> sudo yast samba-server share name=movies browseable=yes guest_ok=yes
```

For a complete list of options, run **`yast samba-server share help`**.

4.4.3.22 yast security

Controls the security level of the host. **`yast security`** accepts the following commands:

level

Specifies the security level of the host:

```
> sudo yast security level server
```

For a complete list of options, run **`yast security level help`**.

set

Sets the value of a specific option:

```
> sudo yast security set passwd=sha512 crack=yes
```

For a complete list of options, run **`yast security set help`**.

summary

Displays a summary of the current security configuration:

```
sudoyast security summary
```

4.4.3.23 yast sound

Configures sound card settings. **`yast sound`** accepts the following commands:

add

Configures a new sound card. Without any parameters, the command adds the first detected card.

```
> sudo yast sound add card=0 volume=75
```

For a complete list of options, run **`yast sound add help`**.

channels

Lists available volume channels of a sound card:

```
> sudo yast sound channels card=0
Master 75
PCM 100
```

modules

Lists all available sound kernel modules:

```
> sudo yast sound modules
snd-atiixp ATI IXP AC97 controller (snd-atiixp)
snd-atiixp-modem ATI IXP MC97 controller (snd-atiixp-modem)
snd-virtuoso Asus Virtuoso driver (snd-virtuoso)
[...]
```

playtest

Plays a test sound on a sound card:

```
> sudo yast sound playtest card=0
```

remove

Removes a configured sound card:

```
> sudo yast sound remove card=0
```

```
> sudo yast sound remove all
```

set

Specifies new values for a sound card:

```
> sudo yast sound set card=0 volume=80
```

show

Displays detailed information about a sound card:

```
> sudo yast sound show card=0
Parameters of card 'ThinkPad X240' (using module snd-hda-intel):

align_buffer_size
 Force buffer and period sizes to be multiple of 128 bytes.
bdl_pos_adj
 BDL position adjustment offset.
beep_mode
 Select HDA Beep registration mode (0=off, 1=on) (default=1).
 Default Value: 0
enable_msi
 Enable Message Signaled Interrupt (MSI)
[...]
```

summary

Prints a configuration summary for all sound cards on the system:

```
> sudo yast sound summary
```

volume

Specifies the volume level of a sound card:

```
sudoyast sound volume card=0 play
```

4.4.3.24 yast sysconfig

Controls the variables in files under `/etc/sysconfig`. **`yast sysconfig`** accepts the following commands:

clear

Sets empty value to a variable:

```
> sudo yast sysconfig clear=POSTFIX_LISTEN
```

Tip: Variable in multiple files

If the variable is available in several files, use the *VARIABLE_NAME*$*FILE_NAME* syntax:

```
> sudo yast sysconfig clear=CONFIG_TYPE$/etc/sysconfig/mail
```

details

Displays detailed information about a variable:

```
> sudo yast sysconfig details variable=POSTFIX_LISTEN
Description:
Value:
File: /etc/sysconfig/postfix
Possible Values: Any value
Default Value:
Configuration Script: postfix
Description:
 Comma separated list of IP's
 NOTE: If not set, LISTEN on all interfaces
```

list

Displays summary of modified variables. Use `all` to list all variables and their values:

```
> sudo yast sysconfig list all
AOU_AUTO_AGREE_WITH_LICENSES="false"
AOU_ENABLE_CRONJOB="true"
AOU_INCLUDE_RECOMMENDS="false"
[...]
```

set

Sets a value for a variable:

```
> sudo yast sysconfig set DISPLAYMANAGER=gdm
```

Tip: Variable in multiple files

If the variable is available in several files, use the *VARIABLE_NAME*$*FILE_NAME* syntax:

```
> sudo yast sysconfig set CONFIG_TYPE$/etc/sysconfig/mail=advanced
```

4.4.3.25 yast tftp-server

Configures a TFTP server. **`yast tftp-server`** accepts the following commands:

directory

Specifies the directory of the TFTP server:

```
> sudo yast tftp-server directory path=/srv/tftp
> sudo yast tftp-server directory list
Directory Path: /srv/tftp
```

status

Controls the status of the TFTP server service:

```
> sudo yast tftp-server status disable
> sudo yast tftp-server status show
Service Status: false
> sudo yast tftp-server status enable
```

4.4.3.26 yast timezone

Configures the time zone. **`yast timezone`** accepts the following commands:

list

Lists all available time zones grouped by region:

```
> sudo yast timezone list
Region: Africa
Africa/Abidjan (Abidjan)
Africa/Accra (Accra)
Africa/Addis_Ababa (Addis Ababa)
[...]
```

set

Specifies new values for the time zone configuration:

```
> sudo yast timezone set timezone=Europe/Prague hwclock=local
```

summary

Displays the time zone configuration summary:

```
> sudo yast timezone summary
Current Time Zone: Europe/Prague
```

```
Hardware Clock Set To: Local time
Current Time and Date: Mon 12. March 2018, 11:36:21 CET
```

4.4.3.27 yast users

Manages user accounts. **`yast users`** accepts the following commands:

add

Adds a new user:

```
> sudo yast users add username=user1 password=secret home=/home/user1
```

For a complete list of options, run **`yast users add help`**.

delete

Deletes an existing user account:

```
> sudo yast users delete username=user1 delete_home
```

For a complete list of options, run **`yast users delete help`**.

edit

Changes an existing user account:

```
> sudo yast users edit username=user1 password=new_secret
```

For a complete list of options, run **`yast users edit help`**.

list

Lists existing users filtered by user type:

```
> sudo yast users list system
```

For a complete list of options, run **`yast users list help`**.

show

Displays details about a user:

```
> sudo yast users show username=wwwrun
Full Name: WWW daemon apache
List of Groups: www
Default Group: wwwrun
Home Directory: /var/lib/wwwrun
Login Shell: /sbin/nologin
```

```
Login Name: wwwrun
UID: 456
```

For a complete list of options, run **`yast users show help`**.

5 Changing language and country settings with YaST

Revision History

2024-05-13

This chapter explains how to configure language and country settings. You can change the language globally for the whole system, individually for certain users or desktops, or temporarily for single applications. Additionally, you can configure secondary languages and adjust the date and country settings.

If you work in different countries or in a multilingual environment, you should configure your system accordingly. SUSE® Linux Enterprise Desktop can handle different `locales` in parallel. A locale is a set of parameters that defines the language and country settings reflected in the user interface.

The main system language is selected during installation, and keyboard and time zone settings are adjusted accordingly. However, you can install additional languages and determine which of the installed languages should be the default.

For those tasks, use the YaST language module as described in *Section 5.1, “Changing the system language”*. Install secondary languages to get optional localization if you need to start applications or desktops in languages other than the primary one.

The YaST time zone module allows you to adjust your country and time zone settings accordingly. It also lets you synchronize your system clock against a time server. For details, refer to *Section 5.2, “Changing the country and time settings”*.

5.1 Changing the system language

Depending on how you use your desktop and whether you want to switch the entire system to another language or only the desktop environment, you have several options:

Changing the system language globally

Proceed as described in *Section 5.1.1, “Modifying system languages with YaST”* and *Section 5.1.2, “Switching the default system language”* to install additional localized packages with YaST and to set the default language. Changes are effective after the next login. To ensure that the entire system reflects the change, reboot the system or close and restart all running services, applications and programs.

Changing the language for the desktop only

Provided you have previously installed the desired language packages for your desktop environment with YaST as described below, you can switch the language of your desktop using the desktop's control center. Refer to *Book "GNOME User Guide", Chapter 3 "Customizing your settings", Section 3.2 "Configuring language settings"* for details. After the X server has been restarted, your entire desktop reflects your new choice of language. Applications not belonging to your desktop framework are not affected by this change and may still appear in the language that was set in YaST.

Temporarily switching languages for one application only

You can also run a single application in another language (that has already been installed with YaST). To do so, start it from the command line by specifying the language code as described in *Section 5.1.3, "Switching languages for standard X and GNOME applications"*.

5.1.1 Modifying system languages with YaST

YaST supports two different language categories:

Primary Language

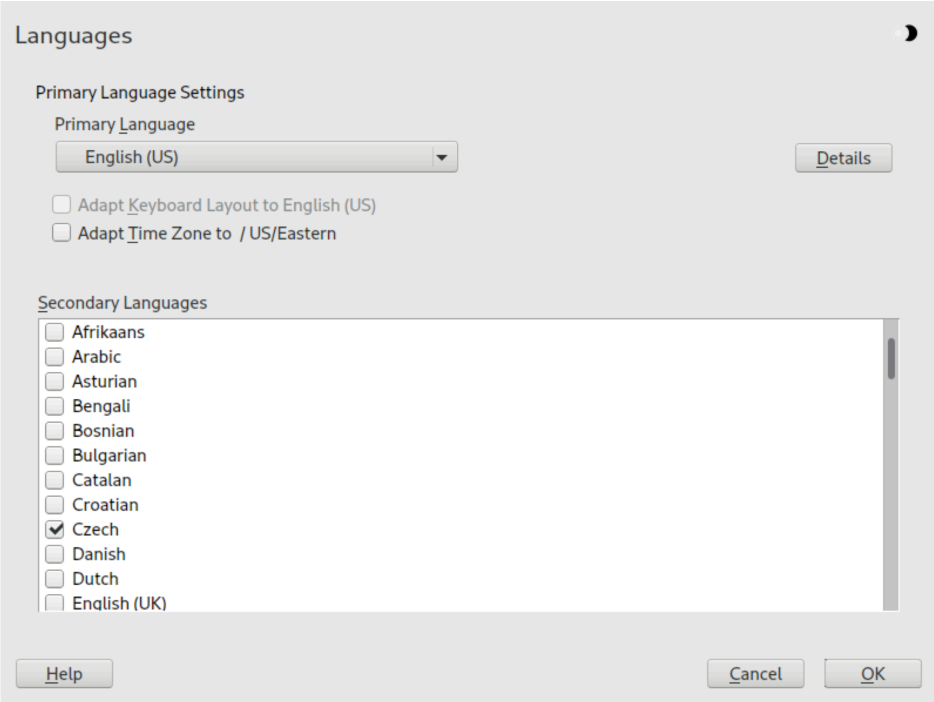
The primary language set in YaST applies to the entire system, including YaST and the desktop environment. This language is used whenever available unless you manually specify another language.

Secondary Languages

Install secondary languages to make your system multilingual. Languages installed as secondary can be selected manually, when needed. For example, use a secondary language to start an application in a certain language to do word processing in this language.

Before installing additional languages, determine which of them should be the default system language (primary language).

To access the YaST language module, start YaST and click *System* › *Language*. Alternatively, start the *Languages* dialog directly by running **`sudo yast2 language &`** from a command line.



PROCEDURE 5.1: INSTALLING ADDITIONAL LANGUAGES

When installing additional languages, YaST allows you to set different locale settings for the user `root`, see *Step 4*. The option *Locale Settings for User root* determines how the locale variables (`LC_*`) in the file `/etc/sysconfig/language` are set for `root`. You can set them to the same locale as for regular users. Alternatively, you can keep them unaffected by any language changes, or only set the variable `RC_LC_CTYPE` to the same values as for the regular users. The `RC_LC_CTYPE` variable sets the localization for language-specific function calls.

1. To add languages in the YaST language module, select the *Secondary Languages* you want to install.
2. To make a language the default language, set it as *Primary Language*.
3. Additionally, adapt the keyboard to the new primary language and adjust the time zone, if appropriate.

> **Tip: Advanced settings**
>
> For advanced keyboard or time zone settings, select *Hardware* › *System Keyboard Layout* or *System* › *Date and Time* in YaST. For more information, refer to *Chapter 32, Setting up your system keyboard layout* and *Section 5.2, "Changing the country and time settings"*.

4. To change language settings specific to the user `root`, click *Details*.
   a. Set *Locale Settings for User root* to the desired value. For more information, click *Help*.
   b. Decide whether to use *Use UTF-8 Encoding* for `root` or not.
5. If your locale was not included in the list of primary languages available, try specifying it with *Detailed Locale Setting*. However, this may result in certain locales being incomplete.
6. Confirm the changes in the dialogs with *OK*. If you have selected secondary languages, YaST installs the localized software packages for the additional languages.

The system is now multilingual. However, to start an application in a language other than the primary one, you need to set the desired language explicitly as explained in *Section 5.1.3, "Switching languages for standard X and GNOME applications"*.

5.1.2 Switching the default system language

To globally change the default language of a system, use the following procedure:

1. Start the YaST language module.
2. Select the desired new system language as *Primary Language*.

> **Important: Deleting former system languages**
>
> If you switch to a different primary language, the localized software packages for the former primary language gets removed from the system. To switch the default system language but keep the former primary language as an additional language, add it as *Secondary Language* by selecting the respective check box.

3. Adjust the keyboard and time zone options as desired.

4. Confirm your changes with *OK*.

5. After YaST has applied the changes, restart current X sessions (for example, by logging out and logging in again) to make YaST and the desktop applications reflect your new language settings.

5.1.3 Switching languages for standard X and GNOME applications

After you have installed the respective language with YaST, you can run a single application in another language.

Start the application from the command line by using the following command:

```
LANG=LANGUAGE application
```

For example, to start f-spot in German, run **`LANG=de_DE f-spot`**. For other languages, use the appropriate language code. Get a list of all language codes available with the **`locale`** `-av` command.

5.2 Changing the country and time settings

Using the YaST date and time module, adjust your system date, clock and time zone information to the area you are working in. To access the YaST module, start YaST and click *System* › *Date and Time*. Alternatively, start the *Clock and Time Zone* dialog directly by running **`sudo yast2 timezone &`** from a command line.

First, select a general region, such as *Europe*. Choose an appropriate country that matches the one you are working in, for example, *Germany*.

Depending on which operating systems run on your workstation, adjust the hardware clock settings accordingly:

- If you run another operating system on your machine, such as Microsoft Windows*, your system may not use UTC, but local time. In this case, deactivate *Hardware Clock Set To UTC*.
- If you only run Linux on your machine, set the hardware clock to UTC and have the switch from standard time to daylight saving time performed automatically.

Important: Set the hardware clock to UTC

The switch from standard time to daylight saving time (and vice versa) can only be performed automatically when the hardware clock (CMOS clock) is set to UTC. This also applies if you use automatic time synchronization with NTP, because automatic synchronization is only performed if the time difference between the hardware and system clock is less than 15 minutes.

Since a wrong system time can cause serious problems (missed backups, dropped mail messages, mount failures on remote file systems, etc.) it is strongly recommended to *always* set the hardware clock to UTC.

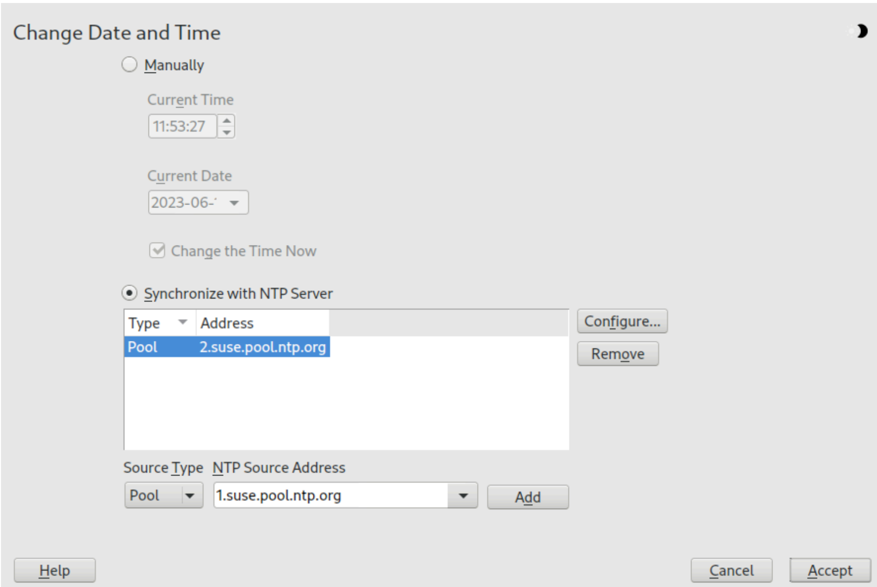
You can change the date and time manually or opt for synchronizing your machine against an NTP server, either permanently or only for adjusting your hardware clock.

PROCEDURE 5.2: MANUALLY ADJUSTING TIME AND DATE

1. In the YaST timezone module, click *Other Settings* to set date and time.
2. Select *Manually* and enter date and time values.
3. Confirm your changes.

PROCEDURE 5.3: SETTING DATE AND TIME WITH NTP SERVER

1. Click *Other Settings* to set date and time.
2. Select *Synchronize with NTP Server*.
3. Enter the address of an NTP server, if not already populated.



4. With the *Configure* button, you can open the advanced NTP configuration. For details, see *Section 39.1, "Configuring an NTP client with YaST"*.
5. Confirm your changes.

6 Managing users with YaST

Revision History

2024-07-16

During installation, you may have created a local user for your system. With the YaST module *User and Group Management* you can add users or edit existing ones. It also lets you configure your system to authenticate users with a network server.

6.1 User and group administration dialog

To administer users or groups, start YaST and click *Security and Users* › *User and Group Management*. Alternatively, start the *User and Group Administration* dialog directly by running **`sudo yast2 users &`** from a command line.

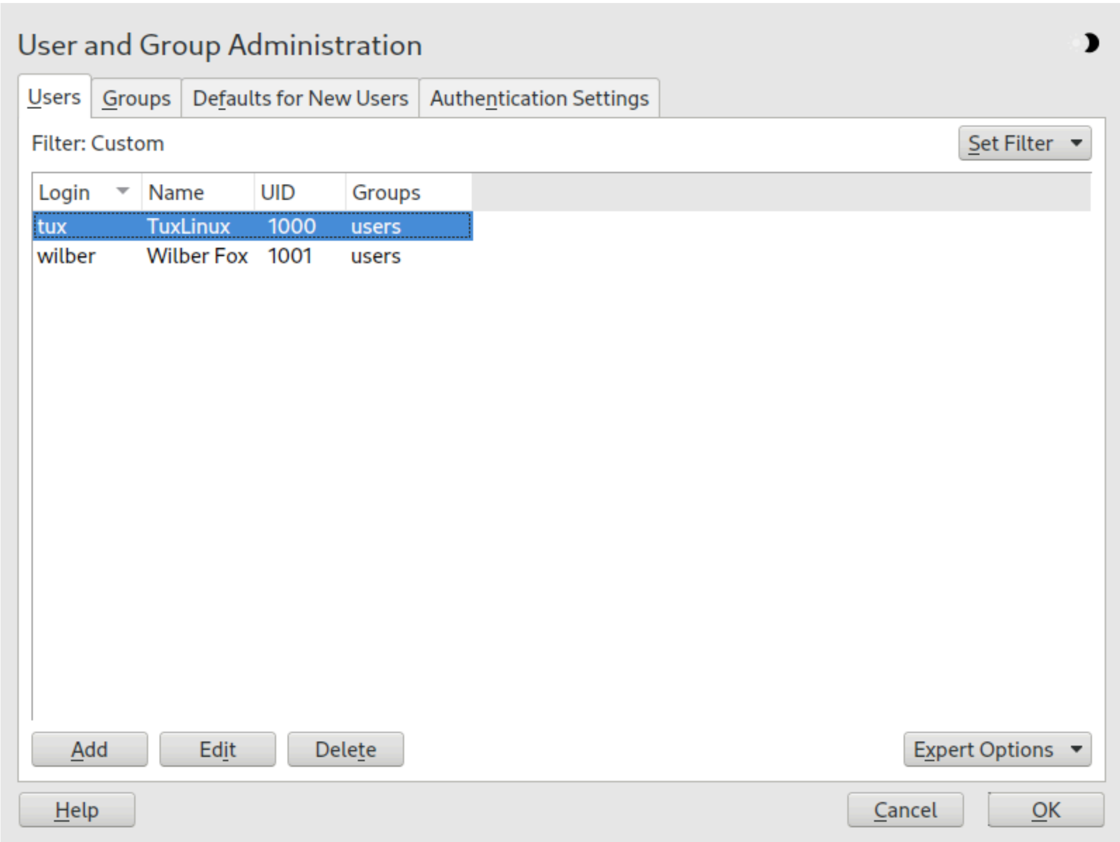


FIGURE 6.1: YAST USER AND GROUP ADMINISTRATION

Every user is assigned a system-wide user ID (UID). Apart from the users that can log in to your machine, there are also several *system users* for internal use only. Each user is assigned to one or more groups. Similar to *system users*, there are also *system groups* for internal use.

The main window shows several tabs, depending on the set of users (local users, network users, system users) you choose to view and modify. The tabs allow you to perform the following tasks:

Managing user accounts

: From the *Users* tab create, modify, delete or temporarily disable user accounts as described in *Section 6.2, "Managing user accounts"*. Learn about advanced options like enforcing password policies, using encrypted home directories, or managing disk quotas in *Section 6.3, "Additional options for user accounts"*.

Changing default settings

: Local user accounts are created according to the settings defined on the *Defaults for New Users* tab. Learn how to change the default group assignment, or the default path and access permissions for home directories in *Section 6.4, "Changing default settings for local users"*.

Assigning users to groups

: Learn how to change the group assignment for individual users in *Section 6.5, "Assigning users to groups"*.

Managing groups

: From the *Groups* tab, you can add, modify or delete existing groups. Refer to *Section 6.6, "Managing groups"* for information on how to do this.

Changing user authentication method

: When your machine is connected to a network that provides user authentication methods like NIS or LDAP, you can choose between several authentication methods on the *Authentication Settings* tab. For more information, refer to *Section 6.7, "Changing the user authentication method"*.

For user and group management, the dialog provides similar functionality. You can easily switch between the user and group administration view by choosing the appropriate tab at the top of the dialog.

Filter options allow you to define the set of users or groups you want to modify: on the *Users* or *Group* tab, click *Set Filter* to view and edit users or groups. They are listed according to certain categories, such as *Local Users* or *LDAP Users*, if applicable. With *Set Filter › Customize Filter* you can also set up and use a custom filter.

Depending on the filter you choose, not all the following options and functions may be available from the dialog.

6.2 Managing user accounts

YaST allows you to create, modify, delete or temporarily disable user accounts. Do not modify user accounts unless you are an experienced user or administrator.

Note: Changing user IDs of existing users

File ownership is bound to the user ID, not to the user name. After a user ID change, the files in the user's home directory are automatically adjusted to reflect this change. However, after an ID change, the user no longer owns the files they created elsewhere in the file system unless the file ownership for those files is manually modified.

The following instructions demonstrate how to set up default user accounts. For further options, refer to *Section 6.3, "Additional options for user accounts"*.

PROCEDURE 6.1: ADDING OR MODIFYING USER ACCOUNTS

1. Open the YaST *User and Group Administration* dialog and click the *Users* tab.
2. With *Set Filter* define the set of users you want to manage. The dialog lists users in the system and the groups the users belong to.
3. To modify options for an existing user, select an entry and click *Edit*.
   To create a new user account, click *Add*.
4. Enter the appropriate user data on the first tab, such as *Username* (which is used for login) and *Password*. This data is sufficient to create a new user. If you click *OK* now, the system automatically assigns a user ID and sets all other values as default.
5. Activate *Receive System Mail* if you want system notifications to be delivered to this user's mailbox. This creates a mail alias for `root` and the user can read the system mail without having to first log in as `root`.
   The mails sent by system services are stored in the local mailbox `/var/spool/mail/USERNAME`, where `USERNAME` is the login name of the selected user. To read e-mails, you can use the **`mail`** command.

6. To adjust further details such as the user ID or the path to the user's home directory, do so on the *Details* tab.
   If you need to relocate the home directory of an existing user, enter the path to the new home directory there and move the contents of the current home directory with *Move to New Location*. Otherwise, a new home directory is created without any of the existing data.

7. To force users to regularly change their password or set other password options, switch to *Password Settings* and adjust the options. For more details, refer to *Section 6.3.2, "Enforcing password policies"*.

8. If all options are set according to your wishes, click *OK*.

9. Click *OK* to close the administration dialog and to save the changes. A newly added user can now log in to the system using the login name and password you created.
   Alternatively, to save all changes without exiting the *User and Group Administration* dialog, click *Expert Options* › *Write Changes Now*.

Warning: Do not rename the `root` account

While it is technically possible to rename the `root` account, certain applications, scripts or third-party products may rely on the existence of a user called `root`. While such a configuration always targets individual environments, necessary adjustments could be overwritten by vendor updates, so this becomes an ongoing task rather than a one-time setting. This is especially true in complex setups involving third-party applications, where it needs to be verified with every vendor involved whether a rename of the `root` account is supported.

As the implications for renaming the `root` account cannot be foreseen, SUSE does not support renaming the `root` account.

Usually, the idea behind renaming the `root` account is to hide it or make it unpredictable. However, `/etc/passwd` requires `644` permissions for regular users, so any user of the system can retrieve the login name for the user ID 0. For better ways to secure the `root` account, refer to *Book "Security and Hardening Guide", Chapter 14 "User management", Section 14.5 "Restricting* `root` *logins"* and *Book "Security and Hardening Guide", Chapter 14 "User management", Section 14.5.3 "Restricting SSH logins"*.

Tip: Matching user IDs

It is useful to match the (local) user ID to the ID in the network. For example, a new (local) user on a laptop should be integrated into a network environment with the same user ID. This ensures that the file ownership of the files the user creates “offline” is the same as if they had created them directly on the network.

PROCEDURE 6.2: DISABLING OR DELETING USER ACCOUNTS

1. Open the YaST *User and Group Administration* dialog and click the *Users* tab.
2. To temporarily disable a user account without deleting it, select the user from the list and click *Edit*. Activate *Disable User Login*. The user cannot log in to your machine until you enable the account again.
3. To delete a user account, select the user from the list and click *Delete*. Choose if you also want to delete the user's home directory or to retain the data.

6.3 Additional options for user accounts

Besides the settings for a default user account, SUSE® Linux Enterprise Desktop offers further options. For example, options to enforce password policies, use encrypted home directories or define disk quotas for users and groups.

6.3.1 Automatic login and passwordless login

If you use the GNOME desktop environment you can configure *Auto Login* for a certain user and *Passwordless Login* for all users. Auto login causes a user to become automatically logged in to the desktop environment on boot. This functionality can only be activated for one user at a time. Login without password allows all users to log in to the system after they have entered their user name in the login manager.

Warning: Security risk

Enabling *Auto Login* or *Passwordless Login* on a machine that can be accessed by more than one person is a security risk. Without the need to authenticate, any user can gain access to your system and your data. If your system contains confidential data, do not use this functionality.

To activate auto login or login without password, access these functions in the YaST *User and Group Administration* with *Expert Options* › *Login Settings*.

6.3.2 Enforcing password policies

On any system with multiple users, it is a good idea to enforce at least basic password security policies. Users should change their passwords regularly and use strong passwords that cannot easily be exploited. For local users, proceed as follows:

PROCEDURE 6.3: CONFIGURING PASSWORD SETTINGS

1. Open the YaST *User and Group Administration* dialog and select the *Users* tab.
2. Select user and click *Edit*.
3. Switch to the *Password Settings* tab. The user's last password change is displayed on the tab.
4. To make the user change their password at next login, activate *Force Password Change*.
5. To enforce password rotation, set a *Maximum Number of Days for the Same Password* and a *Minimum Number of Days for the Same Password*.
6. To remind the user to change their password before it expires, set the number of *Days before Password Expiration to Issue Warning*.
7. To restrict the period of time the user can log in after their password has expired, change the value in *Days after Password Expires with Usable Login*.
8. You can also specify a certain expiration date for the complete account. Enter the *Expiration Date* in `YYYY-MM-DD` format. This setting is not password-related but rather applies to the account itself.
9. For more information about options and default values, click *Help*.

10. Apply your changes with *OK*.

6.3.3 Managing quotas

To prevent system capacities from being exhausted without notification, system administrators can set up quotas for users or groups. Quotas can be defined for one or more file systems and restrict the amount of disk space that can be used and the number of inodes (index nodes) that can be created there. Inodes are data structures on a file system that store basic information about a regular file, directory or other file system object. They store all attributes of a file system object (like user and group ownership, read, write or execute permissions), except file name and contents.

SUSE Linux Enterprise Desktop allows usage of `soft` and `hard` quotas. Additionally, grace intervals can be defined that allow users or groups to temporarily exceed their quotas by certain amounts.

Soft quota

Defines a warning level at which users are informed that they are nearing their limit. Administrators may urge the users to clean up and reduce their data on the partition. The soft quota limit is normally lower than the hard quota limit.

Hard quota

Defines the limit at which write requests are denied. When the hard quota is reached, no more data can be stored and applications may crash.

Grace period

Defines the time between the overflow of the soft quota and a warning being issued. Normally set to a rather low value of one or several hours.

PROCEDURE 6.4: ENABLING QUOTA SUPPORT FOR A PARTITION

To configure quotas for certain users and groups, you need to enable quota support for the respective partition in the YaST Expert Partitioner first.

1. In YaST, select *System* › *Partitioner* and click *Yes* to proceed.
2. In the *Expert Partitioner*, select the partition for which to enable quotas and click *Edit*.
3. Click *Fstab Options* and activate *Enable Quota Support*. If the `quota` package is not already installed, it will be installed when you confirm the respective message with *Yes*.
4. Confirm your changes and leave the *Expert Partitioner*.

5. Make sure the service `quotaon` is running by entering the following command:

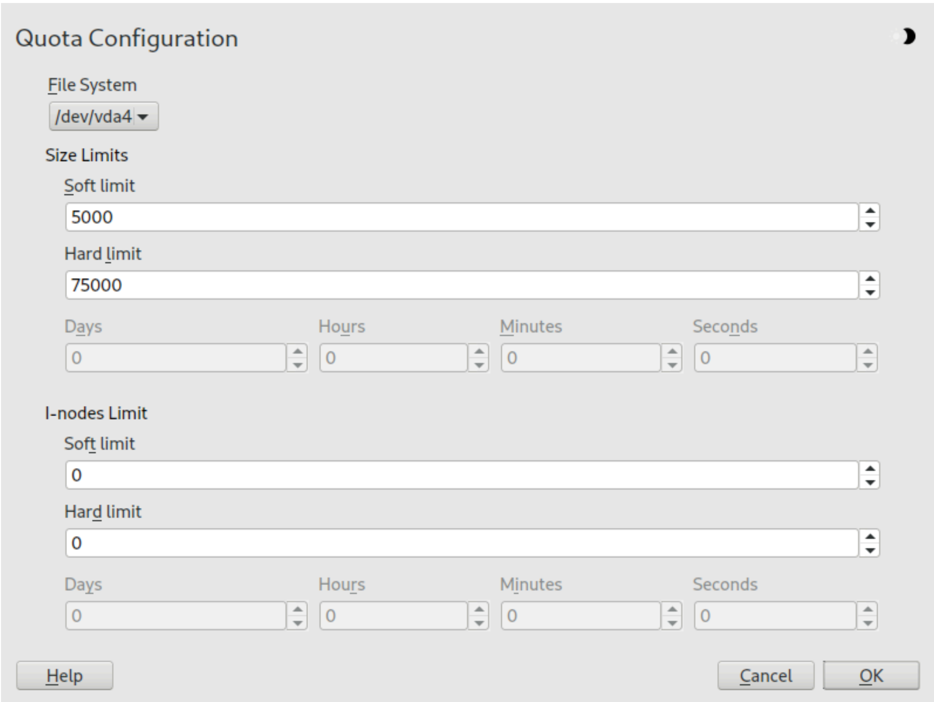
```
> sudo systemctl status quotaon.service
```

It should be marked as being `active`. If this is not the case, start it with the command **`systemctl start quotaon.service`**.

PROCEDURE 6.5: SETTING UP QUOTAS FOR USERS OR GROUPS

Now you can define soft or hard quotas for specific users or groups and set time periods as grace intervals.

1. In the YaST *User and Group Administration*, select the user or the group you want to set the quotas for and click *Edit*.
2. On the *Plug-Ins* tab, select the *Manage User Quota* entry and click *Launch* to open the *Quota Configuration* dialog.
3. From *File System*, select the partition to which the quota should apply.



4. Below *Size Limits*, restrict the amount of disk space. Enter the number of 1 KB blocks the user or group may have on this partition. Specify a *Soft Limit* and a *Hard Limit* value.
5. Additionally, you can restrict the number of inodes the user or group may have on the partition. Below *Inodes Limits*, enter a *Soft Limit* and *Hard Limit*.

6. You can only define grace intervals if the user or group has already exceeded the soft limit specified for size or inodes. Otherwise, the time-related text boxes are not activated. Specify the time period for which the user or group is allowed to exceed the limits set above.

7. Confirm your settings with *OK*.

8. Click *OK* to close the administration dialog and save the changes.
   Alternatively, to save all changes without exiting the *User and Group Administration* dialog, click *Expert Options* › *Write Changes Now*.

SUSE Linux Enterprise Desktop also ships command line tools like `repquota` or `warnquota`. System administrators can use these tools to control the disk usage or send e-mail notifications to users exceeding their quota. Using **`quota_nld`**, administrators can also forward kernel messages about exceeded quotas to D-BUS. For more information, refer to the `repquota`, the `warnquota` and the **`quota_nld`** man page.

6.4 Changing default settings for local users

When creating new local users, several default settings are used by YaST. These include, for example, the group the user belongs to, or the access permissions of the user's home directory. You can change these default settings to meet your requirements:

1. Open the YaST *User and Group Administration* dialog and select the *Defaults for New Users* tab.

2. To change the group the new users should automatically belong to, select another group from *Default Group*.

3. If you do not want to use `/home/USERNAME` as the default path for new users' home directories, modify the *Path Prefix for Home Directory*.

4. To change the default permission modes for newly created home directories, adjust the umask value in *Umask for Home Directory*. For more information about umask, refer to *Book "Security and Hardening Guide", Chapter 19 "Access control lists in Linux"* and to the **`umask`** man page.

5. For information about the individual options, click *Help*.

6. Apply your changes with *OK*.

6.5 Assigning users to groups

Local users are assigned to several groups according to the default settings, which you can access from the *User and Group Administration* dialog on the *Defaults for New Users* tab. In the following, learn how to modify an individual user's group assignment. If you need to change the default group assignments for new users, refer to *Section 6.4, "Changing default settings for local users"*.

PROCEDURE 6.6: CHANGING A USER'S GROUP ASSIGNMENT

1. Open the YaST *User and Group Administration* dialog and click the *Users* tab. It lists users and the groups the users belong to.
2. Click *Edit* and switch to the *Details* tab.
3. To change the group the user belongs to, click *Default Group* and select the group from the list.
4. To assign the user additional secondary groups, activate the corresponding check boxes in the *Additional Groups* list.
5. Click *OK* to apply your changes.
6. Click *OK* to close the administration dialog and save the changes.
   Alternatively, to save all changes without exiting the *User and Group Administration* dialog, click *Expert Options* › *Write Changes Now*.

6.6 Managing groups

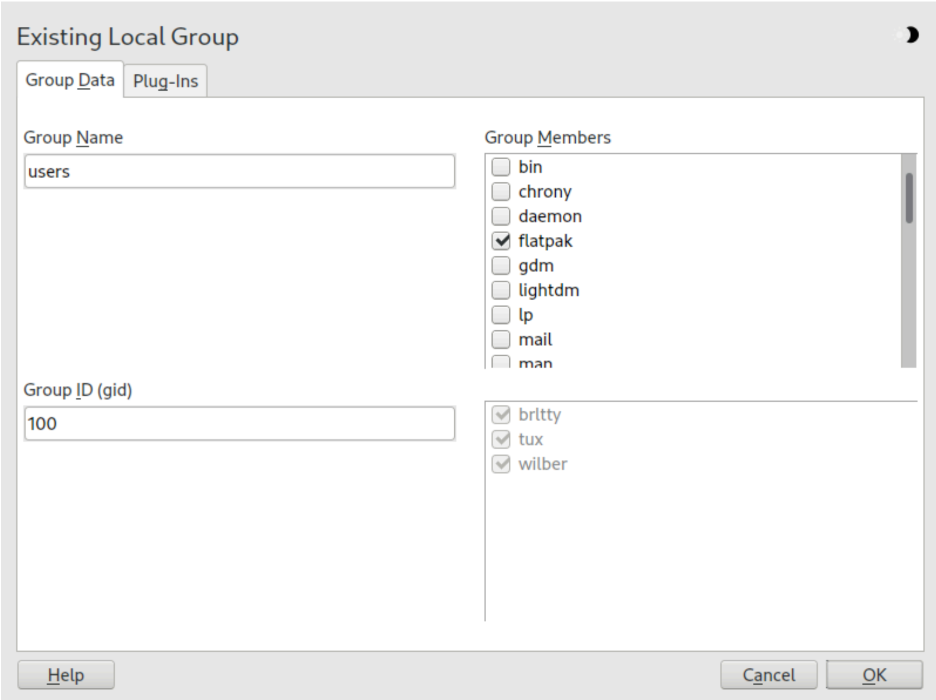
With YaST you can also easily add, modify or delete groups.

PROCEDURE 6.7: CREATING AND MODIFYING GROUPS

1. Open the YaST *User and Group Management* dialog and click the *Groups* tab.
2. With *Set Filter* define the set of groups you want to manage. The dialog lists groups in the system.
3. To create a new group, click *Add*.

4. To modify an existing group, select the group and click *Edit*.

5. In the following dialog, enter or change the data. The list on the right shows an overview of all available users and system users which can be members of the group.



6. To add existing users to a new group select them from the list of possible *Group Members* by checking the corresponding box. To remove them from the group deactivate the box.

7. Click *OK* to apply your changes.

8. Click *OK* to close the administration dialog and save the changes.
Alternatively, to save all changes without exiting the *User and Group Administration* dialog, click *Expert Options* › *Write Changes Now*.

To delete a group, it must not contain any group members. To delete a group, select it from the list and click *Delete*. Click *OK* to close the administration dialog and save the changes. Alternatively, to save all changes without exiting the *User and Group Administration* dialog, click *Expert Options* › *Write Changes Now*.

6.7 Changing the user authentication method

When your machine is connected to a network, you can change the authentication method. The following options are available:

NIS

Users are administered centrally on a NIS server for all systems in the network. For details, see *Book “Security and Hardening Guide”, Chapter 3 “Using NIS”*.

SSSD

The *System Security Services Daemon* (SSSD) can locally cache user data and then allow users to use the data, even if the real directory service is (temporarily) unreachable. For details, see *Book “Security and Hardening Guide”, Chapter 4 “Setting up authentication clients using YaST”, Section 4.2 “SSSD”*.

Samba

SMB authentication is often used in mixed Linux and Windows networks. For details, see *Book “Security and Hardening Guide”, Chapter 7 “Active Directory support”*.

To change the authentication method, proceed as follows:

1. Open the *User and Group Administration* dialog in YaST.
2. Click the *Authentication Settings* tab to show an overview of the available authentication methods and the current settings.
3. To change the authentication method, click *Configure* and select the authentication method you want to modify. This takes you directly to the client configuration modules in YaST. For information about the configuration of the appropriate client, refer to the following sections:

   NIS: *Book “Security and Hardening Guide”, Chapter 3 “Using NIS”, Section 3.2 “Configuring NIS clients”*

   LDAP: *Book “Security and Hardening Guide”, Chapter 4 “Setting up authentication clients using YaST”, Section 4.1 “Configuring an authentication client with YaST”*

   SSSD: *Book “Security and Hardening Guide”, Chapter 4 “Setting up authentication clients using YaST”, Section 4.2 “SSSD”*
4. After accepting the configuration, return to the *User and Group Administration* overview.

5. Click *OK* to close the administration dialog.

6.8 Default system users

By default, SUSE Linux Enterprise Desktop creates user names, which cannot be deleted. These users are typically defined in the Linux Standard Base (see https://refspecs.linuxfoundation.org/lsb.shtml ↗). The following list provides the common user names and their purpose:

COMMON USER NAMES INSTALLED BY DEFAULT

`bin`,
`daemon`
: Legacy user, included for compatibility with legacy applications. New applications should no longer use this user name.

`gdm`
: Used by GNOME Display Manager (GDM) to provide graphical logins and manage local and remote displays.

`lp`
: Used by the Printer daemon for Common Unix Printing System (CUPS).

`mail`
: User reserved for mailer programs like **`sendmail`** or **`postfix`**.

`man`
: Used by man to access man pages.

`messagebus`
: Used to access D-Bus (desktop bus), a software bus for inter-process communication. Daemon is `dbus-daemon`.

`nobody`
: User that owns no files and is in no privileged groups. Nowadays, its use is limited as it is recommended by Linux Standard Base to provide a separate user account for each daemon.

`nscd`
: Used by the Name Service Caching Daemon. This daemon is a lookup service to improve performance with NIS and LDAP. Daemon is `nscd`.

`polkitd`
: Used by the PolicyKit Authorization Framework, which defines and handles authorization requests for unprivileged processes. Daemon is `polkitd`.

`postfix`
: Used by the Postfix mailer.

`pulse`
: Used by the Pulseaudio sound server.

`root`
: Used by the system administrator, providing all appropriate privileges.

`rpc`
: Used by the **`rpcbind`** command, an RPC port mapper.

`rtkit`
: Used by the `rtkit` package providing a D-Bus system service for real time scheduling mode.

`salt`
: User for parallel remote execution provided by Salt. Daemon is named `salt-master`.

`scard`
: User for communication with smart cards and readers. Daemon is named `pcscd`.

`srvGeoClue`
: Used by the GeoClue D-Bus service to provide location information.

`sshd`
: Used by the Secure Shell daemon (SSH) to ensure secured and encrypted communication over an insecure network.

`statd`
: Used by the Network Status Monitor protocol (NSM), implemented in the `rpc.statd` daemon, to listen for reboot notifications.

`systemd-coredump`
: Used by the **`/usr/lib/systemd/systemd-coredump`** command to acquire, save and process core dumps.

`systemd-timesync`
: Used by the **`/usr/lib/systemd/systemd-timesyncd`** command to synchronize the local system clock with a remote Network Time Protocol (NTP) server.

6.9 Default system groups

By default, SLE creates multiple user groups that are used by system services. The following list describes examples of required and common optional groups.

`root`
: Administrative group with all privileges.

`bin`
: Included for compatibility with legacy applications. New applications should not use this group.

`daemon`
: Previously used to limit daemons' access to the system. Daemons should run under their own UID/GID now to separate daemons from one another.

`audio`
: Privileges for audio devices.

`gdm`
: Privileges for the GNOME Display Manager.

`chrony`
: Privileges for the time synchronization service.

`kvm`
: Privileges for the QEMU machine emulator toolkit.

`libvirt`
: Privileges for virtualization stack.

`lp`
: Privileges for printer operation.

`mail`
: Privileges for mail services.

`man`
: Privileges specific to manual pages and the **`man`** command.

`sshd`
: Privileges for SSH communication protocol daemon.

6.10 Reserved user and group IDs

When a user creates a file, they automatically become the owner of that file. Additionally, the file is associated with a group, and the creating user's primary group typically becomes the group owner. Files are assigned separate `read`, `write` and `execute` permissions. The root user can change the file owner. Both the root user and the file owner can change access permissions to the file. A regular user can change the group ownership of a file they own only if they are a member of that group.

Every user has a UID (unique user ID), which is a numerical identification number. Additionally, each group is associated with a GID (group ID). Users belonging to a group share the same `read`, `write` and `execute` permissions for any files owned by that group. SUSE Linux Enterprise Desktop reserves the following user and group IDs:

- `0`: reserved for `root` user
- `1-999`: reserved for system users
- `1000-60000`: reserved for regular users
- `60001-60513`: reserved for `systemd homed`
- `60514-60577`: reserved for `systemd nspawn` user mapping
- `60578-61183`: unused
- `61184-65519` : reserved for `systemd dynamic` service users
- `65520-65533` : unused
- `65534-65534` : reserved for `nobody` user
- `65535-65535` : reserved for `16 bit-1`
- `65536-524287` : unused
- `524288-1879048191` : reserved for `systemd nspawn` containers
- `1879048192-2147483647` : unused
- `2147483648-4294967295` : do not use

For more information on UIDs and GIDs, see:

- https://systemd.io/UIDS-GIDS/
- https://kanidm.github.io/kanidm/stable/accounts/posix_accounts_and_groups.html#gid-number-generation

Important

A new IPA deployment can use the following UID ranges:

- `65536-524287` : unused
- `1879048192-2147483647` : unused

7 YaST online update

Revision History

2024-05-13

SUSE offers a continuous stream of software security updates for your product. By default, the update applet is used to keep your system up to date. Refer to *Section 8.5, "The GNOME package updater"* for further information on the update applet. This chapter covers the alternative tool for updating software packages: YaST Online Update.

The current patches for SUSE® Linux Enterprise Desktop are available from an update software repository. If you have registered your product during the installation, an update repository is already configured. If you have not registered SUSE Linux Enterprise Desktop, you can do so by starting the *Product Registration* in YaST. Alternatively, you can manually add an update repository from a source you trust. To add or remove repositories, start the Repository Manager with *Software* › *Software Repositories* in YaST. Learn more about the Repository Manager in *Section 8.4, "Managing software repositories and services"*.

Note: Error on accessing the update catalog

If you are not able to access the update catalog, this may happen because of an expired subscription. Normally, SUSE Linux Enterprise Desktop comes with a one-year or three-year subscription, during which you have access to the update catalog. This access will be denied after the subscription ends.

If an access to the update catalog is denied, you can see a warning message prompting you to visit the SUSE Customer Center and check your subscription. The SUSE Customer Center is available at https://scc.suse.com//.

Note: Firewall settings for receiving updates

If your system is behind a firewall that blocks outgoing traffic, make sure to allow connections to `https://scc.suse.com/` and `https://updates.suse.com` on ports 80 and 443 in order to receive updates. For more information, such as IP addresses and proxy server configuration, refer to https://www.suse.com/support/kb/doc/?id=000021034.

SUSE provides updates with different relevance levels:

Security updates

Fix severe security hazards and should always be installed.

Recommended updates

Fix issues that could compromise your computer.

Optional updates

Fix non-security relevant issues or provide enhancements.

7.1 The online update dialog

To open the YaST *Online Update* dialog, start YaST and select *Software* › *Online Update*. Alternatively, start it from the command line with **`yast2 online_update`**.

The *Online Update* window consists of four sections.

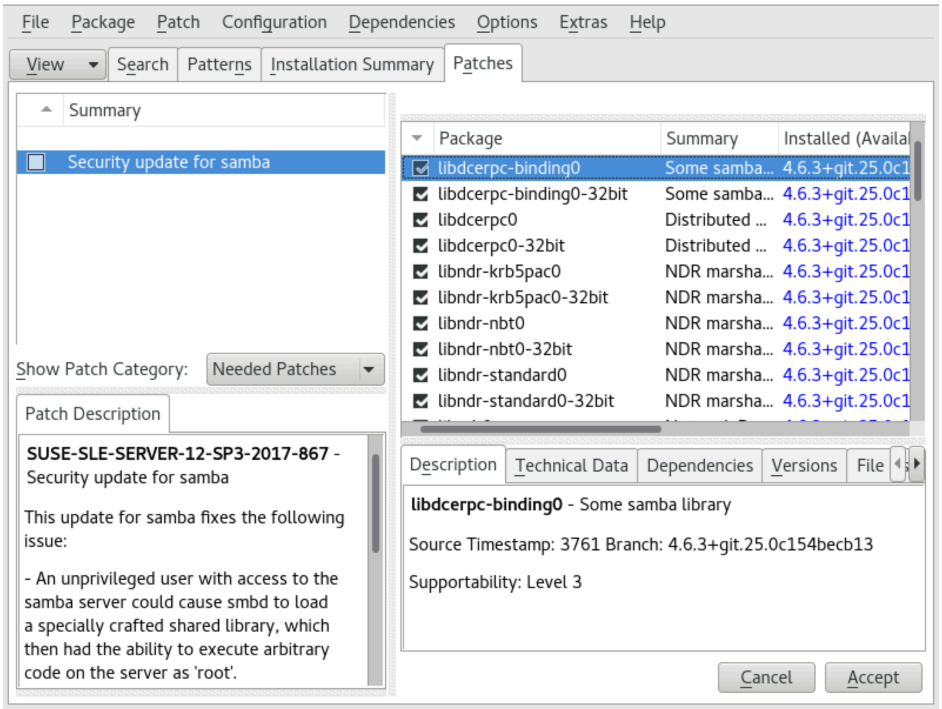


FIGURE 7.1: YAST ONLINE UPDATE

The *Summary* section on the left lists the available patches for SUSE Linux Enterprise Desktop. The patches are sorted by security relevance: `security`, `recommended`, and `optional`. You can change the view of the *Summary* section by selecting one of the following options from *Show Patch Category*:

Needed patches (default view)
: Non-installed patches that apply to packages installed on your system.

Unneeded patches
: Patches that either apply to packages not installed on your system, or patches that have requirements which have already have been fulfilled (because the relevant packages have already been updated from another source).

All patches
: All patches available for SUSE Linux Enterprise Desktop.

Each list entry in the *Summary* section consists of a symbol and the patch name. For an overview of the possible symbols and their meaning, press `Shift`–`F1`. Actions required by `Security` and `Recommended` patches are automatically preset. These actions are *Autoinstall*, *Autoupdate* and *Autodelete*.

If you install an up-to-date package from a repository other than the update repository, the requirements of a patch for this package may be fulfilled with this installation. In this case, a check mark is displayed in front of the patch summary. The patch is visible in the list until you mark it for installation. This does not install the patch (because the package already is up to date), but mark the patch as having been installed.

Select an entry in the *Summary* section to view a short *Patch Description* at the bottom left corner of the dialog. The upper right section lists the packages included in the selected patch (a patch can consist of several packages). Click an entry in the upper right section to view details about the respective package that is included in the patch.

7.2 Installing patches

The YaST Online Update dialog allows you to either install all available patches at once or manually select the desired patches. You may also revert patches that have been applied to the system.

By default, all new patches (except `optional` ones) that are currently available for your system are already marked for installation. They will be applied automatically once you click *Accept* or *Apply*. If one or multiple patches require a system reboot, you will be notified about this before the patch installation starts. You can then either decide to continue with the installation of the selected patches, skip the installation of all patches that need rebooting and install the rest, or go back to the manual patch selection.

PROCEDURE 7.1: APPLYING PATCHES WITH YAST ONLINE UPDATE

1. Start YaST and select *Software* › *Online Update*.
2. To automatically apply all new patches (except `optional` ones) that are currently available for your system, click *Apply* or *Accept*.
3. First modify the selection of patches that you want to apply:
   a. Use the respective filters and views that the interface provides. For details, refer to *Section 7.1, "The online update dialog"*.
   b. Select or deselect patches according to your needs and wishes by right-clicking the patch and choosing the respective action from the context menu.

      Important: Always apply security updates

      Do not deselect any `security`-related patches without a good reason. These patches fix severe security hazards and prevent your system from being exploited.

   c. Most patches include updates for several packages. To change actions for single packages, right-click a package in the package view and choose an action.
   d. To confirm your selection and apply the selected patches, proceed with *Apply* or *Accept*.
4. After the installation is complete, click *Finish* to leave the YaST *Online Update*. Your system is now up to date.

7.3 Viewing retracted patches

Maintenance updates are carefully tested to minimize the risk of introducing a bug. If a patch proves to contain a bug, it is automatically retracted. A new update (with a higher version number) is issued to revert the buggy patch, and is blocked from being installed again. You can see retracted patches, and their history, on the *Package Classification* tab.

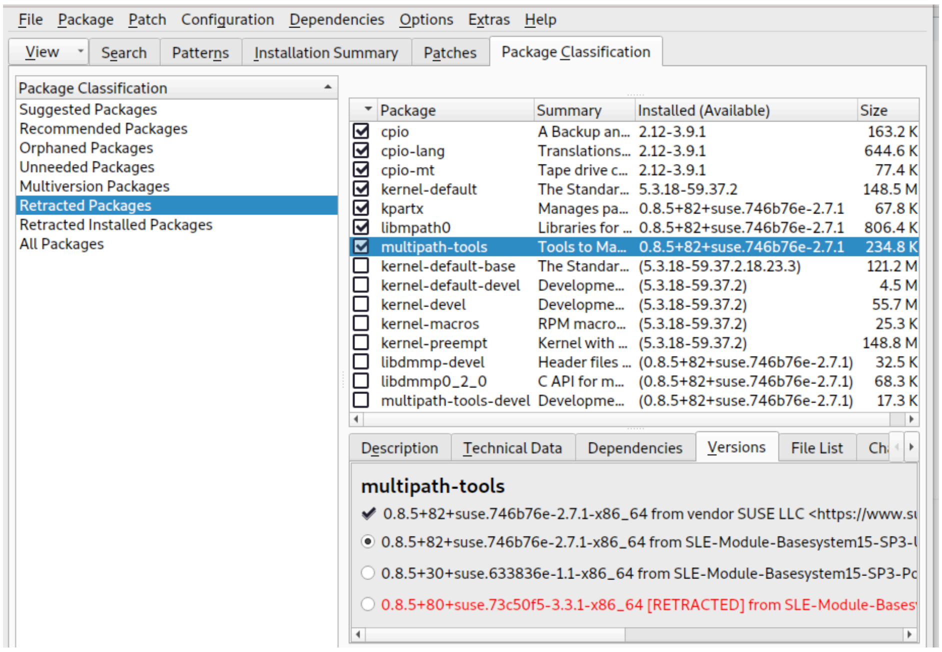


FIGURE 7.2: VIEWING RETRACTED PATCHES AND HISTORY

7.4 Automatic online update

You may configure automatic updates with a daily, weekly or monthly schedule with YaST. Install the `yast2-online-update-configuration` package.

By default, updates are downloaded as delta RPMs. Since rebuilding RPM packages from delta RPMs is a memory- and processor-intensive task, certain setups or hardware configurations may require you to disable the use of delta RPMs for the sake of performance.

Certain patches, such as kernel updates or packages requiring license agreements, require user interaction, which would cause the automatic update procedure to stop. You can configure skipping patches that require user interaction.

Use the *Patches* tab in the YaST *Software* module to review available and installed patches, including references to bug reports and CVE bulletins.

PROCEDURE 7.2: CONFIGURING THE AUTOMATIC ONLINE UPDATE

1. After installation, start YaST and select *Software* › *Online Update*. Choose *Configuration* › *Online Update*. If the yast2-online-update-configuration is not installed, you will be prompted to do that.

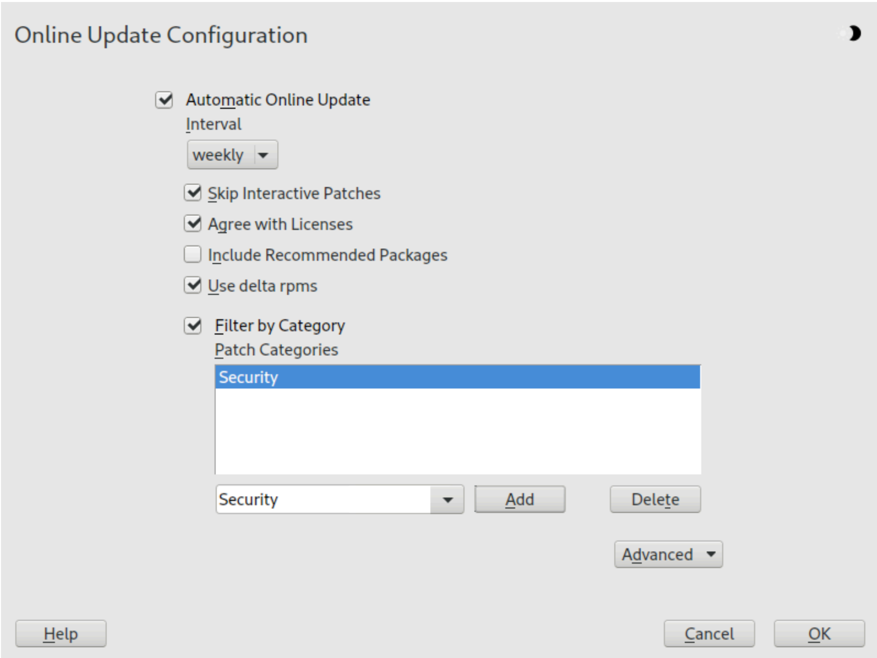


FIGURE 7.3: YAST ONLINE UPDATE CONFIGURATION

Alternatively, start the module with **yast2 online_update_configuration** from the command line.

2. Choose the update interval: *Daily*, *Weekly*, or *Monthly*.

3. Sometimes patches may require the attention of the administrator, for example when restarting critical services. For example, this might be an update for Docker Open Source Engine that requires all containers to be restarted. Before these patches are installed, the user is informed about the consequences and is asked to confirm the installation of the patch. Such patches are called “Interactive Patches”.
When installing patches automatically, it is assumed that you have accepted the installation of interactive patches. If you prefer to review these patches before they get installed, check *Skip Interactive Patches*. In this case, interactive patches will be skipped during automated patching. Make sure to periodically run a manual online update, to check whether interactive patches are waiting to be installed.

4. To automatically accept any license agreements, activate *Agree with Licenses*.

5. To automatically install all packages recommended by updated packages, activate *Include Recommended Packages*.
6. To disable the use of delta RPMs (for performance reasons), un-check *Use Delta RPMs*.
7. To filter the patches by category (such as security or recommended), check *Filter by Category* and add the appropriate patch categories from the list. Only patches of the selected categories will be installed. It is a good practice to enable only automatic *Security* updates, and to manually review all others. Patching is normally reliable, but you may wish to test non-security patches, and roll them back if you encounter any problems.
    - *Packagemanager and YaST* supply patches for package management and YaST features and modules.
    - *Security* patches provide crucial updates and bugfixes.
    - *Recommended* patches are optional bugfixes and enhancements.
    - *Optional* are new packages.
    - *Other* is equivalent to miscellaneous.
    - *Document* is unused.
8. Confirm your configuration by clicking *OK*.

The automatic online update does not automatically restart the system afterward. If there are package updates that require a system reboot, you need to do this manually.

8 Installing or removing software

Revision History

2024-06-21

Using YaST's software management module, you can search for software packages as well as install and remove them. When installing packages, YaST automatically resolves all dependencies. To install packages that are not on the installation medium, you can add software repositories and YaST to manage them. You can also keep your system up to date by managing software updates using the update applet.

The YaST Software Manager makes it possible to manage software sources on your system. There are two versions of this YaST module: a graphical version for X Window and a text-based version to use with the command line. The graphical flavor is described below—for details on the text-based YaST, see *Chapter 4, YaST in text mode*.

Note: Confirmation and review of changes

When installing, updating or removing packages, any changes in the Software Manager are only applied after clicking *Accept* or *Apply*. YaST maintains a list with all actions, allowing you to review and modify your changes before applying them to the system.

8.1 Definition of terms

The following terms are important for understanding installing and removing software in SUSE Linux Enterprise Desktop.

Repository

A local or remote directory containing packages, plus additional information about these packages (package metadata).

(Repository) alias/repository name

A short name for a repository (called `Alias` within Zypper and *Repository Name* within YaST). It can be chosen by the user when adding a repository and must be unique.

Repository description files

Each repository provides files describing content of the repository (package names, versions, etc.). These repository description files are downloaded to a local cache that is used by YaST.

Product

Represents a whole product, for example, SUSE® Linux Enterprise Desktop.

Pattern

A pattern is an installable group of packages dedicated to a certain purpose. For example, the `Laptop` pattern contains all packages that are needed in a mobile computing environment. Patterns define package dependencies (such as required or recommended packages) and come with a preselection of packages marked for installation. This ensures that the most important packages needed for a certain purpose are available on your system after installation of the pattern. If necessary, you can manually select or deselect packages within a pattern.

Package

A package is a compressed file in `rpm` format that contains the files for a particular program.

Patch

A fix or compilation of fixes released by SUSE to correct operation defects (program bugs) in SUSE products. A patch can contain one or multiple files to replace or enhance existing executables, programs, applications or documents.

A patch consists of one or more packages and may be applied by delta RPMs. It may also introduce dependencies to packages that are not installed yet.

Resolvable

A generic term for product, pattern, package or patch. The most commonly used type of resolvable is a package or a patch.

Delta RPM

A delta RPM consists only of the binary diff between two defined versions of a package, and therefore has the smallest download size. Before being installed, the full RPM package is rebuilt on the local machine.

Package dependencies

Certain packages are dependent on other packages, such as shared libraries. In other terms, a package may `require` other packages—if the required packages are not available, the package cannot be installed. Besides dependencies (package requirements) that must be fulfilled, certain packages `recommend` other packages. These recommended packages are only installed if they are available, otherwise they are ignored and the package recommending them is installed nevertheless.

8.2 Registering an installed system

If you skip registration during installation, or you want to re-register your system, you can register the system at any time. Use the YaST module *Product Registration* or the command line tool **`SUSEConnect`**.

8.2.1 Registering with YaST

To register the system, start YaST and switch to *Software*, then *Product Registration*.

By default the system is registered with the SUSE Customer Center. If your organization provides local registration servers, you can either choose one from the list of auto-detected servers or provide the URL manually.

8.2.2 Registering with SUSEConnect

To register from the command line, use the command

```
> sudo SUSEConnect -r REGISTRATION_CODE -e EMAIL_ADDRESS
```

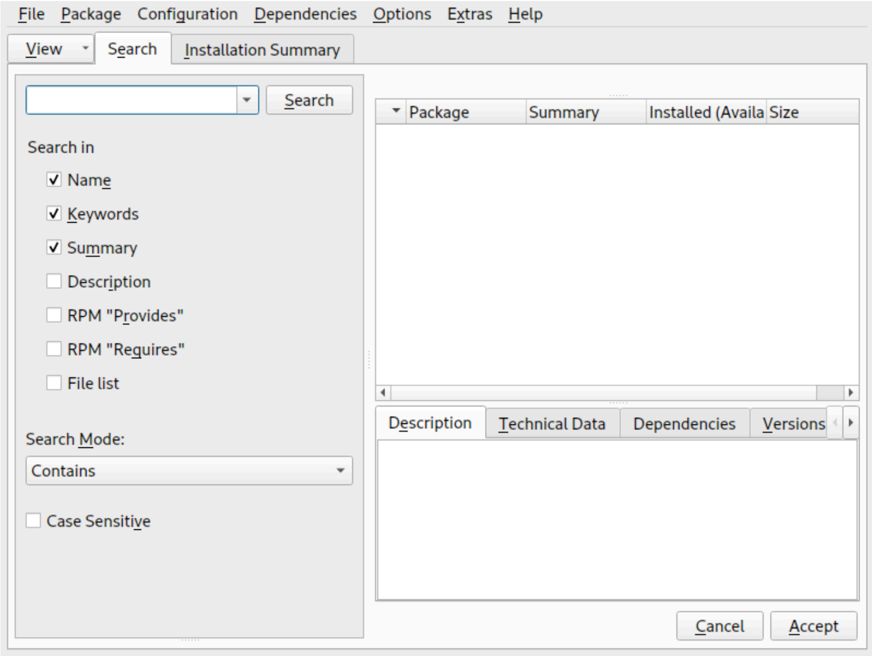
Replace *`REGISTRATION_CODE`* with the registration code you received with your copy of SUSE Linux Enterprise Desktop. Replace *`EMAIL_ADDRESS`* with the e-mail address associated with the SUSE account you or your organization uses to manage subscriptions.

To register with a local registration server, also provide the URL to the server:

```
> sudo SUSEConnect -r REGISTRATION_CODE -e EMAIL_ADDRESS --url "URL"
```

8.3 Using the YaST software manager

Start the software manager from the *YaST Control Center* by choosing *Software* › *Software Management*.



8.3.1 Searching software

The YaST software manager can install packages or patterns from all currently enabled repositories. It offers different views and filters to make it easier to find the software you are searching for. The *Search* view is the default view of the window. To change view, click *View* and select one of the following entries from the drop-down box. The selected view opens in a new tab.

VIEWS FOR SEARCHING PACKAGES OR PATTERNS

Patterns

Lists all patterns available for installation on your system.

Package Groups

Lists all packages sorted by groups such as *Graphics*, *Programming*, or *Security*.

Languages

A filter to list all packages needed to add a new system language.

Repositories

A filter to list packages by repository. To select more than one repository, hold the `Ctrl` key while clicking repository names. The “pseudo repository” *@System* lists all packages currently installed.

Services

Shows which packages belong to a certain module or extension. Select an entry (for example, `Basesystem` or `High Availability`) to display a list of packages that belong to this module or extension.

Search

Lets you search for a package according to certain criteria. Enter a search term and press `Enter`. Refine your search by specifying where to *Search In* and by changing the *Search Mode*. For example, if you do not know the package name but only the name of the application that you are searching for, try including the package *Description* in the search process.

Installation Summary

If you have already selected packages for installation, update or removal, this view shows the changes that will be applied to your system when you click *Accept*. To filter for packages with a certain status in this view, activate or deactivate the respective check boxes. Press `Shift`–`F1` for details on the status flags.

Tip: Finding packages not belonging to an active repository

To list all packages that do not belong to an active repository, choose *View* › *Repositories* › *@System* and then choose *Secondary Filter* › *Unmaintained Packages*. This is useful, for example, if you have deleted a repository and want to make sure no packages from that repository remain installed.

Tip: Searching software online

The online search feature allows searching for packages across all registered and unregistered modules and extensions.

PROCEDURE 8.1: SEARCHING SOFTWARE ONLINE

To search for software packages online, perform the following steps:

1. Open the online search window with *Extras* › *Search Online*.
2. Enter a *Package Name* and press `Enter` or click *Search*. YaST contacts the SUSE Customer Center and shows the results in a table, including the module or extension of each package. Select a package to see additional details.

3. Select one or more packages for installation by clicking the corresponding table row and *Toggle Selection*. Alternatively, you can double-click a row. If the package belongs to an unregistered module or extension, YaST asks for confirmation to register it.
4. Click *Next*, review the changes, and install the packages.

8.3.2 Installing and removing packages or patterns

Certain packages are dependent on other packages, such as shared libraries. Several packages cannot coexist with others on the system. If possible, YaST automatically resolves these dependencies or conflicts. If your choice results in a dependency conflict that cannot be automatically solved, you need to solve it manually as described in *Section 8.3.4, "Package dependencies"*.

Note: Removal of packages

When removing any packages, by default YaST only removes the selected packages. If you want YaST to also remove any other packages that become unneeded after removal of the specified package, select *Options* › *Cleanup when deleting packages* from the main menu.

1. Search for packages as described in *Section 8.3.1, "Searching software"*.
2. The packages found are listed in the right pane. To install a package or remove it, right-click it and choose *Install* or *Delete*. If the relevant option is not available, check the package status indicated by the symbol in front of the package name—press `Shift`–`F1` for help.

   #### Tip: Applying an action to all packages listed

   To apply an action to all packages listed in the right pane, go to the main menu and choose an action from *Package* › *All in This List*.

3. To install a pattern, right-click the pattern name and choose *Install*.
4. It is not possible to remove a pattern. Instead, select the packages for the pattern you want to remove and mark them for removal.
5. To select more packages, repeat the steps mentioned above.

6. Before applying your changes, you can review or modify them by clicking *View* › *Installation Summary*. By default, all packages that will change status are listed.

7. To revert the status for a package, right-click the package and select one of the following entries: *Keep* if the package was scheduled to be deleted or updated, or *Do Not Install* if it was scheduled for installation. To abandon all changes and quit the Software Manager, click *Cancel* and *Abandon*.

8. When you are finished, click *Accept* to apply your changes.

9. If YaST finds additional dependencies, it shows a list of related packages to install, update or remove. Click *Continue* to accept them.
   After all selected packages are installed, updated or removed, the YaST Software Manager automatically closes.

Note: Installing source packages

Installing source packages with YaST Software Manager is not possible at the moment. Use the command line tool **zypper** for this purpose. For more information, see *Section 9.1.3.5, "Installing or downloading source packages"*.

8.3.3 Updating packages

Instead of updating individual packages, you can also update all installed packages or all packages from a certain repository. When mass updating packages, the following aspects are generally considered:

- priorities of the repositories that provide the package,
- architecture of the package (for example, AMD64/Intel 64),
- version number of the package,
- package vendor.

Which of the aspects has the highest importance for choosing the update candidates depends on the respective update option you choose.

1. To update all installed packages to the latest version, choose *Package* › *All Packages* › *Update if Newer Version Available* from the main menu.

All repositories are checked for possible update candidates, using the following policy: YaST first tries to restrict the search to packages with the same architecture and vendor as the installed one. If the search is positive, the “best” update candidate from those is selected according to the process below. However, if no comparable package of the same vendor can be found, the search is expanded to all packages with the same architecture. If still no comparable package can be found, all packages are considered and the “best” update candidate is selected according to the following criteria:

1. Repository priority: prefer the package from the repository with the highest priority.
2. If more than one package results from this selection, choose the one with the “best” architecture (best choice: matching the architecture of the installed one).

If the resulting package has a higher version number than the installed one, the installed package is updated and replaced with the selected update candidate.
This option tries to avoid changes in architecture and vendor for the installed packages, but under certain circumstances, they are tolerated.

Note: Update unconditionally

If you choose *Package* › *All Packages* › *Update Unconditionally* instead, the same criteria apply but any candidate package found is installed unconditionally. Thus, choosing this option may lead to downgrading certain packages.

2. To make sure that the packages for a mass update derive from a certain repository:
   a. Choose the repository from which to update as described in *Section 8.3.1, “Searching software”* .
   b. On the right hand side of the window, click *Switch system packages to the versions in this repository*. This explicitly allows YaST to change the package vendor when replacing the packages.
      When you proceed with *Accept*, all installed packages are replaced by packages deriving from this repository, if available. This may lead to changes in vendor and architecture and even to downgrading certain packages.
   c. To refrain from this, click *Cancel switching system packages to the versions in this repository*. You can only cancel this until you click the *Accept* button.

3. Before applying your changes, you can review or modify them by clicking *View* › *Installation Summary*. By default, all packages that will change status, are listed.

4. If all options are set according to your wishes, confirm your changes with *Accept* to start the mass update.

8.3.4 Package dependencies

Most packages are dependent on other packages. If a package, for example, uses a shared library, it is dependent on the package providing this library. Certain packages cannot coexist, causing a conflict (for example, you can only install one mail transfer agent: sendmail or postfix). When installing or removing software, the Software Manager makes sure no dependencies or conflicts remain unsolved to ensure system integrity.

In case there exists only one solution to resolve a dependency or a conflict, it is resolved automatically. Multiple solutions always cause a conflict which needs to be resolved manually. If solving a conflict involves a vendor or architecture change, it also needs to be solved manually. When clicking *Accept* to apply any changes in the Software Manager, you get an overview of all actions triggered by the automatic resolver which you need to confirm.

By default, dependencies are automatically checked. A check is performed every time you change a package status (for example, by marking a package for installation or removal). This is generally useful, but can become exhausting when manually resolving a dependency conflict. To disable this function, go to the main menu and deactivate *Dependencies* › *Autocheck*. Manually perform a dependency check with *Dependencies* › *Check Now*. A consistency check is always performed when you confirm your selection with *Accept*.

To review a package's dependencies, right-click it and choose *Show Solver Information*. A map showing the dependencies opens. Packages that are already installed are displayed in a green frame.

Note: Manually solving package conflicts

Unless you are experienced, follow the suggestions YaST makes when handling package conflicts, otherwise you may not be able to resolve them. Keep in mind that every change you make potentially triggers other conflicts, so you can easily end up with a steadily increasing number of conflicts. In case this happens, *Cancel* the Software Manager, *Abandon* all your changes and start again.

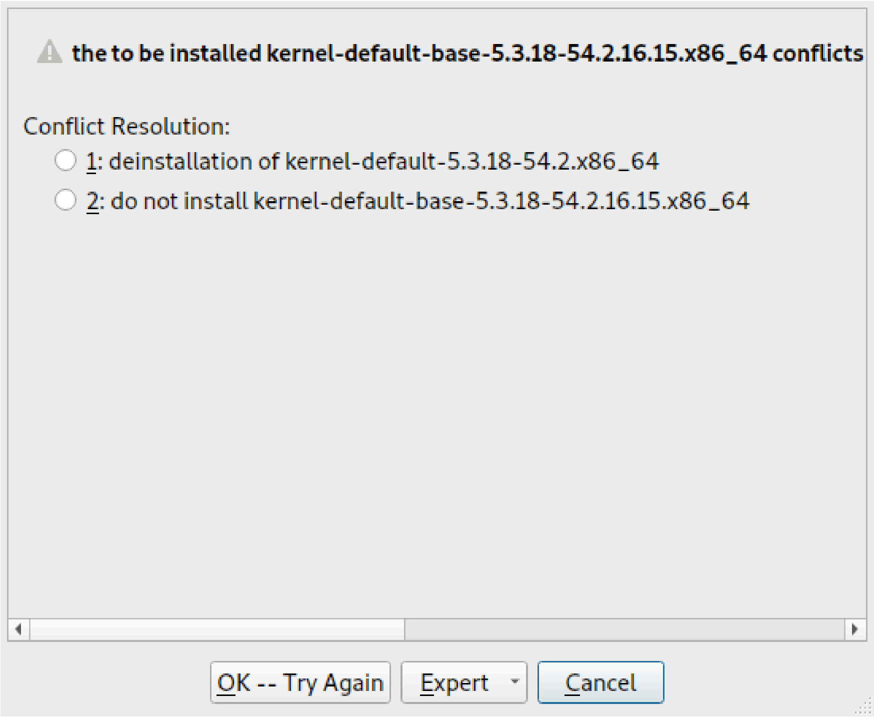


FIGURE 8.1: CONFLICT MANAGEMENT OF THE SOFTWARE MANAGER

8.3.5 Handling package recommendations

In addition to the hard dependencies required to run a program (for example a certain library), a package can also have weak dependencies, which add for example extra functionality or translations. These weak dependencies are called package recommendations.

When installing a new package, recommended packages are still installed by default. When updating an existing package, missing recommendations are not be installed automatically. To change this, set `PKGMGR_RECOMMENDED="yes"` in `/etc/sysconfig/yast2`. To install all missing recommendations for already installed packages, start *YaST* › *Software Manager* and choose *Extras* › *Install All Matching Recommended Packages*.

To disable the installation of recommended packages when installing new packages, deactivate *Dependencies* › *Install Recommended Packages* in the YaST Software Manager. When using the command-line tool Zypper to install packages, use the option `--no-recommends`.

8.4 Managing software repositories and services

To install third-party software, add software repositories to your system. By default, product repositories such as SUSE Linux Enterprise Desktop-DVD 15 SP7 and a matching update repository are automatically configured when you register your system. For more information about registration, see *Book "Deployment Guide", Chapter 5 "Installation steps", Section 5.6 "Registration"* or *Book "Upgrade Guide", Chapter 4 "Upgrading offline", Section 4.7 "Registering your system"*. Depending on the initially selected product, an additional repository containing translations, dictionaries, etc. might also be configured.

To manage repositories, start YaST and select *Software* › *Software Repositories*. The *Configured Software Repositories* dialog opens. Here, you can also manage subscriptions to *Services* by changing the *View* at the right corner of the dialog to *All Services*. A Service in this context is a *Repository Index Service* (RIS) that can offer one or more software repositories. Such a Service can be changed dynamically by its administrator or vendor.

Each repository provides files describing repository content (package names, versions, etc.). YaST downloads these repository description files to a local cache. To ensure their integrity, software repositories can be signed with the GPG Key of the repository maintainer. Whenever you add a new repository, YaST offers the ability to import its key.

Warning: Trusting external software sources

Before adding external software repositories to your list of repositories, make sure this repository can be trusted. SUSE is not responsible for any problems arising from software installed from third-party software repositories.

8.4.1 Adding software repositories

You can either add repositories from DVD/CD, a USB flash drive, a local directory, an ISO image, or a network source.

To add repositories from the *Configured Software Repositories* dialog in YaST proceed as follows:

1. Click *Add*.

2. Select one of the options listed in the dialog:

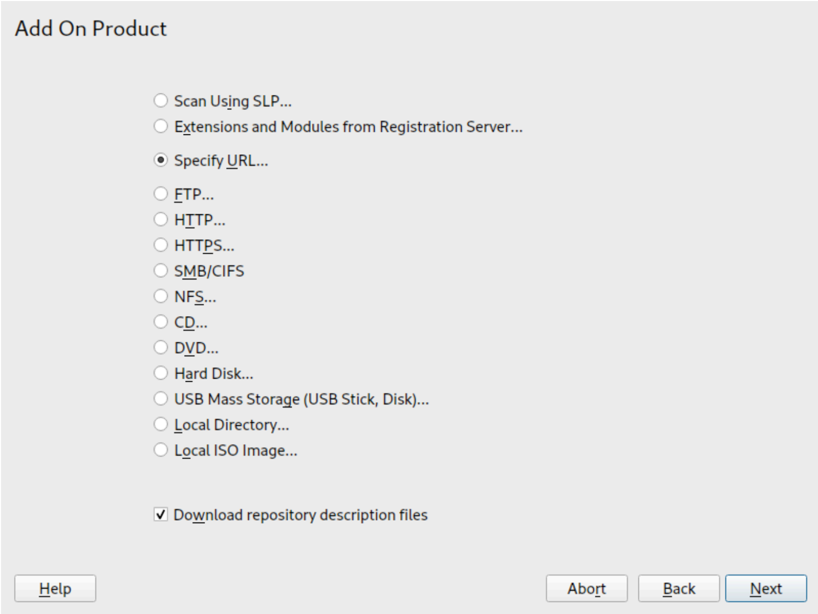


FIGURE 8.2: ADDING A SOFTWARE REPOSITORY

- To scan your network for installation servers announcing their services via SLP, select *Scan Using SLP* and click *Next*.
- To add a repository from a removable medium, choose the relevant option and insert the medium or connect the USB device to the machine, respectively. Click *Next* to start the installation.
- For the majority of repositories, you will be asked to specify the path (or URL) to the media after selecting the respective option and clicking *Next*. Specifying a *Repository Name* is optional. If none is specified, YaST will use the product name or the URL as repository name.

The option *Download Repository Description Files* is activated by default. If you deactivate the option, YaST will automatically download the files later, if needed.

3. Depending on the repository you add, you may be prompted to import the repository's GPG key or asked to agree to a license.
After confirming, YaST will download and parse the metadata. It will add the repository to the list of *Configured Repositories*.

4. If needed, adjust the repository *Properties* as described in *Section 8.4.2, “Managing repository properties”*.

5. Confirm your changes with *OK* to close the configuration dialog.
6. After having successfully added the repository, the software manager starts and you can install packages from this repository. For details, refer to *Chapter 8, Installing or removing software*.

8.4.2 Managing repository properties

The *Configured Software Repositories* overview of the *Software Repositories* lets you change the following repository properties:

Status

The repository status can either be *Enabled* or *Disabled*. You can only install packages from repositories that are enabled. To turn a repository off temporarily, select it and deactivate *Enable*. You can also double-click a repository name to toggle its status. To remove a repository completely, click *Delete*.

Refresh

When refreshing a repository, its content description (package names, versions, etc.) is downloaded to a local cache that is used by YaST. It is sufficient to do this once for static repositories such as CDs or DVDs, whereas repositories whose content changes often should be refreshed frequently. The easiest way to keep a repository's cache up to date is to choose *Automatically Refresh*. To do a manual refresh click *Refresh* and select one of the options.

Keep Downloaded Packages

Packages from remote repositories are downloaded before being installed. By default, they are deleted upon successful installation. Activating *Keep Downloaded Packages* prevents the deletion of downloaded packages. The download location is configured in `/etc/zypp/zypp.conf`, by default it is `/var/cache/zypp/packages`.

Priority

The *Priority* of a repository is a value between `1` and `200`, with `1` being the highest priority and `200` the lowest priority. Any new repositories that are added with YaST get a priority of `99` by default. If you do not care about a priority value for a certain repository, you can also set the value to `0` to apply the default priority to that repository (`99`). If a package is available in more than one repository, then the repository with the highest priority takes precedence. This is useful to avoid downloading packages unnecessarily from the Internet by giving a local repository (for example, a DVD) a higher priority.

Important: Priority compared to version

The repository with the highest priority takes precedence in any case. Therefore, make sure that the update repository always has the highest priority, otherwise you might install an outdated version that will not be updated until the next online update.

Name and URL

To change a repository name or its URL, select it from the list with a single-click and then click *Edit*.

8.4.3 Managing repository keys

To ensure their integrity, software repositories can be signed with the GPG Key of the repository maintainer. Whenever you add a new repository, YaST offers to import its key. Verify it as you would do with any other GPG key and make sure it does not change. If you detect a key change, something might be wrong with the repository. Disable the repository as an installation source until you know the cause of the key change.

To manage all imported keys, click *GPG Keys* in the *Configured Software Repositories* dialog. Select an entry with the mouse to show the key properties at the bottom of the window. *Add*, *Edit*, or *Delete* keys with a click on the respective buttons.

8.5 The GNOME package updater

SUSE offers a continuous stream of software security patches and updates for your product. They can be installed using tools available with your desktop or by running the *YaST online update* module. This section describes how to update the system from the GNOME desktop using the *Package Updater*.

Contrary to the YaST Online Update module, the GNOME *Package Updater* not only offers to install patches from the update repositories, but also new versions of packages that are already installed. (Patches fix security issues or malfunctions; the functionality and version number is usually not changed. New versions of a package increase the version number and add functionality or introduce major changes.)

Whenever new patches or package updates are available, GNOME shows a notification in the notification area or on the lock screen.

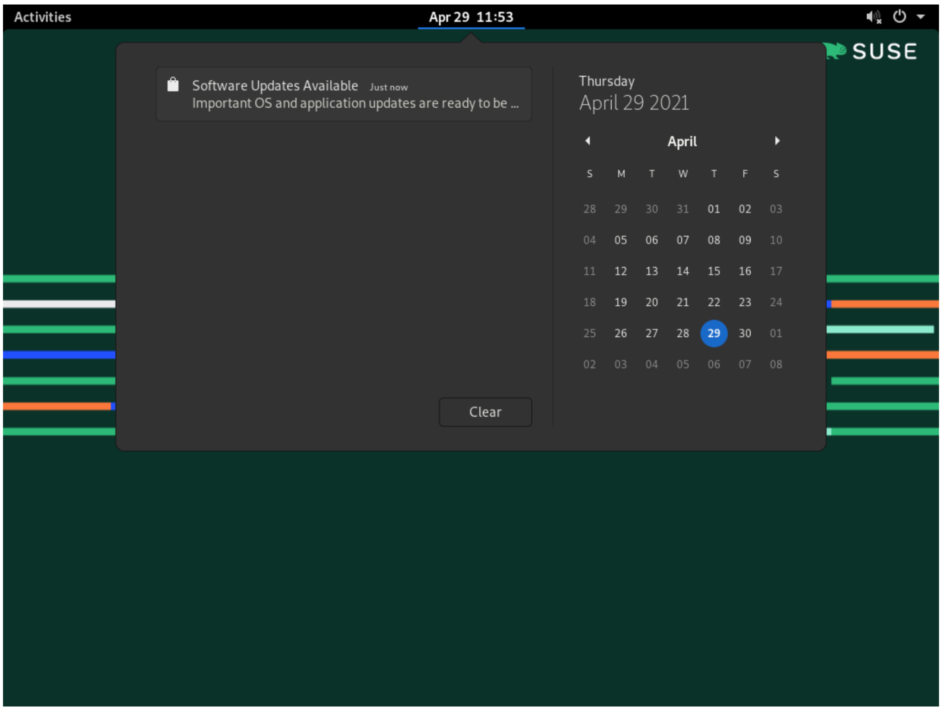
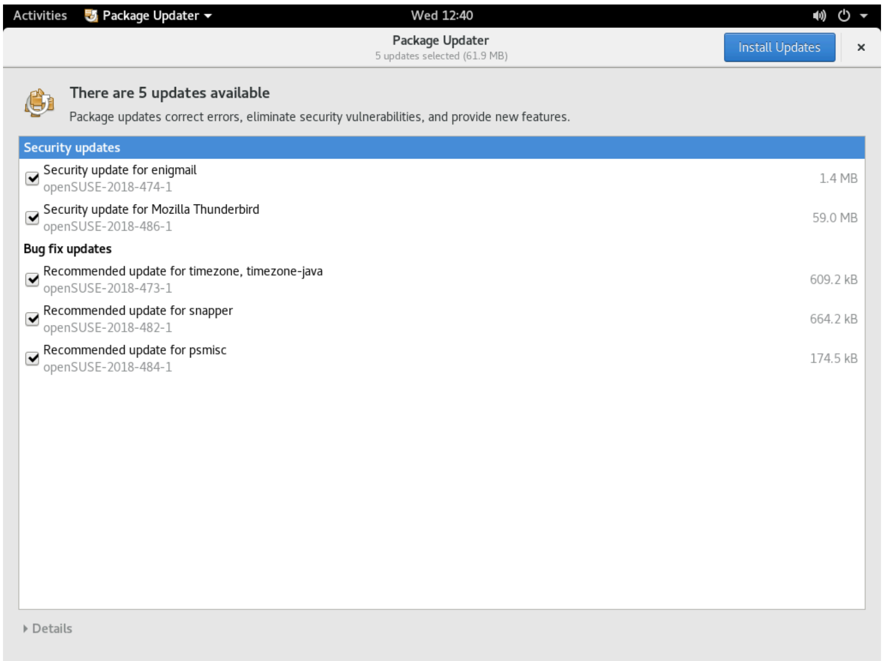


FIGURE 8.3: UPDATE NOTIFICATION ON GNOME DESKTOP

To configure the notification settings for the *Package Updater*, start GNOME *Settings* and choose *Notifications* › *Package Updater*.

PROCEDURE 8.2: INSTALLING PATCHES AND UPDATES WITH THE GNOME PACKAGE UPDATER

1. To install the patches and updates, click the notification message. This opens the GNOME *Package Updater*. Alternatively, open the updater from *Activities* by typing `package U` and choosing *Package Updater*.



2. Updates are sorted into four categories:

Security updates (patches)
: Fix severe security hazards and should always be installed.

Recommended updates (patches)
: Fix issues that could compromise your computer. Installing them is strongly recommended.

Optional updates (patches)
: Fix non-security relevant issues or provide enhancements.

Other updates
: New versions of packages that are installed.

   All available updates are preselected for installation. If you do not want to install all updates, deselect unwanted updates first. It is strongly recommended to always install all security and recommended updates.
   To get detailed information on an update, click its title and then *Details*. The information is displayed in a box beneath the package list.

3. Click *Install Updates* to start the installation.

4. Some updates may require to restart the machine or to log out. Check the message displayed after installation for instructions.

8.6 Updating packages with *GNOME Software*

In addition to the GNOME *Package Updater*, GNOME provides *GNOME Software* which has the following functionality:

- Install, update, and remove software delivered as an RPM via PackageKit
- Install, update, and remove software delivered as a Flatpak
- Install, update, and remove GNOME shell extensions (https://extensions.gnome.org)
- Update firmware for hardware devices using *Linux Vendor Firmware Service* (LVFS, https://fwupd.org)

GNOME Software also provides screenshots, ratings, and reviews for software.

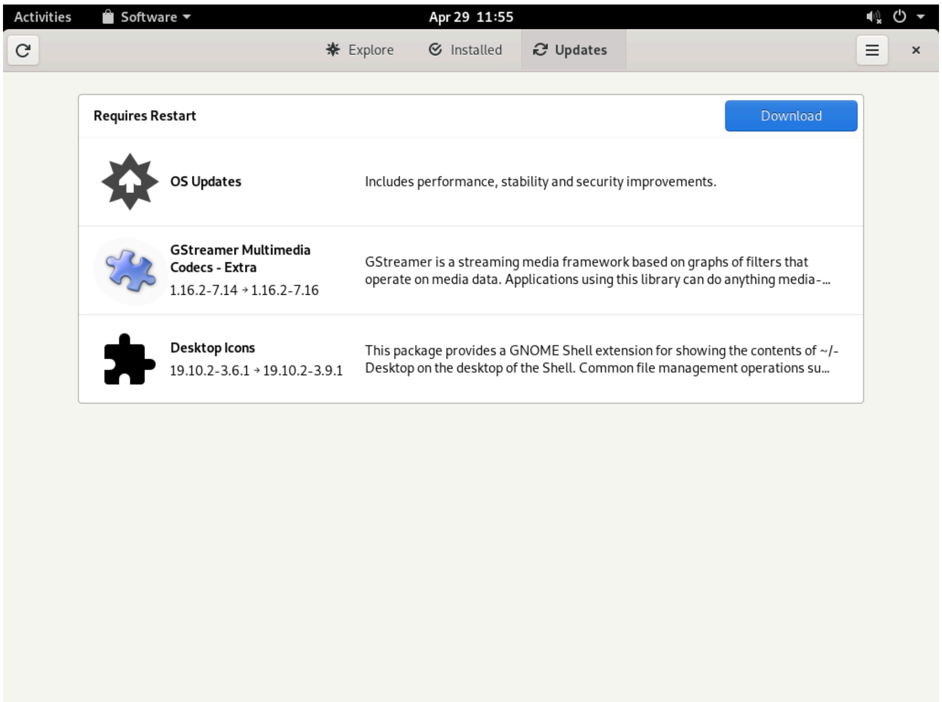


FIGURE 8.4: *GNOME SOFTWARE—UPDATES* VIEW

GNOME Software has the following differences to other tools provided on SUSE Linux Enterprise Desktop:

- Unlike YaST or Zypper, for installing software packaged as an RPM, *GNOME Software* is restricted to software that provides AppStream metadata. This includes most desktop applications.
- While the GNOME *Package Updater* updates packages within the running system (forcing you to restart the respective applications), *GNOME Software* downloads the updates and applies them after reboot.

9 Managing software with command line tools

Revision History

2022-02-11

This chapter describes Zypper and RPM, two command line tools for managing software. For a definition of the terminology used in this context (for example, `repository`, `patch`, or `update`) refer to *Section 8.1, “Definition of terms”*.

9.1 Using Zypper

Zypper is a command line package manager for installing, updating, and removing packages. It also manages repositories. It is especially useful for accomplishing remote software management tasks or managing software from shell scripts.

9.1.1 General usage

The general syntax of Zypper is:

```
zypper [--global-options] COMMAND  [--command-options] [arguments]
```

The components enclosed in brackets are not required. See **`zypper help`** for a list of general options and all commands. To get help for a specific command, type **`zypper help`** *`COMMAND`*.

Zypper commands

The simplest way to execute Zypper is to type its name, followed by a command. For example, to apply all needed patches to the system, use:

```
> sudo zypper patch
```

Global options

Additionally, you can choose from one or more global options by typing them immediately before the command:

```
> sudo zypper --non-interactive patch
```

In the above example, the option `--non-interactive` means that the command is run without asking anything (automatically applying the default answers).

Command-specific options

To use options that are specific to a particular command, type them immediately after the command:

```
> sudo zypper patch --auto-agree-with-licenses
```

In the above example, `--auto-agree-with-licenses` is used to apply all needed patches to a system without you being asked to confirm any licenses. Instead, licenses will be accepted automatically.

Arguments

Some commands require one or more arguments. For example, when using the command **`install`**, you need to specify which package or which packages you want to *install*:

```
> sudo zypper install mplayer
```

Some options also require a single argument. The following command will list all known patterns:

```
> zypper search -t pattern
```

You can combine all of the above. For example, the following command will install the `mc` and `vim` packages from the `factory` repository while being verbose:

```
> sudo zypper -v install --from factory mc vim
```

The `--from` option keeps all repositories enabled (for solving any dependencies) while requesting the package from the specified repository. `--repo` is an alias for `--from`, and you may use either one.

Most Zypper commands have a `dry-run` option that does a simulation of the given command. It can be used for test purposes.

```
> sudo zypper remove --dry-run MozillaFirefox
```

Zypper supports the global `--userdata STRING` option. You can specify a string with this option, which gets written to Zypper's log files and plug-ins (such as the Btrfs plug-in). It can be used to mark and identify transactions in log files.

```
> sudo zypper --userdata STRING patch
```

9.1.2 Using Zypper subcommands

Zypper subcommands are executables that are stored in the directory specified by the `zypper_execdir` configuration option. It is `/usr/lib/zypper/commands` by default. If a subcommand is not found there, Zypper automatically searches the rest of your $PATH locations for it. This lets you create your own local extensions and store them in user space.

Executing subcommands in the Zypper shell, and using global Zypper options are not supported.

List your available subcommands:

```
> zypper help subcommand
[...]
Available zypper subcommands in '/usr/lib/zypper/commands'

  appstream-cache
  lifecycle
  migration
  search-packages

Zypper subcommands available from elsewhere on your $PATH

  log                    Zypper logfile reader
                             (/usr/sbin/zypper-log)
```

View the help screen for a subcommand:

```
> zypper help appstream-cache
```

9.1.3 Installing and removing software with Zypper

To install or remove packages, use the following commands:

```
> sudo zypper install PACKAGE_NAME
> sudo zypper remove PACKAGE_NAME
```

Warning: Do not remove mandatory system packages

Do not remove mandatory system packages like `glibc` , `zypper` , `kernel` . If they are removed, the system can become unstable or stop working altogether.

9.1.3.1 Selecting which packages to install or remove

There are various ways to address packages with the commands **`zypper install`** and **`zypper remove`**.

By exact package name

```
> sudo zypper install MozillaFirefox
```

By exact package name and version number

```
> sudo zypper install MozillaFirefox-52.2
```

By repository alias and package name

```
> sudo zypper install mozilla:MozillaFirefox
```

Where `mozilla` is the alias of the repository from which to install.

By package name using wild cards

You can select all packages that have names starting or ending with a certain string. Use wild cards with care, especially when removing packages. The following command will install all packages starting with “Moz”:

```
> sudo zypper install 'Moz*'
```

> **Tip: Removing all `-debuginfo` packages**
>
> When debugging a problem, you sometimes need to temporarily install a lot of `-debuginfo` packages which give you more information about running processes. After your debugging session finishes and you need to clean the environment, run the following:
>
> ```
> > sudo zypper remove '*-debuginfo'
> ```

By capability

For example, to install a package without knowing its name, capabilities come in handy. The following command will install the package `MozillaFirefox`:

```
> sudo zypper install firefox
```

By capability, hardware architecture, or version

Together with a capability, you can specify a hardware architecture and a version:

- The name of the desired hardware architecture is appended to the capability after a full stop. For example, to specify the AMD64/Intel 64 architectures (which in Zypper is named `x86_64`), use:

  ```
  > sudo zypper install 'firefox.x86_64'
  ```

- Versions must be appended to the end of the string and must be preceded by an operator: `<` (lesser than), `<=` (lesser than or equal), `=` (equal), `>=` (greater than or equal), `>` (greater than).

  ```
  > sudo zypper install 'firefox>=74.2'
  ```

- You can also combine a hardware architecture and version requirement:

  ```
  > sudo zypper install 'firefox.x86_64>=74.2'
  ```

By path to the RPM file

You can also specify a local or remote path to a package:

```
> sudo zypper install /tmp/install/MozillaFirefox.rpm
> sudo zypper install http://download.example.com/MozillaFirefox.rpm
```

9.1.3.2 Combining installation and removal of packages

To install and remove packages simultaneously, use the `+/-` modifiers. To install `emacs` and simultaneously remove `vim` , use:

```
> sudo zypper install emacs -vim
```

To remove `emacs` and simultaneously install `vim` , use:

```
> sudo zypper remove emacs +vim
```

To prevent the package name starting with the `-` being interpreted as a command option, always use it as the second argument. If this is not possible, precede it with `--`:

```
> sudo zypper install -emacs +vim       # Wrong
> sudo zypper install vim -emacs        # Correct
> sudo zypper install -- -emacs +vim    # Correct
> sudo zypper remove emacs +vim         # Correct
```

9.1.3.3 Cleaning up dependencies of removed packages

If (together with a certain package), you automatically want to remove any packages that become unneeded after removing the specified package, use the `--clean-deps` option:

```
> sudo zypper rm --clean-deps PACKAGE_NAME
```

9.1.3.4 Using Zypper in scripts

By default, Zypper asks for a confirmation before installing or removing a selected package, or when a problem occurs. You can override this behavior using the `--non-interactive` option. This option must be given before the actual command (**`install`**, **`remove`**, and **`patch`**), as can be seen in the following:

```
> sudo zypper --non-interactive install PACKAGE_NAME
```

This option allows the use of Zypper in scripts and cron jobs.

9.1.3.5 Installing or downloading source packages

To install the corresponding source package of a package, use:

```
> zypper source-install PACKAGE_NAME
```

When executed as `root`, the default location to install source packages is `/usr/src/packages/` and `~/rpmbuild` when run as user. These values can be changed in your local **`rpm`** configuration.

This command will also install the build dependencies of the specified package. If you do not want this, add the switch `-D`:

```
> sudo zypper source-install -D PACKAGE_NAME
```

To install only the build dependencies use `-d`.

```
> sudo zypper source-install -d PACKAGE_NAME
```

Of course, this will only work if you have the repository with the source packages enabled in your repository list (it is added by default, but not enabled). See *Section 9.1.6, "Managing repositories with Zypper"* for details on repository management.

A list of all source packages available in your repositories can be obtained with:

```
> zypper search -t srcpackage
```

You can also download source packages for all installed packages to a local directory. To download source packages, use:

```
> zypper source-download
```

The default download directory is `/var/cache/zypper/source-download`. You can change it using the `--directory` option. To only show missing or extraneous packages without downloading or deleting anything, use the `--status` option. To delete extraneous source packages, use the `--delete` option. To disable deleting, use the `--no-delete` option.

9.1.3.6 Installing packages from disabled repositories

Normally you can only install or refresh packages from enabled repositories. The `--plus-content TAG` option helps you specify repositories to be refreshed, temporarily enabled during the current Zypper session, and disabled after it completes.

For example, to enable repositories that may provide additional `-debuginfo` or `-debugsource` packages, use `--plus-content debug`. You can specify this option multiple times.

To temporarily enable such 'debug' repositories to install a specific `-debuginfo` package, use the option as follows:

```
> sudo zypper --plus-content debug \
   install "debuginfo(build-id)=eb844a5c20c70a59fc693cd1061f851fb7d046f4"
```

The `build-id` string is reported by **`gdb`** for missing debuginfo packages.

> **Note: Disabled installation media**
>
> Repositories from the SUSE Linux Enterprise Desktop installation media are still configured but disabled after successful installation. You can use the `--plus-content` option to install packages from the installation media instead of the online repositories. Before calling **`zypper`**, ensure the media is available, for example by inserting the DVD into the computer's drive.

9.1.3.7 Utilities

To verify whether all dependencies are still fulfilled and to repair missing dependencies, use:

```
> zypper verify
```

In addition to dependencies that must be fulfilled, some packages “recommend” other packages. These recommended packages are only installed if actually available and installable. In case recommended packages were made available after the recommending package has been installed (by adding additional packages or hardware), use the following command:

```
> sudo zypper install-new-recommends
```

This command is very useful after plugging in a Web cam or Wi-Fi device. It will install drivers for the device and related software, if available. Drivers and related software are only installable if certain hardware dependencies are fulfilled.

9.1.4 Updating software with Zypper

There are three different ways to update software using Zypper: by installing patches, by installing a new version of a package or by updating the entire distribution. The latter is achieved with **`zypper dist-upgrade`**. Upgrading SUSE Linux Enterprise Desktop is discussed in *Book “Upgrade Guide”, Chapter 2 “Upgrade paths and methods”*.

9.1.4.1 Installing all needed patches

Patching SUSE Linux Enterprise Desktop is the most reliable way to install new versions of installed packages. It guarantees that all required packages with correct versions are installed and ensures that package versions considered as *conflicting* are omitted.

To install all officially released patches that apply to your system, run:

```
> sudo zypper patch
```

All patches available from repositories configured on your computer are checked for their relevance to your installation. If they are relevant (and not classified as `optional` or `feature`), they are installed immediately. If **`zypper patch`** succeeds, it is guaranteed that no vulnerable version package is installed unless you confirm the exception. Note that the official update repository is only available after registering your SUSE Linux Enterprise Desktop installation.

If a patch that is about to be installed includes changes that require a system reboot, you will be warned before.

The plain **`zypper patch`** command does not apply patches from third party repositories. To update also the third party repositories, use the `with-update` command option as follows:

```
> sudo zypper patch --with-update
```

To install also optional patches, use:

```
> sudo zypper patch --with-optional
```

To install all patches relating to a specific Bugzilla issue, use:

```
> sudo zypper patch --bugzilla=NUMBER
```

To install all patches relating to a specific CVE database entry, use:

```
> sudo zypper patch --cve=NUMBER
```

For example, to install a security patch with the CVE number `CVE-2010-2713`, execute:

```
> sudo zypper patch --cve=CVE-2010-2713
```

To install only patches which affect Zypper and the package management itself, use:

```
> sudo zypper patch --updatestack-only
```

Bear in mind that other command options that would also update other repositories will be dropped if you use the `updatestack-only` command option.

9.1.4.2 Listing patches

To find out whether patches are available, Zypper allows viewing the following information:

Number of needed patches

To list the number of needed patches (patches that apply to your system but are not yet installed), use **`patch-check`**:

```
> zypper patch-check
Loading repository data...
Reading installed packages...
5 patches needed (1 security patch)
```

This command can be combined with the `--updatestack-only` option to list only the patches which affect Zypper and the package management itself.

List of needed patches

To list all needed patches (patches that apply to your system but are not yet installed), use **`zypper list-patches`**.

List of all patches

To list all patches available for SUSE Linux Enterprise Desktop, regardless of whether they are already installed or apply to your installation, use **`zypper patches`**.

It is also possible to list and install patches relevant to specific issues. To list specific patches, use the **zypper list-patches** command with the following options:

By Bugzilla issues

To list all needed patches that relate to Bugzilla issues, use the option `--bugzilla`.
To list patches for a specific bug, you can also specify a bug number: `--bugzilla=NUMBER`.
To search for patches relating to multiple Bugzilla issues, add commas between the bug numbers, for example:

```
> zypper list-patches --bugzilla=972197,956917
```

By CVE number

To list all needed patches that relate to an entry in the CVE database (Common Vulnerabilities and Exposures), use the option `--cve`.
To list patches for a specific CVE database entry, you can also specify a CVE number: `--cve=NUMBER`. To search for patches relating to multiple CVE database entries, add commas between the CVE numbers, for example:

```
> zypper list-patches --cve=CVE-2016-2315,CVE-2016-2324
```

List retracted patches

In the SUSE Linux Enterprise 15 codestream, some patches are automatically retracted. Maintenance updates are carefully tested, because there is a risk that an update contains a new bug. If an update proves to contain a bug, a new update (with a higher version number) is issued to revert the buggy update, and the buggy update is blocked from being installed again. You can list retracted patches with **zypper**:

```
> zypper lp --all |grep retracted
SLE-Module-Basesystem15-SP3-Updates | SUSE-SLE-Module-Basesystem-15-SP3-2021-1965
 | recommended | important | ---    | retracted  | Recommended update for multipath-
tools
SLE-Module-Basesystem15-SP3-Updates | SUSE-SLE-Module-Basesystem-15-SP3-2021-2689
 | security    | important | ---    | retracted  | Security update for cpio
SLE-Module-Basesystem15-SP3-Updates | SUSE-SLE-Module-Basesystem-15-SP3-2021-3655
 | security    | important | reboot | retracted  | Security update for the Linux
 Kernel
```

See complete information on a retracted (or any) patch:

```
> zypper patch-info SUSE-SLE-Product-SLES-15-2021-2689
Loading repository data...
Reading installed packages...
```

```
Information for patch SUSE-SLE-Product-SLES-15-2021-2689:
--------------------------------------------------------
Repository  : SLE-Product-SLES15-LTSS-Updates
Name        : SUSE-SLE-Product-SLES-15-2021-2689
Version     : 1
Arch        : noarch
Vendor      : maint-coord@suse.de
Status      : retracted
Category    : security
Severity    : important
Created On  : Mon 16 Aug 2021 03:44:00 AM PDT
Interactive : ---
Summary     : Security update for cpio
Description :
    This update for cpio fixes the following issues:

    It was possible to trigger Remote code execution due to a integer overflow
    (CVE-2021-38185, bsc#1189206)

    UPDATE:
    This update was buggy and could lead to hangs, so it has been retracted.
    There will be a follow up update.
    [...]
```

Patch with conflicting packages

```
Information for patch openSUSE-SLE-15.3-2022-333:
------------------------------------------------
Repository  : Update repository with updates from SUSE Linux Enterprise 15
Name        : openSUSE-SLE-15.3-2022-333
Version     : 1
Arch        : noarch
Vendor      : maint-coord@suse.de
Status      : needed
Category    : security
Severity    : important
Created On  : Fri Feb  4 09:30:32 2022
Interactive : reboot
Summary     : Security update for xen
Description :
    This update for xen fixes the following issues:

    - CVE-2022-23033: Fixed guest_physmap_remove_page not removing the p2m mappings.
 (XSA-393) (bsc#1194576)
    - CVE-2022-23034: Fixed possible DoS by a PV guest Xen while unmapping a grant.
 (XSA-394) (bsc#1194581)
    - CVE-2022-23035: Fixed insufficient cleanup of passed-through device IRQs.
 (XSA-395) (bsc#1194588)
```

```
Provides    : patch:openSUSE-SLE-15.3-2022-333 = 1
Conflicts   : [22]
    xen.src < 4.14.3_06-150300.3.18.2
    xen.noarch < 4.14.3_06-150300.3.18.2
    xen.x86_64 < 4.14.3_06-150300.3.18.2
    xen-devel.x86_64 < 4.14.3_06-150300.3.18.2
    xen-devel.noarch < 4.14.3_06-150300.3.18.2
[...]
```

The above patch conflicts with the affected or vulnerable versions of 22 packages. If any of these affected or vulnerable packages are installed, it triggers a conflict, and the patch is classified as *needed*. **`zypper patch`** tries to install all available patches. If it encounters problems, it reports them, thus informing you that not all updates are installed. The conflict can be resolved by either updating the affected or vulnerable packages or by removing them. Because SUSE update repositories also ship fixed packages, updating is a standard way to resolve conflicts. If the package cannot be updated—for example, because of dependency issues or package locks—it is deleted after the user's approval.

To list all patches regardless of whether they are needed, use the option `--all` additionally. For example, to list all patches with a CVE number assigned, use:

```
> zypper list-patches --all --cve
Issue | No.            | Patch             | Category    | Severity  | Status
------+----------------+-------------------+-------------+-----------+----------
cve   | CVE-2019-0287 | SUSE-SLE-Module.. | recommended | moderate  | needed
cve   | CVE-2019-3566 | SUSE-SLE-SERVER.. | recommended | moderate  | not needed
[...]
```

9.1.4.3 Installing new package versions

If a repository contains only new packages, but does not provide patches, **`zypper patch`** does not show any effect. To update all installed packages with newer available versions, use the following command:

```
> sudo zypper update
```

Important

`zypper update` ignores problematic packages. For example, if a package is locked, **`zypper update`** omits the package, even if a higher version of it is available. Conversely, **`zypper patch`** reports a conflict if the package is considered vulnerable.

To update individual packages, specify the package with either the update or install command:

```
> sudo zypper update PACKAGE_NAME
> sudo zypper install PACKAGE_NAME
```

A list of all new installable packages can be obtained with the command:

```
> zypper list-updates
```

Note that this command only lists packages that match the following criteria:

- has the same vendor like the already installed package,
- is provided by repositories with at least the same priority than the already installed package,
- is installable (all dependencies are satisfied).

A list of *all* new available packages (regardless whether installable or not) can be obtained with:

```
> sudo zypper list-updates --all
```

To find out why a new package cannot be installed, use the **`zypper install`** or **`zypper update`** command as described above.

9.1.4.4 Identifying orphaned packages

Whenever you remove a repository from Zypper or upgrade your system, some packages can get in an “orphaned” state. These *orphaned* packages belong to no active repository anymore. The following command gives you a list of these:

```
> sudo zypper packages --orphaned
```

With this list, you can decide if a package is still needed or can be removed safely.

9.1.5 Identifying processes and services using deleted files

When patching, updating, or removing packages, there may be running processes on the system which continue to use files having been deleted by the update or removal. Use **`zypper ps`** to list processes using deleted files. In case the process belongs to a known service, the service name is listed, making it easy to restart the service. By default **`zypper ps`** shows a table:

```
> zypper ps
```

```
PID   | PPID | UID | User  | Command      | Service      | Files
------+------+-----+-------+--------------+--------------+-------------------
814   | 1    | 481 | avahi | avahi-daemon | avahi-daemon | /lib64/ld-2.19.s->
      |      |     |       |              |              | /lib64/libdl-2.1->
      |      |     |       |              |              | /lib64/libpthrea->
      |      |     |       |              |              | /lib64/libc-2.19->
[...]
```

PID: ID of the process

PPID: ID of the parent process

UID: ID of the user running the process

Login: Login name of the user running the process

Command: Command used to execute the process

Service: Service name (only if command is associated with a system service)

Files: The list of the deleted files

The output format of **`zypper ps`** can be controlled as follows:

`zypper ps``-s`

Create a short table not showing the deleted files.

```
> zypper ps -s
PID   | PPID | UID  | User    | Command      | Service
------+------+------+---------+--------------+--------------
814   | 1    | 481  | avahi   | avahi-daemon | avahi-daemon
817   | 1    | 0    | root    | irqbalance   | irqbalance
1567  | 1    | 0    | root    | sshd         | sshd
1761  | 1    | 0    | root    | master       | postfix
1764  | 1761 | 51   | postfix | pickup       | postfix
1765  | 1761 | 51   | postfix | qmgr         | postfix
2031  | 2027 | 1000 | tux     | bash         |
```

`zypper ps``-ss`

Show only processes associated with a system service.

```
PID   | PPID | UID  | User    | Command      | Service
------+------+------+---------+--------------+--------------
814   | 1    | 481  | avahi   | avahi-daemon | avahi-daemon
817   | 1    | 0    | root    | irqbalance   | irqbalance
1567  | 1    | 0    | root    | sshd         | sshd
1761  | 1    | 0    | root    | master       | postfix
1764  | 1761 | 51   | postfix | pickup       | postfix
1765  | 1761 | 51   | postfix | qmgr         | postfix
```

`zypper ps``-sss`

Only show system services using deleted files.

```
avahi-daemon
irqbalance
postfix
sshd
```

zypper ps--print "systemctl status %s"

Show the commands to retrieve status information for services which might need a restart.

```
systemctl status avahi-daemon
systemctl status irqbalance
systemctl status postfix
systemctl status sshd
```

For more information about service handling refer to *Chapter 19, The* `systemd` *daemon*.

9.1.6 Managing repositories with Zypper

All installation or patch commands of Zypper rely on a list of known repositories. To list all repositories known to the system, use the command:

```
> zypper repos
```

The result will look similar to the following output:

EXAMPLE 9.1: ZYPPER—LIST OF KNOWN REPOSITORIES

```
> zypper repos
# | Alias        | Name          | Enabled | Refresh
--+--------------+---------------+---------+--------
1 | SLEHA-15-GEO | SLEHA-15-GEO  | Yes     | No
2 | SLEHA-15     | SLEHA-15      | Yes     | No
3 | SLES15       | SLES15        | Yes     | No
```

When specifying repositories in various commands, an alias, URI or repository number from the **zypper repos** command output can be used. A repository alias is a short version of the repository name for use in repository handling commands. Note that the repository numbers can change after modifying the list of repositories. The alias will never change by itself.

By default, details such as the URI or the priority of the repository are not displayed. Use the following command to list all details:

```
> zypper repos -d
```

9.1.6.1 Adding repositories

To add a repository, run

```
> sudo zypper addrepo URI ALIAS
```

URI can either be an Internet repository, a network resource, a directory or a CD or DVD (see https://en.opensuse.org/openSUSE:Libzypp_URIs for details). The *ALIAS* is a shorthand and unique identifier of the repository. You can freely choose it, with the only exception that it needs to be unique. Zypper will issue a warning if you specify an alias that is already in use.

9.1.6.2 Refreshing repositories

zypper enables you to fetch changes in packages from configured repositories. To fetch the changes, run:

```
> sudo zypper refresh
```

> **Note: Default behavior of zypper**
>
> By default, some commands perform **refresh** automatically, so you do not need to run the command explicitly.

The **refresh** command enables you to view changes also in disabled repositories, by using the `--plus-content` option:

```
> sudo zypper --plus-content refresh
```

This option fetches changes in repositories, but keeps the disabled repositories in the same state —disabled.

9.1.6.3 Removing repositories

To remove a repository from the list, use the command **zypper removerepo** together with the alias or number of the repository you want to delete. For example, to remove the repository `SLEHA-12-GEO` from *Example 9.1, “Zypper—list of known repositories”*, use one of the following commands:

```
> sudo zypper removerepo 1
> sudo zypper removerepo "SLEHA-12-GEO"
```

9.1.6.4 Modifying repositories

Enable or disable repositories with **`zypper modifyrepo`**. You can also alter the repository's properties (such as refreshing behavior, name or priority) with this command. The following command will enable the repository named `updates`, turn on auto-refresh and set its priority to 20:

```
> sudo zypper modifyrepo -er -p 20 'updates'
```

Modifying repositories is not limited to a single repository—you can also operate on groups:

`-a`: all repositories
`-l`: local repositories
`-t`: remote repositories
`-m TYPE`: repositories of a certain type (where *TYPE* can be one of the following: `http`, `https`, `ftp`, `cd`, `dvd`, `dir`, `file`, `cifs`, `smb`, `nfs`, `hd`, `iso`)

To rename a repository alias, use the `renamerepo` command. The following example changes the alias from `Mozilla Firefox` to `firefox`:

```
> sudo zypper renamerepo 'Mozilla Firefox' firefox
```

9.1.7 Querying repositories and packages with Zypper

Zypper offers various methods to query repositories or packages. To get lists of all products, patterns, packages or patches available, use the following commands:

```
> zypper products
> zypper patterns
> zypper packages
> zypper patches
```

To query all repositories for certain packages, use `search`. To get information regarding particular packages, use the `info` command.

9.1.7.1 Searching for software

The **`zypper search`** command works on package names, or, optionally, on package summaries and descriptions. Strings wrapped in `/` are interpreted as regular expressions. By default, the search is not case-sensitive.

Simple search for a package name containing `fire`

```
> zypper search "fire"
```

Simple search for the exact package `MozillaFirefox`

```
> zypper search --match-exact "MozillaFirefox"
```

Also search in package descriptions and summaries

```
> zypper search -d fire
```

Only display packages not already installed

```
> zypper search -u fire
```

Display packages containing the string `fir` not followed be `e`

```
> zypper se "/fir[^e]/"
```

9.1.7.2 Searching for packages across all SLE modules

To search for packages both within and outside of currently enabled SLE modules, use the **`search-packages`** subcommand. This command contacts the SUSE Customer Center and searches all modules for matching packages, for example:

```
> zypper search-packages package1 package2
```

`zypper search-packages` provides the following options:

- Search for an exact match of your search string: `-x`, `--match-exact`
- Group the results by module (default: group by package): `-g`, `--group-by-module`
- Display more detailed information about packages: `-d`, `--details`
- Output search results in XML: `--xmlout`

9.1.7.3 Searching for specific capability

To search for packages which provide a special capability, use the command `what-provides`. For example, if you want to know which package provides the Perl module `SVN::Core`, use the following command:

```
> zypper what-provides 'perl(SVN::Core)'
```

The `what-provides` *`PACKAGE_NAME`* is similar to **`rpm -q --whatprovides`** *`PACKAGE_NAME`*, but RPM is only able to query the RPM database (that is the database of all installed packages). Zypper, on the other hand, will tell you about providers of the capability from any repository, not only those that are installed.

9.1.7.4 Showing package information

To query single packages, use **`info`** with an exact package name as an argument. This displays detailed information about a package. In case the package name does not match any package name from repositories, the command outputs detailed information for non-package matches. If you request a specific type (by using the `-t` option) and the type does not exist, the command outputs other available matches but without detailed information.

If you specify a source package, the command displays binary packages built from the source package. If you specify a binary package, the command outputs the source packages used to build the binary package.

To also show what is required/recommended by the package, use the options `--requires` and `--recommends`:

```
> zypper info --requires MozillaFirefox
```

9.1.8 Showing lifecycle information

SUSE products are generally supported for 10 years. Often, you can extend that standard lifecycle by using the extended support offerings of SUSE which add three years of support. Depending on your product, find the exact support lifecycle at https://www.suse.com/lifecycle ↗.

To check the lifecycle of your product and the supported package, use the **`zypper lifecycle`** command as shown below:

```
# zypper lifecycle
    Product end of support
Codestream: SUSE Linux Enterprise Server 15          2028-07-31
    Product: SUSE Linux Enterprise Server 15 SP3     n/a*

Module end of support
Basesystem Module                                    n/a*
Desktop Applications Module                          n/a*
Server Applications Module                           n/a*
```

```
Package end of support if different from product:
autofs                                  Now, installed 5.1.3-7.3.1, update available
 5.1.3-7.6.1
```

9.1.9 Configuring Zypper

Zypper now comes with a configuration file, allowing you to permanently change Zypper's behavior (either system-wide or user-specific). For system-wide changes, edit `/etc/zypp/zypper.conf`. For user-specific changes, edit `~/.zypper.conf`. If `~/.zypper.conf` does not yet exist, you can use `/etc/zypp/zypper.conf` as a template: copy it to `~/.zypper.conf` and adjust it to your liking. Refer to the comments in the file for help about the available options.

9.1.10 Troubleshooting

If you have trouble accessing packages from configured repositories (for example, Zypper cannot find a certain package even though you know it exists in one of the repositories), refreshing the repositories may help:

```
> sudo zypper refresh
```

If that does not help, try

```
> sudo zypper refresh -fdb
```

This forces a complete refresh and rebuild of the database, including a forced download of raw metadata.

9.1.11 Zypper rollback feature on Btrfs file system

If the Btrfs file system is used on the root partition and **`snapper`** is installed, Zypper automatically calls **`snapper`** when committing changes to the file system to create appropriate file system snapshots. These snapshots can be used to revert any changes made by Zypper. See *Chapter 10, System recovery and snapshot management with Snapper* for more information.

9.1.12 More information

For more information on managing software from the command line, enter **zypper help**, **zypper help** *COMMAND* or refer to the **zypper(8)** man page. For a complete and detailed command reference, cheat sheets with the most important commands, and information on how to use Zypper in scripts and applications, refer to https://en.opensuse.org/SDB:Zypper_usage. A list of software changes for the latest SUSE Linux Enterprise Desktop version can be found at https://en.opensuse.org/openSUSE:Zypper_versions.

9.2 RPM—the package manager

RPM (RPM Package Manager) is used for managing software packages. Its main commands are **rpm** and **rpmbuild**. The powerful RPM database can be queried by the users, system administrators and package builders for detailed information about the installed software.

rpm has five modes: installing, uninstalling (or updating) software packages, rebuilding the RPM database, querying RPM bases or individual RPM archives, integrity checking of packages and signing packages. **rpmbuild** can be used to build installable packages from pristine sources.

Installable RPM archives are packed in a special binary format. These archives consist of the program files to install and certain meta information used during the installation by **rpm** to configure the software package or stored in the RPM database for documentation purposes. RPM archives normally have the extension `.rpm`.

Tip: Software development packages

For several packages, the components needed for software development (libraries, headers, include files, etc.) have been put into separate packages. These development packages are only needed if you want to compile software yourself (for example, the most recent GNOME packages). They can be identified by the name extension `-devel`, such as the packages `alsa-devel` and `gimp-devel`.

9.2.1 Verifying package authenticity

RPM packages have a GPG signature. To verify the signature of an RPM package, use the command **`rpm --checksig`** *`PACKAGE`*-1.2.3.rpm to determine whether the package originates from SUSE or from another trustworthy facility. This is especially recommended for update packages from the Internet.

While fixing issues in the operating system, you might need to install a Problem Temporary Fix (PTF) into a production system. The packages provided by SUSE are signed against a special PTF key. However, in contrast to SUSE Linux Enterprise 11, this key is not imported by default on SUSE Linux Enterprise 12 systems. To manually import the key, use the following command:

```
> sudo rpm --import \
/usr/share/doc/packages/suse-build-key/suse_ptf_key.asc
```

After importing the key, you can install PTF packages on your system.

9.2.2 Managing packages: install, update, and uninstall

Normally, the installation of an RPM archive is quite simple: **`rpm -i`** *`PACKAGE`*.rpm. With this command the package is installed, but only if its dependencies are fulfilled and if there are no conflicts with other packages. With an error message, **`rpm`** requests those packages that need to be installed to meet dependency requirements. In the background, the RPM database ensures that no conflicts arise—a specific file can only belong to one package. By choosing different options, you can force **`rpm`** to ignore these defaults, but this is only for experts. Otherwise, you risk compromising the integrity of the system and possibly jeopardize the ability to update the system.

The options `-U` or `--upgrade` and `-F` or `--freshen` can be used to update a package (for example, **`rpm -F`** *`PACKAGE`*.rpm). This command removes the files of the old version and immediately installs the new files. The difference between the two versions is that `-U` installs packages that previously did not exist in the system, while `-F` merely updates previously installed packages. When updating, **`rpm`** updates configuration files carefully using the following strategy:

- If a configuration file was not changed by the system administrator, **`rpm`** installs the new version of the appropriate file. No action by the system administrator is required.
- If a configuration file was changed by the system administrator before the update, **`rpm`** saves the changed file with the extension `.rpmorig` or `.rpmsave` (backup file) and installs the version from the new package. This is done only if the originally installed file and

the newer version are different. If this is the case, compare the backup file (`.rpmorig` or `.rpmsave`) with the newly installed file and make your changes again in the new file. Afterward, delete all `.rpmorig` and `.rpmsave` files to avoid problems with future updates.

- `.rpmnew` files appear if the configuration file already exists *and* if the `noreplace` label was specified in the `.spec` file.

Following an update, `.rpmsave` and `.rpmnew` files should be removed after comparing them, so they do not obstruct future updates. The `.rpmorig` extension is assigned if the file has not previously been recognized by the RPM database.

Otherwise, `.rpmsave` is used. In other words, `.rpmorig` results from updating from a foreign format to RPM. `.rpmsave` results from updating from an older RPM to a newer RPM. `.rpmnew` does not disclose any information to whether the system administrator has made any changes to the configuration file. A list of these files is available in `/var/adm/rpmconfigcheck`. Some configuration files (like `/etc/httpd/httpd.conf`) are not overwritten to allow continued operation.

The `-U` switch is *not* only an equivalent to uninstalling with the `-e` option and installing with the `-i` option. Use `-U` whenever possible.

To remove a package, enter **`rpm -e`** *PACKAGE*. This command only deletes the package if there are no unresolved dependencies. It is theoretically impossible to delete Tcl/Tk, for example, as long as another application requires it. Even in this case, RPM calls for assistance from the database. If such a deletion is, for whatever reason, impossible (even if *no* additional dependencies exist), it may be helpful to rebuild the RPM database using the option `--rebuilddb`.

9.2.3 Delta RPM packages

Delta RPM packages contain the difference between an old and a new version of an RPM package. Applying a delta RPM onto an old RPM results in a completely new RPM. It is not necessary to have a copy of the old RPM because a delta RPM can also work with an installed RPM. The delta RPM packages are even smaller in size than patch RPMs, which is an advantage when transferring update packages over the Internet. The drawback is that update operations with delta RPMs involved consume considerably more CPU cycles than plain or patch RPMs.

The **makedeltarpm** and **applydelta** binaries are part of the delta RPM suite (package deltarpm) and help you create and apply delta RPM packages. With the following commands, you can create a delta RPM called new.delta.rpm. The following command assumes that old.rpm and new.rpm are present:

```
> sudo makedeltarpm old.rpm new.rpm new.delta.rpm
```

Using **applydeltarpm**, you can reconstruct the new RPM from the file system if the old package is already installed:

```
> sudo applydeltarpm new.delta.rpm new.rpm
```

To derive it from the old RPM without accessing the file system, use the -r option:

```
> sudo applydeltarpm -r old.rpm new.delta.rpm new.rpm
```

See /usr/share/doc/packages/deltarpm/README for technical details.

9.2.4 RPM queries

With the -q option **rpm** initiates queries, making it possible to inspect an RPM archive (by adding the option -p) and to query the RPM database of installed packages. Several switches are available to specify the type of information required. See *Table 9.1, "Essential RPM query options"*.

TABLE 9.1: ESSENTIAL RPM QUERY OPTIONS

-i	Package information
-l	File list
-f FILE	Query the package that contains the file *FILE* (the full path must be specified with *FILE*)
-s	File list with status information (implies -l)
-d	List only documentation files (implies -l)
-c	List only configuration files (implies -l)
--dump	File list with complete details (to be used with -l, -c, or -d)

`--provides`	List features of the package that another package can request with `--requires`
`--requires`, `-R`	Capabilities the package requires
`--scripts`	Installation scripts (preinstall, postinstall, uninstall)

For example, the command **`rpm -q -i wget`** displays the information shown in *Example 9.2, “`rpm -q -i wget`”*.

EXAMPLE 9.2: **`rpm -q -i wget`**

```
Name        : wget
Version     : 1.14
Release     : 17.1
Architecture: x86_64
Install Date: Mon 30 Jan 2017 14:01:29 CET
Group       : Productivity/Networking/Web/Utilities
Size        : 2046483
License     : GPL-3.0+
Signature   : RSA/SHA256, Thu 08 Dec 2016 07:48:44 CET, Key ID 70af9e8139db7c82
Source RPM  : wget-1.14-17.1.src.rpm
Build Date  : Thu 08 Dec 2016 07:48:34 CET
Build Host  : sheep09
Relocations : (not relocatable)
Packager    : https://www.suse.com/
Vendor      : SUSE LLC <https://www.suse.com/>
URL         : http://www.gnu.org/software/wget/
Summary     : A Tool for Mirroring FTP and HTTP Servers
Description :
Wget enables you to retrieve WWW documents or FTP files from a server.
This can be done in script files or via the command line.
Distribution: SUSE Linux Enterprise 15
```

The option `-f` only works if you specify the complete file name with its full path. Provide as many file names as desired. For example:

```
> rpm -q -f /bin/rpm /usr/bin/wget
rpm-4.14.1-lp151.13.10.x86_64
wget-1.19.5-lp151.4.1.x86_64
```

If only part of the file name is known, use a shell script as shown in *Example 9.3, “Script to search for packages”*. Pass the partial file name to the script shown as a parameter when running it.

EXAMPLE 9.3: SCRIPT TO SEARCH FOR PACKAGES

```
#! /bin/sh
for i in $(rpm -q -a -l | grep $1); do
    echo "\"$i\" is in package:"
    rpm -q -f $i
    echo ""
done
```

The command **rpm -q --changelog** *PACKAGE* displays a detailed list of change information about a specific package, sorted by date.

With the installed RPM database, verification checks can be made. Initiate these with `-V`, or `--verify`. With this option, **rpm** shows all files in a package that have been changed since installation. **rpm** uses eight character symbols to give some hints about the following changes:

TABLE 9.2: RPM VERIFY OPTIONS

`5`	MD5 check sum
`S`	File size
`L`	Symbolic link
`T`	Modification time
`D`	Major and minor device numbers
`U`	Owner
`G`	Group
`M`	Mode (permissions and file type)

In the case of configuration files, the letter `c` is printed. For example, for changes to `/etc/wgetrc` (`wget` package):

```
> rpm -V wget
S.5....T c /etc/wgetrc
```

The files of the RPM database are placed in `/var/lib/rpm`. If the partition `/usr` has a size of 1 GB, this database can occupy nearly 30 MB, especially after a complete update. If the database is much larger than expected, it is useful to rebuild the database with the option `--rebuild-`

db. Before doing this, make a backup of the old database. The **cron** script **cron.daily** makes daily copies of the database (packed with gzip) and stores them in /var/adm/backup/rpmdb. The number of copies is controlled by the variable MAX_RPMDB_BACKUPS (default: 5) in /etc/sysconfig/backup. The size of a single backup is approximately 1 MB for 1 GB in /usr.

9.2.5 Installing and compiling source packages

All source packages carry a .src.rpm extension (source RPM).

Note: Installed source packages

Source packages can be copied from the installation medium to the hard disk and unpacked with YaST. They are not, however, marked as installed ([i]) in the package manager. This is because the source packages are not entered in the RPM database. Only *installed* operating system software is listed in the RPM database. When you “install” a source package, only the source code is added to the system.

The following directories must be available for **rpm** and **rpmbuild** in /usr/src/packages (unless you specified custom settings in a file like /etc/rpmrc):

SOURCES

for the original sources (.tar.bz2 or .tar.gz files, etc.) and for distribution-specific adjustments (mostly .diff or .patch files)

SPECS

for the .spec files, similar to a meta Makefile, which control the *build* process

BUILD

all the sources are unpacked, patched and compiled in this directory

RPMS

where the completed binary packages are stored

SRPMS

here are the source RPMs

When you install a source package with YaST, all the necessary components are installed in /usr/src/packages: the sources and the adjustments in SOURCES and the relevant .spec file in SPECS.

Warning: System integrity

Do not experiment with system components (`glibc`, `rpm`, etc.), because this endangers the stability of your system.

The following example uses the `wget.src.rpm` package. After installing the source package, you should have files similar to those in the following list:

```
/usr/src/packages/SOURCES/wget-1.19.5.tar.bz2
/usr/src/packages/SOURCES/wgetrc.patch
/usr/src/packages/SPECS/wget.spec
```

`rpmbuild` `-b`*X* `/usr/src/packages/SPECS/wget.spec` starts the compilation. *X* is a wild card for various stages of the build process (see the output of `--help` or the RPM documentation for details). The following is merely a brief explanation:

`-bp`
: Prepare sources in `/usr/src/packages/BUILD`: unpack and patch.

`-bc`
: Do the same as `-bp`, but with additional compilation.

`-bi`
: Do the same as `-bp`, but with additional installation of the built software. Caution: if the package does not support the BuildRoot feature, you might overwrite configuration files.

`-bb`
: Do the same as `-bi`, but with the additional creation of the binary package. If the compile was successful, the binary should be in `/usr/src/packages/RPMS`.

`-ba`
: Do the same as `-bb`, but with the additional creation of the source RPM. If the compilation was successful, the binary should be in `/usr/src/packages/SRPMS`.

`--short-circuit`
: Skip some steps.

The binary RPM created can now be installed with **`rpm`** `-i` or, preferably, with **`rpm`** `-U`. Installation with **`rpm`** makes it appear in the RPM database.

Keep in mind that the `BuildRoot` directive in the spec file is deprecated. If you still need this feature, use the `--buildroot` option as a workaround.

9.2.6 Compiling RPM packages with build

The danger with many packages is that unwanted files are added to the running system during the build process. To prevent this use `build`, which creates a defined environment in which the package is built. To establish this chroot environment, the **`build`** script must be provided with a complete package tree. This tree can be made available on the hard disk, via NFS, or from DVD. Set the position with **`build --rpms`** *`DIRECTORY`*. Unlike **`rpm`**, the **`build`** command looks for the `.spec` file in the source directory. To build `wget` (like in the above example) with the DVD mounted in the system under `/media/dvd`, use the following commands as `root`:

```
# cd /usr/src/packages/SOURCES/
# mv ../SPECS/wget.spec .
# build --rpms /media/dvd/suse/ wget.spec
```

Subsequently, a minimum environment is established at `/var/tmp/build-root`. The package is built in this environment. Upon completion, the resulting packages are located in `/var/tmp/build-root/usr/src/packages/RPMS`.

The **`build`** script offers several additional options. For example, cause the script to prefer your own RPMs, omit the initialization of the build environment or limit the **`rpm`** command to one of the above-mentioned stages. Access additional information with **`build`** `--help` and by reading the **`build`** man page.

9.2.7 Tools for RPM archives and the RPM database

Midnight Commander (**`mc`**) can display the contents of RPM archives and copy parts of them. It represents archives as virtual file systems, offering all usual menu options of Midnight Commander. Display the `HEADER` with **F3**. View the archive structure with the cursor keys and **Enter**. Copy archive components with **F5**.

A full-featured package manager is available as a YaST module. For details, see *Chapter 8, Installing or removing software*.

10 System recovery and snapshot management with Snapper

Revision History

2024-07-22

Snapper allows creating and managing file system snapshots. File system snapshots allow keeping a copy of the state of a file system at a certain point of time. The standard setup of Snapper is designed to allow rolling back system changes. However, you can also use it to create on-disk backups of user data. As the basis for this functionality, Snapper uses the Btrfs file system or thinly provisioned LVM volumes with an XFS or Ext4 file system.

Snapper has a command line interface and a YaST interface. Snapper lets you create and manage file system snapshots on the following types of file systems:

- Btrfs, a copy-on-write file system for Linux that natively supports file system snapshots of subvolumes. (Subvolumes are separately mountable file systems within a physical partition.)
  You can also boot from `Btrfs` snapshots. For more information, see *Section 10.3, “System rollback by booting from snapshots”*.
- Thinly provisioned LVM volumes formatted with XFS or Ext4.

Using Snapper, you can perform the following tasks:

- Undo system changes made by **`zypper`** and YaST. See *Section 10.2, “Using Snapper to undo changes”* for details.
- Restore files from previous snapshots. See *Section 10.2.2, “Using Snapper to restore files”* for details.
- Do a system rollback by booting from a snapshot. See *Section 10.3, “System rollback by booting from snapshots”* for details.
- Manually create and manage snapshots, within the running system. See *Section 10.6, “Manually creating and managing snapshots”* for details.

10.1 Default setup

Snapper on SUSE Linux Enterprise Desktop is set up as an undo and recovery tool for system changes. By default, the root partition (`/`) of SUSE Linux Enterprise Desktop is formatted with `Btrfs`. Taking snapshots is automatically enabled if the root partition (`/`) is big enough (more than approximately 16 GB). By default, snapshots are disabled on partitions other than `/`.

Tip: Enabling Snapper in the installed system

If you disabled Snapper during the installation, you can enable it at any time later. To do so, create a default Snapper configuration for the root file system by running:

```
> sudo snapper -c root create-config /
```

Afterward enable the different snapshot types as described in *Section 10.1.4.1, "Disabling/enabling snapshots"*.

On a Btrfs root file system, snapshots require a file system with subvolumes configured as proposed by the installer and a partition size of at least 16 GB.

When a snapshot is created, both the snapshot and the original point to the same blocks in the file system. So, initially a snapshot does not occupy additional disk space. If data in the original file system is modified, changed data blocks are copied while the old data blocks are kept for the snapshot. Therefore, a snapshot occupies the same amount of space as the data modified. So, over time, the amount of space a snapshot allocates, constantly grows. As a consequence, deleting files from a `Btrfs` file system containing snapshots may *not* free disk space.

Note: Snapshot location

Snapshots always reside on the same partition or subvolume on which the snapshot has been taken. It is not possible to store snapshots on a different partition or subvolume.

As a result, partitions containing snapshots need to be larger than partitions not containing snapshots. The exact amount depends strongly on the number of snapshots you keep and the amount of data modifications. As a rule of thumb, give partitions twice as much space as you normally would. To prevent disks from running out of space, old snapshots are automatically cleaned up. Refer to *Section 10.1.4.4, "Controlling snapshot archiving"* for details.

10.1.1 Default settings

Disks larger than 16 GB

- Configuration file: `/etc/snapper/configs/root`
- `USE_SNAPPER=yes`
- `TIMELINE_CREATE=no`

Disks smaller than 16 GB

- Configuration file: not created
- `USE_SNAPPER=no`
- `TIMELINE_CREATE=yes`

10.1.2 Types of snapshots

Although snapshots themselves do not differ in a technical sense, we distinguish between three types of snapshots, based on the events that trigger them:

Timeline snapshots

A single snapshot is created every hour. Using the YaST OS installation method (default), timeline snapshots are enabled, except for the root file system. You can configure timeline snapshots to be taken at different intervals: hourly, daily, weekly, monthly and yearly. Old snapshots are automatically deleted. By default, the first snapshot of the last ten days, months and years is kept.

Installation snapshots

Whenever one or more packages are installed with Zypper or YaST, three installation snapshots are created. In case an important system component such as the kernel has been installed, the snapshot pair is marked as important. Old snapshots are automatically deleted. Installation snapshots are enabled by default.

Administration snapshots

Whenever you make changes to the system using Zypper or YaST, a pair of snapshots is created: one prior to the system change (“pre”) and the other one after the system change (“post”). Old snapshots are automatically deleted. Administration snapshots are enabled by default.

10.1.3 Directories that are excluded from snapshots

Certain directories need to be excluded from snapshots for different reasons. The following list shows all directories that are excluded:

`/boot/grub2/i386-pc`, `/boot/grub2/x86_64-efi`, `/boot/grub2/powerpc-ieee1275`, `/boot/grub2/s390x-emu`

: A rollback of the boot loader configuration is not supported. The directories listed above are architecture-specific. The first two directories are present on AMD64/Intel 64 machines, the latter two on IBM POWER and on IBM Z, respectively.

`/home`

: If `/home` does not reside on a separate partition, it is excluded to avoid data loss on rollbacks.

`/opt`

: Third-party products usually get installed to `/opt`. It is excluded to avoid uninstalling these applications on rollbacks.

`/srv`

: Contains data for Web and FTP servers. It is excluded to avoid data loss on rollbacks.

`/tmp`

: All directories containing temporary files and caches are excluded from snapshots.

`/usr/local`

: This directory is used when manually installing software. It is excluded to avoid uninstalling these installations on rollbacks.

`/var`

: This directory contains many variable files, including logs, temporary caches, third party products in `/var/opt`, and is the default location for virtual machine images and databases. Therefore this subvolume is created to exclude all of this variable data from snapshots and has Copy-On-Write disabled.

10.1.4 Customizing the setup

SUSE Linux Enterprise Desktop comes with a reasonable default setup, which should be sufficient for most use cases. However, all aspects of taking automatic snapshots and snapshot keeping can be configured according to your needs.

10.1.4.1 Disabling/enabling snapshots

Each of the three snapshot types (timeline, installation, administration) can be enabled or disabled independently.

Disabling/enabling timeline snapshots

Enabling. `snapper -c root set-config "TIMELINE_CREATE=yes"`

Disabling. `snapper -c root set-config "TIMELINE_CREATE=no"`
Using the YaST OS installation method (default), timeline snapshots are enabled, except for the root file system.

Disabling/enabling installation snapshots

Enabling: Install the package `snapper-zypp-plugin`

Disabling: Uninstall the package `snapper-zypp-plugin`
Installation snapshots are enabled by default.

Disabling/enabling administration snapshots

Enabling: Set `USE_SNAPPER` to `yes` in `/etc/sysconfig/yast2`.

Disabling: Set `USE_SNAPPER` to `no` in `/etc/sysconfig/yast2`.
Administration snapshots are enabled by default.

10.1.4.2 Controlling installation snapshots

Taking snapshot pairs upon installing packages with YaST or Zypper is handled by the `snapper-zypp-plugin`. An XML configuration file, `/etc/snapper/zypp-plugin.conf` defines, when to make snapshots. By default the file looks like the following:

```
<?xml version="1.0" encoding="utf-8"?>
<snapper-zypp-plugin-conf>
 <solvables>
  <solvable match="w" ❶ important="true" ❷ >kernel-* ❸ </solvable>
  <solvable match="w" important="true">dracut</solvable>
  <solvable match="w" important="true">glibc</solvable>
  <solvable match="w" important="true">systemd*</solvable>
  <solvable match="w" important="true">udev</solvable>
  <solvable match="w">*</solvable> ❹
 </solvables>
</snapper-zypp-plugin-conf>
```

1. The match attribute defines whether the pattern is a Unix shell-style wild card (`w`) or a Python regular expression (`re`).
2. If the given pattern matches and the corresponding package is marked as important (for example, kernel packages), the snapshot is also marked as important.
3. Pattern to match a package name. Based on the setting of the `match` attribute, special characters are either interpreted as shell wild cards or regular expressions. This pattern matches all package names starting with `kernel-`.
4. This line unconditionally matches all packages.

With this configuration snapshot, pairs are made whenever a package is installed (line 9). When the kernel, dracut, glibc, systemd or udev packages marked as important are installed, the snapshot pair is also marked as important (lines 4 to 8). All rules are evaluated.

To disable a rule, either delete it or deactivate it using XML comments. To prevent the system from making snapshot pairs for every package installation for example, comment line 9:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <snapper-zypp-plugin-conf>
3  <solvables>
4   <solvable match="w" important="true">kernel-*</solvable>
5   <solvable match="w" important="true">dracut</solvable>
6   <solvable match="w" important="true">glibc</solvable>
7   <solvable match="w" important="true">systemd*</solvable>
8   <solvable match="w" important="true">udev</solvable>
9   <!-- <solvable match="w">*</solvable> -->
10  </solvables>
11 </snapper-zypp-plugin-conf>
```

10.1.4.3 Creating and mounting new subvolumes

Creating a new subvolume underneath the `/` hierarchy and permanently mounting it is supported. Such a subvolume is excluded from snapshots. You need to make sure not to create it inside an existing snapshot, since you would not be able to delete snapshots anymore after a rollback.

SUSE Linux Enterprise Desktop is configured with the `/@/` subvolume which serves as an independent root for permanent subvolumes such as `/opt`, `/srv`, `/home` and others. Any new subvolumes you create and permanently mount need to be created in this initial root file system.

To do so, run the following commands. In this example, a new subvolume `/usr/important` is created from `/dev/sda2`.

```
> sudo mount /dev/sda2 -o subvol=@ /mnt
```

```
> sudo btrfs subvolume create /mnt/usr/important
> sudo umount /mnt
```

The corresponding entry in `/etc/fstab` needs to look like the following:

```
/dev/sda2 /usr/important btrfs subvol=@/usr/important 0 0
```

Tip: Disable copy-on-write (cow)

A subvolume may contain files that constantly change, such as virtualized disk images, database files, or log files. If so, consider disabling the copy-on-write feature for this volume, to avoid duplication of disk blocks. Use the `nodatacow` mount option in `/etc/fstab` to do so:

```
/dev/sda2 /usr/important btrfs nodatacow,subvol=@/usr/important 0 0
```

To alternatively disable copy-on-write for single files or directories, use the command **`chattr +C`** *PATH*.

10.1.4.4 Controlling snapshot archiving

Snapshots occupy disk space. To prevent disks from running out of space and thus causing system outages, old snapshots are automatically deleted. By default, up to ten important installation and administration snapshots and up to ten regular installation and administration snapshots are kept. If these snapshots occupy more than 50% of the root file system size, additional snapshots are deleted. A minimum of four important and two regular snapshots are always kept.

Refer to *Section 10.5.1, “Managing existing configurations”* for instructions on how to change these values.

10.1.4.5 Using Snapper on thinly provisioned LVM volumes

Apart from snapshots on `Btrfs` file systems, Snapper also supports taking snapshots on thinly provisioned LVM volumes (snapshots on regular LVM volumes are *not* supported) formatted with XFS, Ext4 or Ext3. For more information and setup instructions on LVM volumes, refer to *Book “Deployment Guide”, Chapter 7 “Expert Partitioner”, Section 7.3 “LVM configuration”*.

To use Snapper on a thinly provisioned LVM volume, you need to create a Snapper configuration for it. On LVM it is required to specify the file system with `--fstype=lvm(FILESYSTEM)`. `ext3`, `etx4` or `xfs` are valid values for *`FILESYSTEM`*. Example:

```
> sudo snapper -c lvm create-config --fstype="lvm(xfs)" /thin_lvm
```

You can adjust this configuration according to your needs as described in *Section 10.5.1, "Managing existing configurations"*.

10.2 Using Snapper to undo changes

Snapper on SUSE Linux Enterprise Desktop is preconfigured to serve as a tool that lets you undo changes made by **`zypper`** and YaST. For this purpose, Snapper is configured to create a pair of snapshots before and after each run of **`zypper`** and YaST. Snapper also lets you restore system files that have been accidentally deleted or modified. Timeline snapshots for the root partition need to be enabled for this purpose—see *Section 10.1.4.1, "Disabling/enabling snapshots"* for details.

By default, automatic snapshots as described above are configured for the root partition and its subvolumes. To make snapshots available for other partitions such as `/home` for example, you can create custom configurations.

Important: Undoing changes compared to rollback

When working with snapshots to restore data, it is important to know that there are two fundamentally different scenarios Snapper can handle:

Undoing changes

When undoing changes as described in the following, two snapshots are being compared and the changes between these two snapshots are made undone. Using this method also allows to explicitly select the files that should be restored.

Rollback

When doing rollbacks as described in *Section 10.3, "System rollback by booting from snapshots"*, the system is reset to the state at which the snapshot was taken.

When undoing changes, it is also possible to compare a snapshot against the current system. When restoring *all* files from such a comparison, this will have the same result as doing a rollback. However, using the method described in *Section 10.3, "System rollback by booting from snapshots"* for rollbacks should be preferred, since it is faster and allows you to review the system before doing the rollback.

Warning: Data consistency

There is no mechanism to ensure data consistency when creating a snapshot. Whenever a file (for example, a database) is written at the same time as the snapshot is being created, it will result in a corrupted or partly written file. Restoring such a file will cause problems. Furthermore, certain system files such as `/etc/mtab` must never be restored. Therefore it is strongly recommended to *always* closely review the list of changed files and their diffs. Only restore files that really belong to the action you want to revert.

10.2.1 Undoing YaST and Zypper changes

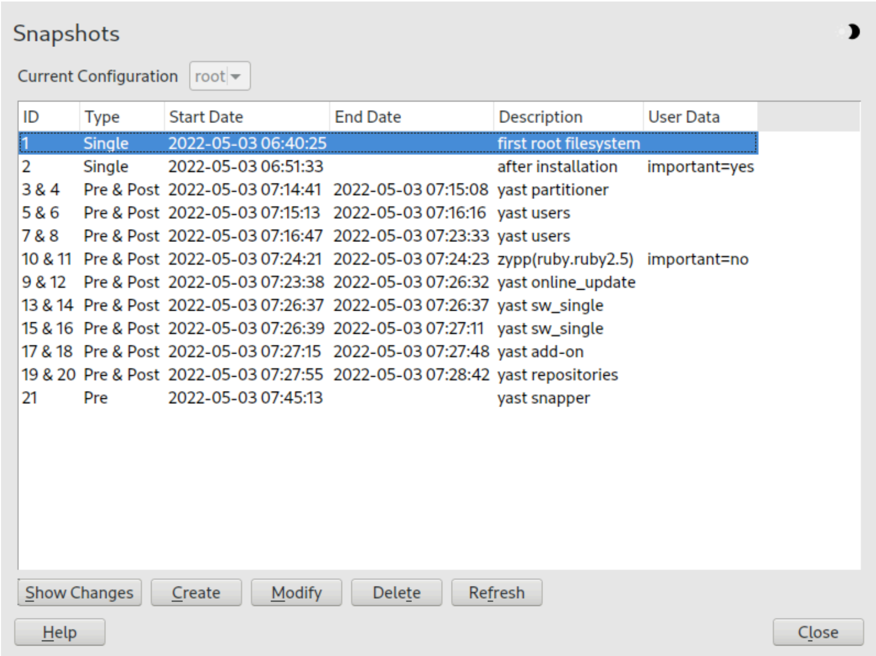
If you set up the root partition with `Btrfs` during the installation, Snapper—preconfigured for doing rollbacks of YaST or Zypper changes—will automatically be installed. Every time you start a YaST module or a Zypper transaction, two snapshots are created: a "pre-snapshot" capturing the state of the file system before the start of the module and a "post-snapshot" after the module has been finished.

Using the YaST Snapper module or the **`snapper`** command line tool, you can undo the changes made by YaST/Zypper by restoring files from the "pre-snapshot". Comparing two snapshots the tools also allow you to see which files have been changed. You can also display the differences between two versions of a file (diff).

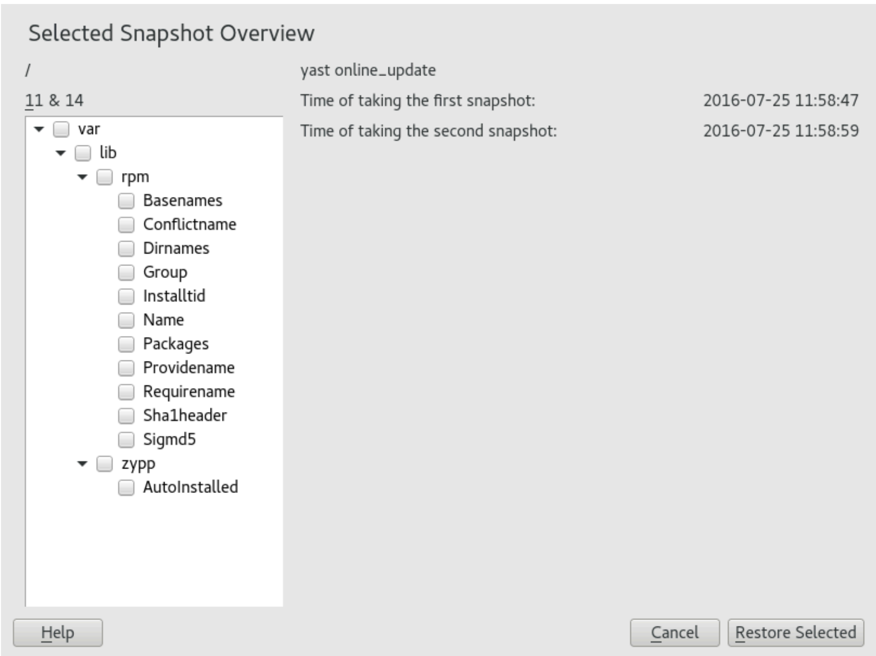
PROCEDURE 10.1: UNDOING CHANGES USING THE YAST *SNAPPER* MODULE

1. Start the *Snapper* module from the *Miscellaneous* section in YaST or by entering **`yast2 snapper`**.
2. Make sure *Current Configuration* is set to *root*. This is always the case unless you have manually added own Snapper configurations.

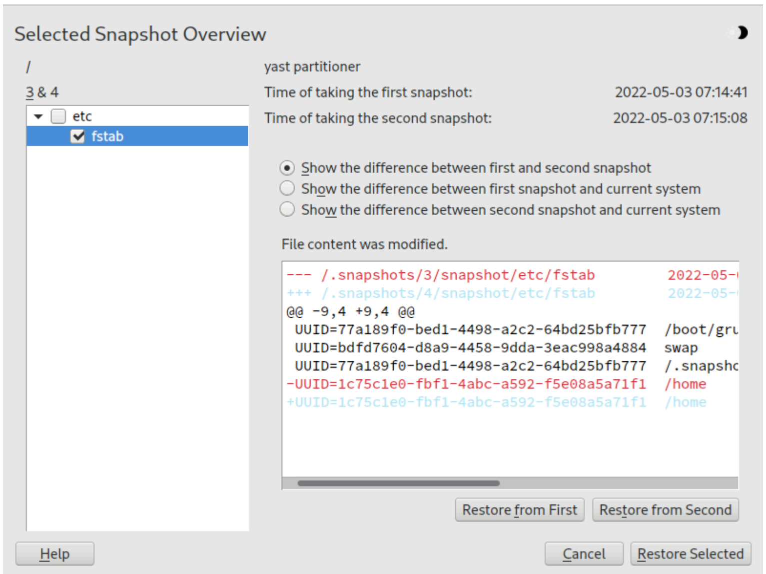
3. Choose a pair of pre- and post-snapshots from the list. Both, YaST and Zypper snapshot pairs are of the type *Pre & Post*. YaST snapshots are labeled as `zypp(y2base)` in the *Description column*; Zypper snapshots are labeled `zypp(zypper)`.



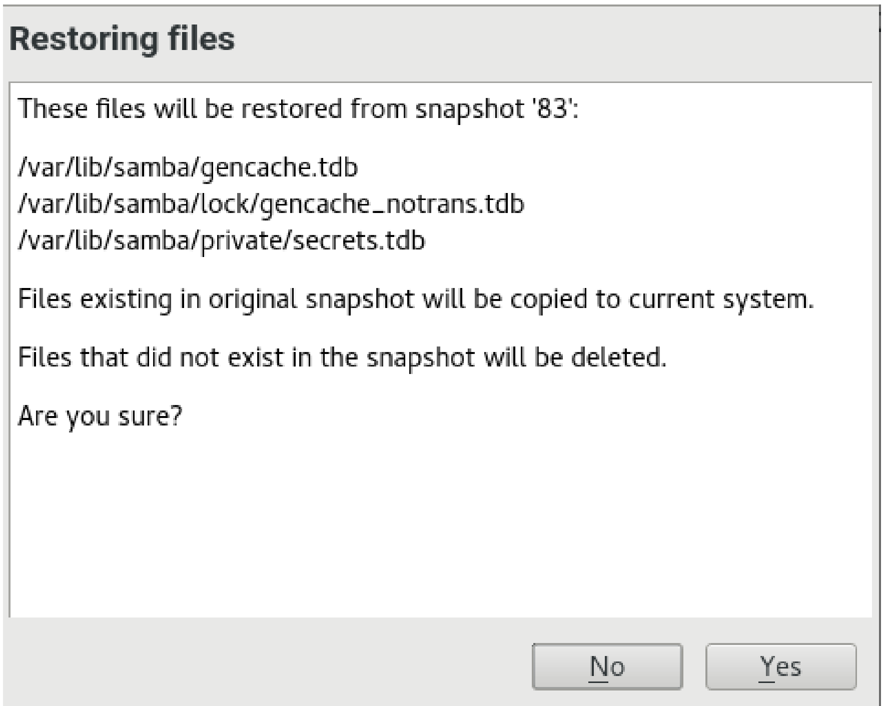
4. Click *Show Changes* to open the list of files that differ between the two snapshots.



5. Review the list of files. To display a “diff” between the pre- and post-version of a file, select it from the list.



6. To restore one or more files, select the relevant files or directories by activating the respective check box. Click *Restore Selected* and confirm the action by clicking *Yes*.



To restore a single file, activate its diff view by clicking its name. Click *Restore From First* and confirm your choice with *Yes*.

PROCEDURE 10.2: UNDOING CHANGES USING THE **snapper** COMMAND

1. Get a list of YaST and Zypper snapshots by running **`snapper list -t pre-post`**. YaST snapshots are labeled as `yast` *`MODULE_NAME`* in the *Description column*; Zypper snapshots are labeled `zypp(zypper)`.

```
> sudo snapper list -t pre-post
Pre # | Post # | Pre Date                      | Post Date                     | Description
------+--------+-------------------------------+-------------------------------+--------------
311   | 312    | Tue 06 May 2018 14:05:46 CEST | Tue 06 May 2018 14:05:52 CEST | zypp(y2base)
340   | 341    | Wed 07 May 2018 16:15:10 CEST | Wed 07 May 2018 16:15:16 CEST | zypp(zypper)
342   | 343    | Wed 07 May 2018 16:20:38 CEST | Wed 07 May 2018 16:20:42 CEST | zypp(y2base)
344   | 345    | Wed 07 May 2018 16:21:23 CEST | Wed 07 May 2018 16:21:24 CEST | zypp(zypper)
346   | 347    | Wed 07 May 2018 16:41:06 CEST | Wed 07 May 2018 16:41:10 CEST | zypp(y2base)
348   | 349    | Wed 07 May 2018 16:44:50 CEST | Wed 07 May 2018 16:44:53 CEST | zypp(y2base)
350   | 351    | Wed 07 May 2018 16:46:27 CEST | Wed 07 May 2018 16:46:38 CEST | zypp(y2base)
```

2. Get a list of changed files for a snapshot pair with **`snapper status`** *`PRE`*..*`POST`*. Files with content changes are marked with *c*, files that have been added are marked with + and deleted files are marked with -.

```
> sudo snapper status 350..351
+..... /usr/share/doc/packages/mikachan-fonts
+..... /usr/share/doc/packages/mikachan-fonts/COPYING
+..... /usr/share/doc/packages/mikachan-fonts/dl.html
c..... /usr/share/fonts/truetype/fonts.dir
c..... /usr/share/fonts/truetype/fonts.scale
+..... /usr/share/fonts/truetype/#####-p.ttf
+..... /usr/share/fonts/truetype/#####-pb.ttf
+..... /usr/share/fonts/truetype/#####-ps.ttf
+..... /usr/share/fonts/truetype/#####.ttf
c..... /var/cache/fontconfig/7ef2298fde41cc6eeb7af42e48b7d293-x86_64.cache-4
c..... /var/lib/rpm/Basenames
c..... /var/lib/rpm/Dirnames
c..... /var/lib/rpm/Group
c..... /var/lib/rpm/Installtid
c..... /var/lib/rpm/Name
c..... /var/lib/rpm/Packages
c..... /var/lib/rpm/Providename
c..... /var/lib/rpm/Requirename
c..... /var/lib/rpm/Sha1header
c..... /var/lib/rpm/Sigmd5
```

3. To display the diff for a certain file, run **`snapper diff`** *`PRE`*..*`POST`* *`FILENAME`*. If you do not specify *`FILENAME`*, a diff for all files will be displayed.

```
> sudo snapper diff 350..351 /usr/share/fonts/truetype/fonts.scale
```

```
--- /.snapshots/350/snapshot/usr/share/fonts/truetype/fonts.scale       2014-04-23
 15:58:57.000000000 +0200
+++ /.snapshots/351/snapshot/usr/share/fonts/truetype/fonts.scale       2014-05-07
 16:46:31.000000000 +0200
@@ -1,4 +1,4 @@
-1174
+1486
 ds=y:ai=0.2:luximr.ttf -b&h-luxi mono-bold-i-normal--0-0-0-0-c-0-iso10646-1
 ds=y:ai=0.2:luximr.ttf -b&h-luxi mono-bold-i-normal--0-0-0-0-c-0-iso8859-1
[...]
```

4. To restore one or more files run **`snapper -v undochange`** *`PRE`*..*`POST`* *`FILENAMES`*. If you do not specify a *`FILENAMES`*, all changed files will be restored.

```
> sudo snapper -v undochange 350..351
     create:0 modify:13 delete:7
     undoing change...
     deleting /usr/share/doc/packages/mikachan-fonts
     deleting /usr/share/doc/packages/mikachan-fonts/COPYING
     deleting /usr/share/doc/packages/mikachan-fonts/dl.html
     deleting /usr/share/fonts/truetype/#####-p.ttf
     deleting /usr/share/fonts/truetype/#####-pb.ttf
     deleting /usr/share/fonts/truetype/#####-ps.ttf
     deleting /usr/share/fonts/truetype/#####.ttf
     modifying /usr/share/fonts/truetype/fonts.dir
     modifying /usr/share/fonts/truetype/fonts.scale
     modifying /var/cache/fontconfig/7ef2298fde41cc6eeb7af42e48b7d293-x86_64.cache-4
     modifying /var/lib/rpm/Basenames
     modifying /var/lib/rpm/Dirnames
     modifying /var/lib/rpm/Group
     modifying /var/lib/rpm/Installtid
     modifying /var/lib/rpm/Name
     modifying /var/lib/rpm/Packages
     modifying /var/lib/rpm/Providename
     modifying /var/lib/rpm/Requirename
     modifying /var/lib/rpm/Sha1header
     modifying /var/lib/rpm/Sigmd5
     undoing change done
```

Warning: Reverting user additions

Reverting user additions via undoing changes with Snapper is not recommended. Since certain directories are excluded from snapshots, files belonging to these users will remain in the file system. If a user with the same user ID as a deleted user is created, this user will inherit the files. Therefore it is strongly recommended to use the YaST *User and Group Management* tool to remove users.

10.2.2 Using Snapper to restore files

Apart from the installation and administration snapshots, Snapper creates timeline snapshots. You can use these backup snapshots to restore files that have accidentally been deleted or to restore a previous version of a file. By using Snapper's diff feature you can also find out which modifications have been made at a certain point of time.

Being able to restore files is especially interesting for data, which may reside on subvolumes or partitions for which snapshots are not taken by default. To be able to restore files from home directories, for example, create a separate Snapper configuration for `/home` doing automatic timeline snapshots. See *Section 10.5, "Creating and modifying Snapper configurations"* for instructions.

Warning: Restoring files compared to rollback

Snapshots taken from the root file system (defined by Snapper's root configuration), can be used to do a system rollback. The recommended way to do such a rollback is to boot from the snapshot and then perform the rollback. See *Section 10.3, "System rollback by booting from snapshots"* for details.

Performing a rollback would also be possible by restoring all files from a root file system snapshot as described below. However, this is not recommended. You may restore single files, for example, a configuration file from the `/etc` directory, but not the complete list of files from the snapshot.

This restriction only affects snapshots taken from the root file system.

PROCEDURE 10.3: RESTORING FILES USING THE YAST *SNAPPER* MODULE

1. Start the *Snapper* module from the *Miscellaneous* section in YaST or by entering **`yast2 snapper`**.
2. Choose the *Current Configuration* from which to choose a snapshot.

3. Select a timeline snapshot from which to restore a file and choose *Show Changes*. Timeline snapshots are of the type *Single* with a description value of *timeline*.
4. Select a file from the text box by clicking the file name. The difference between the snapshot version and the current system is shown. Activate the check box to select the file for restore. Do so for all files you want to restore.
5. Click *Restore Selected* and confirm the action by clicking *Yes*.

PROCEDURE 10.4: RESTORING FILES USING THE **snapper** COMMAND

1. Get a list of timeline snapshots for a specific configuration by running the following command:

```
> sudo snapper -c CONFIG list -t single | grep timeline
```

   CONFIG needs to be replaced by an existing Snapper configuration. Use **snapper list-configs** to display a list.

2. Get a list of changed files for a given snapshot by running the following command:

```
> sudo snapper -c CONFIG status SNAPSHOT_ID..0
```

   Replace *SNAPSHOT_ID* by the ID for the snapshot from which you want to restore the files.

3. Optionally list the differences between the current file version and the one from the snapshot by running

```
> sudo snapper -c CONFIG diff SNAPSHOT_ID..0 FILE NAME
```

   If you do not specify <*FILE NAME*>, the difference for all files are shown.

4. To restore one or more files, run

```
> sudo snapper -c CONFIG -v undochange SNAPSHOT_ID..0 FILENAME1 FILENAME2
```

   If you do not specify file names, all changed files will be restored.

10.3 System rollback by booting from snapshots

The GRUB 2 version included on SUSE Linux Enterprise Desktop can boot from Btrfs snapshots. Together with Snapper's rollback feature, this allows to recover a misconfigured system. Only snapshots created for the default Snapper configuration (`root`) are bootable.

Important: Supported configuration

As of SUSE Linux Enterprise Desktop 15 SP7 system rollbacks are only supported if the default subvolume configuration of the root partition has not been changed.

When booting a snapshot, the parts of the file system included in the snapshot are mounted read-only; all other file systems and parts that are excluded from snapshots are mounted read-write and can be modified.

Important: Undoing changes compared to rollback

When working with snapshots to restore data, it is important to know that there are two fundamentally different scenarios Snapper can handle:

Undoing changes

When undoing changes as described in *Section 10.2, "Using Snapper to undo changes"*, two snapshots are compared and the changes between these two snapshots are reverted. Using this method also allows to explicitly exclude selected files from being restored.

Rollback

When doing rollbacks as described in the following, the system is reset to the state at which the snapshot was taken.

To do a rollback from a bootable snapshot, the following requirements must be met. When doing a default installation, the system is set up accordingly.

REQUIREMENTS FOR A ROLLBACK FROM A BOOTABLE SNAPSHOT

- The root file system needs to be Btrfs. Booting from LVM volume snapshots is not supported.
- The root file system needs to be on a single device. To check, run **`sudo /sbin/btrfs filesystem show`**. It needs to report `Total devices 1`. If more than `1` device is listed, your setup is not supported.

Note: Directories excluded from snapshots

Directories that are excluded from snapshots such as `/srv` (see *Section 10.1.3, "Directories that are excluded from snapshots"* for a full list) may reside on separate devices.

- The system needs to be bootable via the installed boot loader.
- Only contents of the subvolume `/` will be rolled back. It is not possible to include other subvolumes.

To perform a rollback from a bootable snapshot, do as follows:

1. Boot the system. In the boot menu choose *Bootable snapshots* and select the snapshot you want to boot. The list of snapshots is listed by date—the most recent snapshot is listed first.
2. Log in to the system. Carefully check whether everything works as expected. Note that you cannot write to any directory that is part of the snapshot. Data you write to other directories will *not* get lost, regardless of what you do next.
3. Depending on whether you want to perform the rollback or not, choose your next step:
   a. If the system is in a state where you do not want to do a rollback, reboot to boot into the current system state. You can then choose a different snapshot, or start the rescue system.
   b. To perform the rollback, run

      ```
      > sudo snapper rollback
      ```

      and reboot afterward. On the boot screen, choose the default boot entry to reboot into the reinstated system. A snapshot of the file system status before the rollback is created. The default subvolume for root will be replaced with a fresh read-write snapshot. For details, see *Section 10.3.1, "Snapshots after rollback"*.
      It is useful to add a description for the snapshot with the `-d` option. For example:

      ```
      New file system root since rollback on DATE TIME
      ```

Tip: Rolling back to a specific installation state

If snapshots are not disabled during installation, an initial bootable snapshot is created at the end of the initial system installation. You can go back to that state at any time by booting this snapshot. The snapshot can be identified by the description `after installation`.

A bootable snapshot is also created when starting a system upgrade to a service pack or a new major release (provided snapshots are not disabled).

10.3.1 Snapshots after rollback

Before a rollback is performed, a snapshot of the running file system is created. The description references the ID of the snapshot that was restored in the rollback.

Snapshots created by rollbacks receive the value `number` for the `Cleanup` attribute. The rollback snapshots are therefore automatically deleted when the set number of snapshots is reached. Refer to *Section 10.7, “Automatic snapshot clean-up”* for details. If the snapshot contains important data, extract the data from the snapshot before it is removed.

10.3.1.1 Example of rollback snapshot

For example, after a fresh installation the following snapshots are available on the system:

```
# snapper --iso list
Type   | # |     | Cleanup | Description           | Userdata
-------+---+ ... +---------+-----------------------+--------------
single | 0 |     |         | current               |
single | 1 |     |         | first root filesystem |
single | 2 |     | number  | after installation    | important=yes
```

After running **`sudo snapper rollback`** snapshot `3` is created and contains the state of the system before the rollback was executed. Snapshot `4` is the new default Btrfs subvolume and thus the system after a reboot.

```
# snapper --iso list
Type   | # |     | Cleanup | Description           | Userdata
-------+---+ ... +---------+-----------------------+--------------
single | 0 |     |         | current               |
single | 1 |     | number  | first root filesystem |
single | 2 |     | number  | after installation    | important=yes
```

```
single | 3 |     | number   | rollback backup of #1 | important=yes
single | 4 |     |          |                       |
```

10.3.2 Accessing and identifying snapshot boot entries

To boot from a snapshot, reboot your machine and choose *Start Bootloader from a read-only snapshot*. A screen listing all bootable snapshots opens. The most recent snapshot is listed first, the oldest last. Use the keys `↓` and `↑` to navigate and press `Enter` to activate the selected snapshot. Activating a snapshot from the boot menu does not reboot the machine immediately, but rather opens the boot loader of the selected snapshot.

FIGURE 10.1: BOOT LOADER: SNAPSHOTS

Warning: Booting Xen from a Btrfs snapshot using UEFI currently fails

Refer to https://www.suse.com/support/kb/doc/?id=000020602 for more details.

Each snapshot entry in the boot loader follows a naming scheme which makes it possible to identify it easily:

```
[*]❶OS❷ (KERNEL❸,DATE❹TTIME❺,DESCRIPTION❻)
```

1. If the snapshot was marked `important`, the entry is marked with a `*`.
2. Operating system label.
4. Date in the format `YYYY-MM-DD`.
5. Time in the format `HH:MM`.
6. This field contains a description of the snapshot. In case of a manually created snapshot this is the string created with the option `--description` or a custom string (see *Tip: Setting a custom description for boot loader snapshot entries*). In case of an automatically created snapshot, it is the tool that was called, for example `zypp(zypper)` or `yast_sw_single`. Long descriptions may be truncated, depending on the size of the boot screen.

Tip: Setting a custom description for boot loader snapshot entries

It is possible to replace the default string in the description field of a snapshot with a custom string. This is for example useful if an automatically created description is not sufficient, or a user-provided description is too long. To set a custom string *`STRING`* for snapshot *`NUMBER`*, use the following command:

```
> sudo snapper modify --userdata "bootloader=STRING" NUMBER
```

The description should be no longer than 25 characters—everything that exceeds this size will not be readable on the boot screen.

10.3.3 Limitations

A *complete* system rollback, restoring the complete system to the identical state as it was in when a snapshot was taken, is not possible.

10.3.3.1 Directories excluded from snapshots

Root file system snapshots do not contain all directories. See *Section 10.1.3, "Directories that are excluded from snapshots"* for details and reasons. As a general consequence, data from these directories is not restored, resulting in the following limitations.

Add-ons and third-party software may be unusable after a rollback

Applications and add-ons installing data in subvolumes excluded from the snapshot, such as `/opt`, may not work after a rollback if other parts of the application data are also installed on subvolumes included in the snapshot. Re-install the application or the add-on to solve this problem.

File access problems

If an application had changed file permissions and/or ownership in between snapshot and current system, the application may not be able to access these files. Reset permissions and/or ownership for the affected files after the rollback.

Incompatible data formats

If a service or an application has established a new data format in between snapshot and current system, the application may not be able to read the affected data files after a rollback.

Subvolumes with a mixture of code and data

Subvolumes like `/srv` may contain a mixture of code and data. A rollback may result in non-functional code. A downgrade of the PHP version, for example, may result in broken PHP scripts for the Web server.

User data

If a rollback removes users from the system, data that is owned by these users in directories excluded from the snapshot, is not removed. If a user with the same user ID is created, this user will inherit the files. Use a tool like **`find`** to locate and remove orphaned files.

10.3.3.2 No rollback of boot loader data

A rollback of the boot loader is not possible, since all “stages” of the boot loader must fit together. This cannot be guaranteed when doing rollbacks of `/boot`.

10.4 Enabling Snapper in user home directories

You may enable snapshots for users' /home directories, which supports several use cases:

- Individual users may manage their own snapshots and rollbacks.
- System users, for example, database, system, and network admins who want to track copies of configuration files, documentation, and so on.
- Samba shares with home directories and Btrfs back-end.

Each user's directory is a Btrfs subvolume of /home. It is possible to set this up manually (see *Section 10.4.3, "Manually enabling snapshots in home directories"*). However, a more convenient way is to use pam_snapper. The pam_snapper package installs the pam_snapper.so module and helper scripts, which automate user creation and Snapper configuration.

pam_snapper provides integration with the **useradd** command, pluggable authentication modules (PAM), and Snapper. By default it creates snapshots at user login and logout, and also creates time-based snapshots as certain users remain logged in for extended periods of time. You may change the defaults using the normal Snapper commands and configuration files.

10.4.1 Installing pam_snapper and creating users

The easiest way is to start with a new /home directory formatted with Btrfs, and no existing users. Install pam_snapper:

```
# zypper in pam_snapper
```

Add this line to /etc/pam.d/common-session:

```
session optional pam_snapper.so
```

Use the **/usr/lib/pam_snapper/pam_snapper_useradd.sh** script to create a new user and home directory. By default the script performs a dry run. Edit the script to change DRYRUN=1 to DRYRUN=0. Now you can create a new user:

```
# /usr/lib/pam_snapper/pam_snapper_useradd.sh \
username group passwd=password
Create subvolume '/home/username'
useradd: warning: the home directory already exists.
Not copying any file from skel directory into it.
```

The files from /etc/skel will be copied into the user's home directory at their first login. Verify that the user's configuration was created by listing your Snapper configurations:

```
# snapper list --all
Config: home_username, subvolume: /home/username
Type   | # | Pre # | Date | User | Cleanup | Description | Userdata
-------+---+-------+------+------+---------+-------------+---------
single | 0 |       |      | root |         | current     |
```

Over time, this output will become populated with a list of snapshots, which the user can manage with the standard Snapper commands.

10.4.2 Removing users

Remove users with the **/usr/lib/pam_snapper/pam_snapper_userdel.sh** script. By default it performs a dry run, so edit it to change DRYRUN=1 to DRYRUN=0. This removes the user, the user's home subvolume, Snapper configuration, and deletes all snapshots.

```
# /usr/lib/pam_snapper/pam_snapper_userdel.sh username
```

10.4.3 Manually enabling snapshots in home directories

These are the steps for manually setting up users' home directories with Snapper. /home must be formatted with Btrfs, and the users not yet created.

```
# btrfs subvol create /home/username
# snapper -c home_username create-config /home/username
# sed -i -e "s/ALLOW_USERS=\"\"/ALLOW_USERS=\"username\"/g" \
/etc/snapper/configs/home_username
# yast users add username=username home=/home/username password=password
# chown username.group /home/username
# chmod 755 /home/username/.snapshots
```

10.5 Creating and modifying Snapper configurations

The way Snapper behaves is defined in a configuration file that is specific for each partition or Btrfs subvolume. These configuration files reside under /etc/snapper/configs/.

In case the root file system is big enough (approximately 12 GB), snapshots are automatically enabled for the root file system `/` upon installation. The corresponding default configuration is named `root`. It creates and manages the YaST and Zypper snapshot. See *Section 10.5.1.1, "Configuration data"* for a list of the default values.

Note: Minimum root file system size for enabling snapshots

As explained in *Section 10.1, "Default setup"*, enabling snapshots requires additional free space in the root file system. The amount depends on the amount of packages installed and the amount of changes made to the volume that is included in snapshots. The snapshot frequency and the number of snapshots that get archived also matter.

There is a minimum root file system size that is required to automatically enable snapshots during the installation. Currently this size is approximately 12 GB. This value may change in the future, depending on architecture and the size of the base system. It depends on the values for the following tags in the file `/control.xml` from the installation media:

```
<root_base_size>
<btrfs_increase_percentage>
```

It is calculated with the following formula: *ROOT_BASE_SIZE* * (1 + *BTRFS_INCREASE_PERCENTAGE*/100)

Keep in mind that this value is a minimum size. Consider using more space for the root file system. As a rule of thumb, double the size you would use when not having enabled snapshots.

You may create your own configurations for other partitions formatted with `Btrfs` or existing subvolumes on a `Btrfs` partition. In the following example we will set up a Snapper configuration for backing up the Web server data residing on a separate, `Btrfs`-formatted partition mounted at `/srv/www`.

After a configuration has been created, you can either use **`snapper`** itself or the YaST *Snapper* module to restore files from these snapshots. In YaST you need to select your *Current Configuration*, while you need to specify your configuration for **`snapper`** with the global switch `-c` (for example, **`snapper -c myconfig list`**).

To create a new Snapper configuration, run **`snapper create-config`**:

```
> sudo snapper -c www-data❶ create-config /srv/www❷
```

1. Name of configuration file.
2. Mount point of the partition or `Btrfs` subvolume on which to take snapshots.

This command will create a new configuration file `/etc/snapper/configs/www-data` with reasonable default values (taken from `/etc/snapper/config-templates/default`). Refer to *Section 10.5.1, "Managing existing configurations"* for instructions on how to adjust these defaults.

> **Tip: Configuration defaults**
>
> Default values for a new configuration are taken from `/etc/snapper/config-templates/default`. To use your own set of defaults, create a copy of this file in the same directory and adjust it to your needs. To use it, specify the `-t` option with the create-config command:
>
> ```
> > sudo snapper -c www-data create-config -t MY_DEFAULTS /srv/www
> ```

10.5.1 Managing existing configurations

The **`snapper`** command offers several subcommands for managing existing configurations. You can list, show, delete and modify them:

Listing configurations

Use the subcommand **`snapper list-configs`** to get all existing configurations:

```
> sudo snapper list-configs
Config | Subvolume
-------+----------
root   | /
usr    | /usr
local  | /local
```

Showing a configuration

Use the subcommand **`snapper -c`** *`CONFIG`* **`get-config`** to display the specified configuration. Replace *`CONFIG`* with one of the configuration names shown by **`snapper list-configs`**. For more information about the configuration options, see *Section 10.5.1.1, "Configuration data"*.

To display the default configuration, run:

```
> sudo snapper -c root get-config
```

Modifying a configuration

Use the subcommand `snapper -c CONFIG set-config OPTION=VALUE` to modify an option in the specified configuration. Replace *CONFIG* with one of the configuration names shown by `snapper list-configs`. Possible values for *OPTION* and *VALUE* are listed in *Section 10.5.1.1, "Configuration data"*.

Deleting a configuration

Use the subcommand `snapper -c CONFIG delete-config` to delete a configuration. Replace *CONFIG* with one of the configuration names shown by `snapper list-configs`.

10.5.1.1 Configuration data

Each configuration contains a list of options that can be modified from the command line. The following list provides details for each option. To change a value, run `snapper -c CONFIG set-config "KEY=VALUE"`.

`ALLOW_GROUPS`, `ALLOW_USERS`

Granting permissions to use snapshots to regular users. See *Section 10.5.1.2, "Using Snapper as regular user"* for more information.

The default value is `""`.

`BACKGROUND_COMPARISON`

Defines whether pre and post snapshots should be compared in the background after creation.

The default value is `"yes"`.

`EMPTY_*`

Defines the clean-up algorithm for snapshots pairs with identical pre and post snapshots. See *Section 10.7.3, "Cleaning up snapshot pairs that do not differ"* for details.

`FSTYPE`

File system type of the partition. Do not change.

The default value is `"btrfs"`.

`NUMBER_*`

Defines the clean-up algorithm for installation and administration snapshots. See *Section 10.7.1, "Cleaning up numbered snapshots"* for details.

`QGROUP` / `SPACE_LIMIT`

Adds quota support to the clean-up algorithms. See *Section 10.7.5, “Adding disk quota support”* for details.

`SUBVOLUME`

Mount point of the partition or subvolume to snapshot. Do not change.
The default value is `"/"`.

`SYNC_ACL`

If Snapper is used by regular users (see *Section 10.5.1.2, “Using Snapper as regular user”*), the users must be able to access the `.snapshot` directories and to read files within them. If SYNC_ACL is set to `yes`, Snapper automatically makes them accessible using ACLs for users and groups from the ALLOW_USERS or ALLOW_GROUPS entries.
The default value is `"no"`.

`TIMELINE_CREATE`

If set to `yes`, hourly snapshots are created. Valid values: `yes`, `no`.
The default value is `"no"`.

`TIMELINE_CLEANUP` / `TIMELINE_LIMIT_*`

Defines the clean-up algorithm for timeline snapshots. See *Section 10.7.2, “Cleaning up timeline snapshots”* for details.

10.5.1.2 Using Snapper as regular user

By default Snapper can only be used by `root`. However, there are cases in which certain groups or users need to be able to create snapshots or undo changes by reverting to a snapshot:

- Web site administrators who want to take snapshots of `/srv/www`
- Users who want to take a snapshot of their home directory

For these purposes, you can create Snapper configurations that grant permissions to users or/and groups. The corresponding `.snapshots` directory needs to be readable and accessible by the specified users. The easiest way to achieve this is to set the SYNC_ACL option to `yes`.

PROCEDURE 10.5: ENABLING REGULAR USERS TO USE SNAPPER

All steps in this procedure need to be run by `root`.

1. If a Snapper configuration does not exist yet, create one for the partition or subvolume on which the user should be able to use Snapper. Refer to *Section 10.5, "Creating and modifying Snapper configurations"* for instructions. Example:

```
> sudo snapper --config web_data create /srv/www
```

2. The configuration file is created under `/etc/snapper/configs/CONFIG`, where CONFIG is the value you specified with `-c/--config` in the previous step (for example `/etc/snapper/configs/web_data`). Adjust it according to your needs. For more information, see *Section 10.5.1, "Managing existing configurations"*.

3. Set values for `ALLOW_USERS` and/or `ALLOW_GROUPS` to grant permissions to users and/or groups, respectively. Multiple entries need to be separated by **Space**. To grant permissions to the user `www_admin` for example, run:

```
> sudo snapper -c web_data set-config "ALLOW_USERS=www_admin" SYNC_ACL="yes"
```

4. The given Snapper configuration can now be used by the specified users and/or groups. You can test it with the `list` command, for example:

```
www_admin:~ > snapper -c web_data list
```

10.6 Manually creating and managing snapshots

Snapper is not restricted to creating and managing snapshots automatically by configuration; you can also create snapshot pairs ("before and after") or single snapshots manually using either the command-line tool or the YaST module.

All Snapper operations are carried out for an existing configuration (see *Section 10.5, "Creating and modifying Snapper configurations"* for details). You can only take snapshots of partitions or volumes for which a configuration exists. By default the system configuration (`root`) is used. To create or manage snapshots for your own configuration you need to explicitly choose it. Use the *Current Configuration* drop-down box in YaST or specify the `-c` on the command line (`snapper -c MYCONFIG COMMAND`).

10.6.1 Snapshot metadata

Each snapshot consists of the snapshot itself and certain metadata. When creating a snapshot you also need to specify the metadata. Modifying a snapshot means changing its metadata—you cannot modify its content. Use **`snapper list`** to show existing snapshots and their metadata:

`snapper --config home list`

Lists snapshots for the configuration `home`. To list snapshots for the default configuration (root), use **`snapper -c root list`** or **`snapper list`**.

`snapper list -a`

Lists snapshots for all existing configurations.

`snapper list -t pre-post`

Lists all pre and post snapshot pairs for the default (`root`) configuration.

`snapper list -t single`

Lists all snapshots of the type `single` for the default (`root`) configuration.

The following metadata is available for each snapshot:

- **Type**: snapshot type, see *Section 10.6.1.1, "Snapshot types"* for details. This data cannot be changed.
- **Number**: unique number of the snapshot. This data cannot be changed.
- **Pre Number**: specifies the number of the corresponding pre snapshot. For snapshots of type post only. This data cannot be changed.
- **Description**: a description of the snapshot.
- **Userdata**: an extended description where you can specify custom data in the form of a comma-separated key=value list: `reason=testing, project=foo`. This field is also used to mark a snapshot as important (`important=yes`) and to list the user that created the snapshot (user=tux).
- **Cleanup-Algorithm**: cleanup-algorithm for the snapshot, see *Section 10.7, "Automatic snapshot clean-up"* for details.

10.6.1.1 Snapshot types

Snapper knows three different types of snapshots: pre, post and single. Physically they do not differ, but Snapper handles them differently.

`pre`

Snapshot of a file system *before* a modification. Each `pre` snapshot corresponds to a `post` snapshot. For example, this is used for the automatic YaST/Zypper snapshots.

`post`

Snapshot of a file system *after* a modification. Each `post` snapshot corresponds to a `pre` snapshot. For example, this is used for the automatic YaST/Zypper snapshots.

`single`

Stand-alone snapshot. For example, this is used for the automatic hourly snapshots. This is the default type when creating snapshots.

10.6.1.2 Cleanup algorithms

Snapper provides three algorithms to clean up old snapshots. The algorithms are executed in a daily `cron` job. It is possible to define the number of different types of snapshots to keep in the Snapper configuration (see *Section 10.5.1, "Managing existing configurations"* for details).

number

Deletes old snapshots when a certain snapshot count is reached.

timeline

Deletes old snapshots having passed a certain age but keeps several hourly, daily, monthly and yearly snapshots.

empty-pre-post

Deletes pre/post snapshot pairs with empty diffs.

10.6.2 Creating snapshots

To create a snapshot, run **`snapper create`** or click *Create* in the YaST module *Snapper*. The following examples explain how to create snapshots from the command line. The YaST interface for Snapper is not explicitly described here but provides equivalent functionality.

Tip: Snapshot description

Always specify a meaningful description to later be able to identify its purpose. You can also specify additional information via the option `--userdata`.

`snapper create --from` *`17`* `--description "with package2"`

Creates a stand-alone snapshot (type single) from an existing snapshot, which is specified by the snapshot's number from **`snapper list`**. (This applies to Snapper version 0.8.4 and newer.)

`snapper create --description "Snapshot for week 2 2014"`

Creates a stand-alone snapshot (type single) for the default (`root`) configuration with a description. Because no cleanup-algorithm is specified, the snapshot will never be deleted automatically.

`snapper --config home create --description "Cleanup in ~tux"`

Creates a stand-alone snapshot (type single) for a custom configuration named `home` with a description. Because no cleanup-algorithm is specified, the snapshot will never be deleted automatically.

`snapper --config home create --description "Daily data backup" --cleanup-algorithm timeline`>

Creates a stand-alone snapshot (type single) for a custom configuration named `home` with a description. The snapshot will automatically be deleted when it meets the criteria specified for the timeline cleanup-algorithm in the configuration.

`snapper create --type pre --print-number --description "Before the Apache config cleanup" --userdata "important=yes"`

Creates a snapshot of the type `pre` and prints the snapshot number. First command needed to create a pair of snapshots used to save a “before” and “after” state. The snapshot is marked as important.

`snapper create --type post --pre-number 30 --description "After the Apache config cleanup" --userdata "important=yes"`

Creates a snapshot of the type `post` paired with the `pre` snapshot number `30`. Second command needed to create a pair of snapshots used to save a “before” and “after” state. The snapshot is marked as important.

`snapper create --command` *`COMMAND`* `--description "Before and after COMMAND"`

Automatically creates a snapshot pair before and after running *`COMMAND`*. This option is only available when using snapper on the command line.

10.6.3 Modifying snapshot metadata

Snapper allows you to modify the description, the cleanup algorithm, and the user data of a snapshot. All other metadata cannot be changed. The following examples explain how to modify snapshots from the command line. It should be easy to adopt them when using the YaST interface.

To modify a snapshot on the command line, you need to know its number. Use **`snapper list`** to display all snapshots and their numbers.

The YaST *Snapper* module already lists all snapshots. Choose one from the list and click *Modify*.

`snapper modify --cleanup-algorithm "timeline"` 10
: Modifies the metadata of snapshot 10 for the default (`root`) configuration. The cleanup algorithm is set to `timeline`.

`snapper --config home modify --description "daily backup" -cleanup-algorithm "timeline" 120`
: Modifies the metadata of snapshot 120 for a custom configuration named `home`. A new description is set and the cleanup algorithm is unset.

10.6.4 Deleting snapshots

To delete a snapshot with the YaST *Snapper* module, choose a snapshot from the list and click *Delete*.

To delete a snapshot with the command-line tool, you need to know its number. Get it by running **`snapper list`**. To delete a snapshot, run **`snapper delete`** *`NUMBER`*.

Deleting the current default subvolume snapshot is not allowed.

When deleting snapshots with Snapper, the freed space will be claimed by a Btrfs process running in the background. Thus the visibility and the availability of free space is delayed. In case you need space freed by deleting a snapshot to be available immediately, use the option `--sync` with the delete command.

Tip: Deleting snapshot pairs

When deleting a `pre` snapshot, you should always delete its corresponding `post` snapshot (and vice versa).

`snapper delete 65`

Deletes snapshot 65 for the default (`root`) configuration.

`snapper -c home delete 89 90`

Deletes snapshots 89 and 90 for a custom configuration named `home`.

`snapper delete --sync 23`

Deletes snapshot 23 for the default (`root`) configuration and makes the freed space available immediately.

Tip: Delete unreferenced snapshots

Sometimes the Btrfs snapshot is present but the XML file containing the metadata for Snapper is missing. In this case, the snapshot is not visible for Snapper and needs to be deleted manually:

```
btrfs subvolume delete /.snapshots/SNAPSHOTNUMBER/snapshot
rm -rf /.snapshots/SNAPSHOTNUMBER
```

Tip: Old snapshots occupy more disk space

If you delete snapshots to free space on your hard disk, make sure to delete old snapshots first. The older a snapshot is, the more disk space it occupies.

Snapshots are also automatically deleted by a daily cron job. Refer to *Section 10.6.1.2, “Cleanup algorithms”* for details.

10.7 Automatic snapshot clean-up

Snapshots occupy disk space and over time the amount of disk space occupied by the snapshots may become large. To prevent disks from running out of space, Snapper offers algorithms to automatically delete old snapshots. These algorithms differentiate between timeline snapshots and numbered snapshots (administration plus installation snapshot pairs). You can specify the number of snapshots to keep for each type.

Additionally, you can optionally specify a disk space quota, defining the maximum amount of disk space the snapshots may occupy. It is also possible to automatically delete pre and post snapshots pairs that do not differ.

A clean-up algorithm is always bound to a single Snapper configuration, so you need to configure algorithms for each configuration. To prevent certain snapshots from being automatically deleted, refer to *Q:*.

The default setup (`root`) is configured to do clean-up for numbered snapshots and empty pre and post snapshot pairs. Quota support is enabled—snapshots may not occupy more than 50% of the available disk space of the root partition. Timeline snapshots are disabled by default, therefore the timeline clean-up algorithm is also disabled.

10.7.1 Cleaning up numbered snapshots

Cleaning up numbered snapshots—administration plus installation snapshot pairs—is controlled by the following parameters of a Snapper configuration.

`NUMBER_CLEANUP`

Enables or disables clean-up of installation and admin snapshot pairs. If enabled, snapshot pairs are deleted when the total snapshot count exceeds a number specified with `NUMBER_LIMIT` and/or `NUMBER_LIMIT_IMPORTANT` *and* an age specified with `NUMBER_MIN_AGE`. Valid values: `yes` (enable), `no` (disable).
The default value is `"yes"`.
Example command to change or set:

```
> sudo snapper -c CONFIG set-config "NUMBER_CLEANUP=no"
```

`NUMBER_LIMIT` / `NUMBER_LIMIT_IMPORTANT`

Defines how many regular and/or important installation and administration snapshot pairs to keep. Ignored if `NUMBER_CLEANUP` is set to `"no"`.
The default value is `"2-10"` for `NUMBER_LIMIT` and `"4-10"` for `NUMBER_LIMIT_IMPORTANT`.
The cleaning algorithms delete snapshots above the specified maximum value, without taking the snapshot and file system space into account. The algorithms also delete snapshots above the minimum value until the limits for the snapshot and file system are reached.
Example command to change or set:

```
> sudo snapper -c CONFIG set-config "NUMBER_LIMIT=10"
```

Important: Ranged compared to constant values

If quota support is enabled (see *Section 10.7.5, “Adding disk quota support”*), the limit needs to be specified as a minimum-maximum range, for example, `2-10`. If quota support is disabled, a constant value, for example, `10`, needs to be provided, otherwise cleaning up fails with an error.

`NUMBER_MIN_AGE`

Defines the minimum age in seconds a snapshot must have before it can automatically be deleted. Snapshots younger than the value specified here will not be deleted, regardless of how many exist.
The default value is `"1800"`.
Example command to change or set:

```
> sudo snapper -c CONFIG set-config "NUMBER_MIN_AGE=864000"
```

Note: Limit and age

`NUMBER_LIMIT`, `NUMBER_LIMIT_IMPORTANT` and `NUMBER_MIN_AGE` are always evaluated. Snapshots are only deleted when *all* conditions are met.

If you always want to keep the number of snapshots defined with `NUMBER_LIMIT*` regardless of their age, set `NUMBER_MIN_AGE` to `0`.

The following example shows a configuration to keep the last 10 important and regular snapshots regardless of age:

```
NUMBER_CLEANUP=yes
NUMBER_LIMIT_IMPORTANT=10
NUMBER_LIMIT=10
NUMBER_MIN_AGE=0
```

If you do not want to keep snapshots beyond a certain age, set `NUMBER_LIMIT*` to `0` and provide the age with `NUMBER_MIN_AGE`.

The following example shows a configuration to only keep snapshots younger than ten days:

```
NUMBER_CLEANUP=yes
NUMBER_LIMIT_IMPORTANT=0
NUMBER_LIMIT=0
```

```
NUMBER_MIN_AGE=864000
```

10.7.2 Cleaning up timeline snapshots

Cleaning up timeline snapshots is controlled by the following parameters of a Snapper configuration.

`TIMELINE_CLEANUP`

Enables or disables clean-up of timeline snapshots. If enabled, snapshots are deleted when the total snapshot count exceeds a number specified with `TIMELINE_LIMIT_*` *and* an age specified with `TIMELINE_MIN_AGE`. Valid values: `yes`, `no`.
The default value is `"yes"`.
Example command to change or set:

```
> sudo snapper -c CONFIG set-config "TIMELINE_CLEANUP=yes"
```

`TIMELINE_LIMIT_DAILY`, `TIMELINE_LIMIT_HOURLY`, `TIMELINE_LIMIT_MONTHLY`, `TIMELINE_LIMIT_WEEKLY`, `TIMELINE_LIMIT_YEARLY`

Number of snapshots to keep for hour, day, month, week and year.
The default value for each entry is `"10"`, except for `TIMELINE_LIMIT_WEEKLY`, which is set to `"0"` by default.

`TIMELINE_MIN_AGE`

Defines the minimum age in seconds a snapshot must have before it can automatically be deleted.
The default value is `"1800"`.

EXAMPLE 10.1: EXAMPLE TIMELINE CONFIGURATION

```
TIMELINE_CLEANUP="yes"
TIMELINE_CREATE="yes"
TIMELINE_LIMIT_DAILY="7"
TIMELINE_LIMIT_HOURLY="24"
TIMELINE_LIMIT_MONTHLY="12"
TIMELINE_LIMIT_WEEKLY="4"
TIMELINE_LIMIT_YEARLY="2"
TIMELINE_MIN_AGE="1800"
```

This example configuration enables hourly snapshots which are automatically cleaned up. `TIMELINE_MIN_AGE` and `TIMELINE_LIMIT_*` are always both evaluated. In this example, the minimum age of a snapshot before it can be deleted is set to 30 minutes (1800 seconds).

Since we create hourly snapshots, this ensures that only the latest snapshots are kept. If `TIMELINE_LIMIT_DAILY` is set to not zero, this means that the first snapshot of the day is kept, too.

SNAPSHOTS TO BE KEPT

- Hourly: the last 24 snapshots that have been made.
- Daily: the first daily snapshot that has been made is kept from the last seven days.
- Monthly: the first snapshot made on the last day of the month is kept for the last twelve months.
- Weekly: the first snapshot made on the last day of the week is kept from the last four weeks.
- Yearly: the first snapshot made on the last day of the year is kept for the last two years.

10.7.3 Cleaning up snapshot pairs that do not differ

As explained in *Section 10.1.2, "Types of snapshots"*, whenever you run a YaST module or execute Zypper, a pre snapshot is created on start-up and a post snapshot is created when exiting. In case you have not made any changes there will be no difference between the pre and post snapshots. Such "empty" snapshot pairs can be automatically be deleted by setting the following parameters in a Snapper configuration:

`EMPTY_PRE_POST_CLEANUP`

If set to `yes`, pre and post snapshot pairs that do not differ will be deleted.
The default value is `"yes"`.

`EMPTY_PRE_POST_MIN_AGE`

Defines the minimum age in seconds a pre and post snapshot pair that does not differ must have before it can automatically be deleted.
The default value is `"1800"`.

10.7.4 Cleaning up manually created snapshots

Snapper does not offer custom clean-up algorithms for manually created snapshots. However, you can assign the number or timeline clean-up algorithm to a manually created snapshot. If you do so, the snapshot will join the “clean-up queue” for the algorithm you specified. You can specify a clean-up algorithm when creating a snapshot, or by modifying an existing snapshot:

`snapper create --description "Test" --cleanup-algorithm number`

Creates a stand-alone snapshot (type single) for the default (root) configuration and assigns the `number` clean-up algorithm.

`snapper modify --cleanup-algorithm "timeline" 25`

Modifies the snapshot with the number 25 and assigns the clean-up algorithm `timeline`.

10.7.5 Adding disk quota support

In addition to the number and/or timeline clean-up algorithms described above, Snapper supports quotas. You can define what percentage of the available space snapshots are allowed to occupy. This percentage value always applies to the Btrfs subvolume defined in the respective Snapper configuration.

Btrfs quotas are applied to subvolumes, not to users. You may apply disk space quotas to users and groups (for example, with the **`quota`** command) in addition to using Btrfs quotas.

If Snapper was enabled during the installation, quota support is automatically enabled. In case you manually enable Snapper at a later point in time, you can enable quota support by running **`snapper setup-quota`**. This requires a valid configuration (see *Section 10.5, “Creating and modifying Snapper configurations”* for more information).

Quota support is controlled by the following parameters of a Snapper configuration.

`QGROUP`

The Btrfs quota group used by Snapper. If not set, run **`snapper setup-quota`**. If already set, only change if you are familiar with **`man 8 btrfs-qgroup`**. This value is set with **`snapper setup-quota`** and should not be changed.

`SPACE_LIMIT`

Limit of space snapshots are allowed to use in fractions of 1 (100%). Valid values range from 0 to 1 (0.1 = 10%, 0.2 = 20%, ...).

The following limitations and guidelines apply:

- Quotas are only activated in *addition* to an existing number and/or timeline clean-up algorithm. If no clean-up algorithm is active, quota restrictions are not applied.
- With quota support enabled, Snapper will perform two clean-up runs if required. The first run will apply the rules specified for number and timeline snapshots. Only if the quota is exceeded after this run, the quota-specific rules will be applied in a second run.
- Even if quota support is enabled, Snapper will always keep the number of snapshots specified with the `NUMBER_LIMIT*` and `TIMELINE_LIMIT*` values, even if the quota will be exceeded. It is therefore recommended to specify ranged values (*MIN-MAX*) for `NUMBER_LIMIT*` and `TIMELINE_LIMIT*` to ensure the quota can be applied.
  If, for example, `NUMBER_LIMIT=5-20` is set, Snapper will perform a first clean-up run and reduce the number of regular numbered snapshots to 20. In case these 20 snapshots exceed the quota, Snapper will delete the oldest ones in a second run until the quota is met. A minimum of five snapshots will always be kept, regardless of the amount of space they occupy.

10.8 Showing exclusive disk space used by snapshots

Snapshots share data, for efficient use of storage space, so using ordinary commands like **du** and **df** will not measure used disk space accurately. When you want to free up disk space on Btrfs with quotas enabled, you need to know how much exclusive disk space is used by each snapshot, rather than shared space. Snapper 0.6 and up reports the used disk space for each snapshot in the `Used Space` column:

```
# snapper --iso list
  # | Type   | Pre # | Date                | User | Used Space | Cleanup | Description
        | Userdata
----+--------+-------+---------------------+------+------------+---------
+----------------------+-------------
 0  | single |       |                     | root |            |         | current
        |
 1* | single |       | 2019-07-22 13:08:38 | root |  16.00 KiB |         | first root
 filesystem |
 2  | single |       | 2019-07-22 14:21:05 | root |  14.23 MiB | number  | after
 installation     | important=yes
 3  | pre    |       | 2019-07-22 14:26:03 | root | 144.00 KiB | number  | zypp(zypper)
        | important=no
```

```
 4  | post   |      3 | 2019-07-22 14:26:04 | root | 112.00 KiB | number   |
        | important=no
 5  | pre    |        | 2019-07-23 08:19:36 | root | 128.00 KiB | number   | zypp(zypper)
        | important=no
 6  | post   |      5 | 2019-07-23 08:19:43 | root |  80.00 KiB | number   |
        | important=no
 7  | pre    |        | 2019-07-23 08:20:50 | root | 256.00 KiB | number   | yast sw_single
        |
 8  | pre    |        | 2019-07-23 08:23:22 | root | 112.00 KiB | number   |
 zypp(ruby.ruby2.5)     | important=no
 9  | post   |      8 | 2019-07-23 08:23:35 | root |  64.00 KiB | number   |
        | important=no
10  | post   |      7 | 2019-07-23 08:24:05 | root |  16.00 KiB | number   |
        |
```

The **btrfs** command provides another view of space used by snapshots:

```
# btrfs qgroup show -p /
qgroupid         rfer         excl parent
--------         ----         ---- ------
0/5          16.00KiB     16.00KiB ---
[...]
0/272         3.09GiB     14.23MiB 1/0
0/273         3.11GiB    144.00KiB 1/0
0/274         3.11GiB    112.00KiB 1/0
0/275         3.11GiB    128.00KiB 1/0
0/276         3.11GiB     80.00KiB 1/0
0/277         3.11GiB    256.00KiB 1/0
0/278         3.11GiB    112.00KiB 1/0
0/279         3.12GiB     64.00KiB 1/0
0/280         3.12GiB     16.00KiB 1/0
1/0           3.33GiB    222.95MiB ---
```

The `qgroupid` column displays the identification number for each subvolume, assigning a qgroup level/ID combination.

The `rfer` column displays the total amount of data referred to in the subvolume.

The `excl` column displays the exclusive data in each subvolume.

The `parent` column shows the parent qgroup of the subvolumes.

The final item, `1/0`, shows the totals for the parent qgroup. In the above example, 222.95 MiB will be freed if all subvolumes are removed. Run the following command to see which snapshots are associated with each subvolume:

```
# btrfs subvolume list -st /
ID gen top level path
```

```
-- --- --------- ----
267 298 266  @/.snapshots/1/snapshot
272 159 266  @/.snapshots/2/snapshot
273 170 266  @/.snapshots/3/snapshot
274 171 266  @/.snapshots/4/snapshot
275 287 266  @/.snapshots/5/snapshot
276 288 266  @/.snapshots/6/snapshot
277 292 266  @/.snapshots/7/snapshot
278 296 266  @/.snapshots/8/snapshot
279 297 266  @/.snapshots/9/snapshot
280 298 266  @/.snapshots/10/snapshot
```

Doing an upgrade from one service pack to another results in snapshots occupying a lot of disk space on the system subvolumes. Manually deleting these snapshots after they are no longer needed is recommended. See *Section 10.6.4, "Deleting snapshots"* for details.

10.9 Frequently asked questions

Q: *Why does Snapper never show changes in `/var/log`, `/tmp` and other directories?*

A: For certain directories, we decided to exclude them from snapshots. See *Section 10.1.3, "Directories that are excluded from snapshots"* for a list and reasons. To exclude a path from snapshots we create a subvolume for that path.

Q: *Can I boot a snapshot from the boot loader?*

A: Yes—refer to *Section 10.3, "System rollback by booting from snapshots"* for details.

Q: *Can a snapshot be protected from deletion?*

A: Currently Snapper does not offer means to prevent a snapshot from being deleted manually. However, you can prevent snapshots from being automatically deleted by clean-up algorithms. Manually created snapshots (see *Section 10.6.2, "Creating snapshots"*) have no clean-up algorithm assigned unless you specify one with `--cleanup-algorithm`. Automatically created snapshots always either have the `number` or `timeline` algorithm assigned. To remove such an assignment from one or more snapshots, proceed as follows:

1. List all available snapshots:

   ```
   > sudo snapper list -a
   ```

2. Memorize the number of the snapshots you want to prevent from being deleted.

3. Run the following command and replace the number placeholders with the numbers you memorized:

```
> sudo snapper modify --cleanup-algorithm "" #1 #2 #n
```

4. Check the result by running **`snapper list -a`** again. The entry in the column `Cleanup` should now be empty for the snapshots you modified.

Q: *Where can I get more information on Snapper?*

A: See the Snapper home page at http://snapper.io/.

11 Live kernel patching with KLP

Revision History

2025-03-31

This document describes the basic principles of the Kernel Live Patching (KLP) technology, and provides usage guidelines for the SLE Live Patching service.

KLP makes it possible to apply the latest security updates to Linux kernels without rebooting. This maximizes system uptime and availability, which is especially important for mission-critical systems.

The information provided in this document relates to the AMD64/Intel 64, POWER, and IBM Z architectures.

11.1 Advantages of Kernel Live Patching

KLP offers several benefits.

- Keeping a large number of servers automatically up to date is essential for organizations obtaining or maintaining certain compliance certifications. KLP can help achieve compliance, while reducing the need for costly maintenance windows.
- Companies that work with service-level agreement contracts must guarantee a specific level of their system accessibility and uptime. Live patching makes it possible to patch systems without incurring downtime.
- Since KLP is part of the standard system update mechanism, there is no need for specialized training or introduction of complicated maintenance routines.

11.2 Kernel Live Patching overview

Kernel live patches are delivered as packages with modified code that are separate from the main kernel package. The live patches are cumulative, so the latest patch contains all fixes from the previous ones for the kernel package. Each kernel live package is tied to the exact kernel revision for which it is issued. The live patch package version number increases with every addition of fixes.

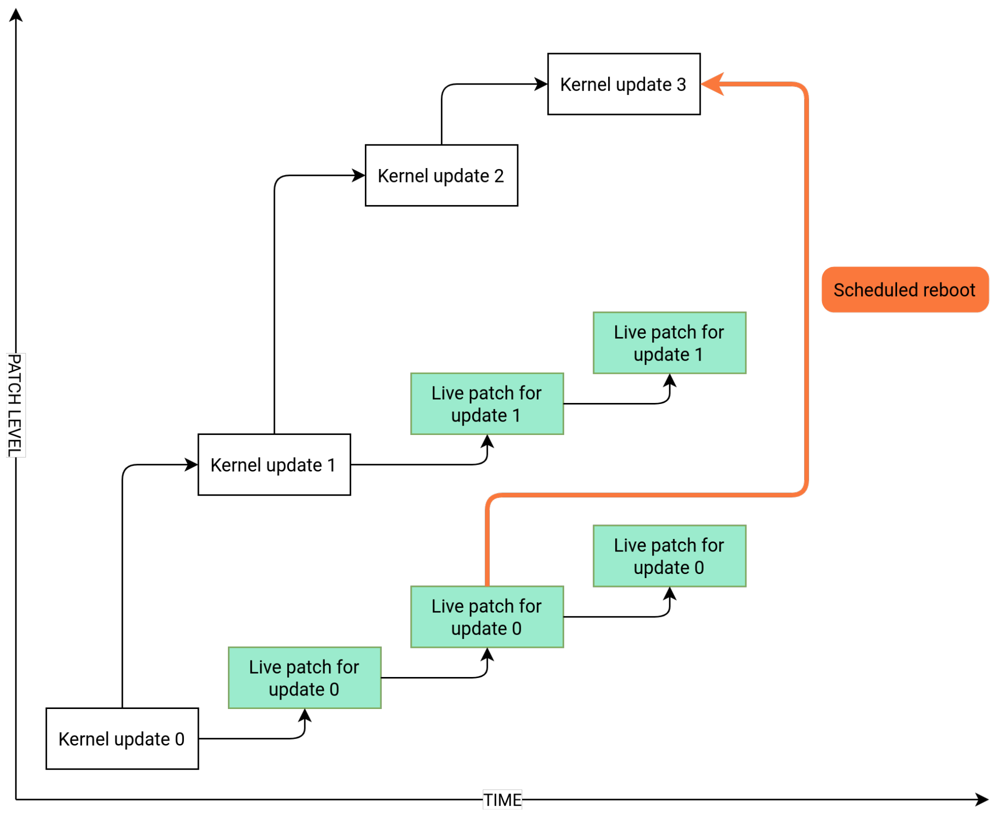
Note: Live patches and the running kernel

To determine the kernel patching status, use the `klp -v patches` command. The `uname` command's output does not change for patched kernels.

Important: Live patches compared to kernel updates

Live patches contain only critical fixes, and they do not replace regular kernel updates that require a reboot. Consider live patches as temporary measures that protect the kernel until a proper kernel update and a reboot are performed.

The diagram below illustrates the overall relationship between live patches and kernel updates. The list of CVEs and defect reports addressed by the currently active live patch can be viewed using the `klp -v patches` command.



It is possible to have multiple versions of the kernel package installed along with their live patches. These packages do not conflict. You can install updated kernel packages along with live patches for the running kernel. In this case, you may be prompted to reboot the system. Users with SLE Live Patching subscriptions are eligible for technical support as long as there are live patch updates for the running kernel (see *Section 11.5.1, "Checking expiration date of the live patch"*).

With KLP activated, every kernel update comes with a live patch package. This live patch does not contain any fixes and serves as a seed for future live patches for the corresponding kernel. These empty seed patches are called `initial patches`.

11.2.1 Kernel Live Patching scope

The scope of SLE Live Patching includes fixes for SUSE Common Vulnerability Scoring System (CVSS; SUSE CVSS is based on the CVSS v3.1 system) level 7+ vulnerabilities and bug fixes related to system stability or data corruption. However, it may not be technically feasible to create live patches for all fixes that fall under the specified categories. SUSE therefore reserves the right to skip fixes in situations where creating a kernel live patch is not possible for technical reasons. Currently, over 95% of qualifying fixes are released as live patches. For more information on CVSS (the base for the SUSE CVSS rating), see Common Vulnerability Scoring System SIG (https://www.first.org/cvss/).

11.2.2 Kernel Live Patching limitations

KLP involves replacing functions and gracefully handling replacement of interdependent function sets. This is done by redirecting calls to old code to updated code in a different memory location. Changes in data structures make the situation more complicated, as the data remain in place and cannot be extended or reinterpreted. While there are techniques that allow indirect alteration of data structures, certain fixes cannot be converted to live patches. In this situation, a system restart is the only way to apply the fixes.

11.3 Activating Kernel Live Patching using YaST

To activate KLP on your system, you need to have active SLES and SLE Live Patching subscriptions. Visit SUSE Customer Center (https://scc.suse.com/) to check the status of your subscriptions and obtain a registration code for the SLE Live Patching subscription.

To activate Kernel Live Patching on your system, follow these steps:

1. Run the **`yast2 registration`** command and click *Select Extensions*.
2. Select *SUSE Linux Enterprise Live Patching 15* in the list of available extensions and click *Next*.
3. Confirm the license terms and click *Next*.
4. Enter your SLE Live Patching registration code and click *Next*.
5. Check the *Installation Summary* and selected *Patterns*. The patterns `Live Patching` and `SLE Live Patching Lifecycle Data` should be automatically selected for installation along with additional packages to satisfy dependencies.
6. Click *Accept* to complete the installation. This installs the base Kernel Live Patching components on your system, the initial live patch, and the required dependencies.

11.4 Activating Kernel Live Patching from the command line

To activate Kernel Live Patching, you need to have active SLES and SLES Live Patching subscriptions. Visit SUSE Customer Center (https://scc.suse.com/) to check the status of your subscriptions and obtain a registration code for the SLES Live Patching subscription.

1. Run **`sudo SUSEConnect --list-extensions`**. Note the exact activation command for SLES Live Patching. Example command output (abbreviated):

```
$ SUSEConnect --list-extensions
...
SUSE Linux Enterprise Live Patching 15 SP7 x86_64
Activate with: SUSEConnect -p sle-module-live-patching/15.7/x86_64 \
  -r ADDITIONAL REGCODE
```

2. Activate SLES Live Patching using the obtained command followed by `-r` *`LIVE_PATCHING_REGISTRATION_CODE`*, for example:

```
SUSEConnect -p sle-module-live-patching/15.7/x86_64 \
  -r LIVE_PATCHING_REGISTRATION_CODE
```

3. Install the required packages and dependencies using the command **`zypper install -t pattern lp_sles`**

At this point, the system has already been live-patched.

Here is how the process works behind the scenes: when the package installation system detects that there is an installed kernel that can be live-patched, and that there is a live patch for it in the software channel, the system selects the live patch for installation. The kernel then receives the live patch fixes *as part of the package installation*. The kernel gets live-patched even before the product installation is complete.

11.5 Performing Kernel Live Patching

Kernel live patches are installed as part of regular system updates. However, there are several things you should be aware of.

- The kernel is live-patched if a `kernel-livepatch-*` package has been installed for the running kernel. You can use the command **`zypper se --details kernel-livepatch-*`** to check what kernel live patch packages are installed on your system.
- When the `kernel-default` package is installed, the update manager prompts you to reboot the system. To prevent this message from appearing, you can filter out kernel updates from the patching operation. This can be done by adding package locks with Zypper. SUSE Multi-Linux Manager also makes it possible to filter channel contents (see Live Patching with SUSE Multi-Linux Manager (https://documentation.suse.com/suma/5.0/en/suse-manager/administration/live-patching.html)).
- You can check patching status using the **`klp status`** command. To examine installed patches, run the **`klp -v patches`** command.
- Keep in mind that while there may be multiple kernel packages installed on the system, only one of them is running at any given time. Similarly, there may be multiple live patch packages installed, but only one live patch is loaded into the kernel.
- The active live patch is included in the `initrd`. This means that in case of an unexpected reboot, the system comes up with the live patch fixes applied, so there is no need to perform patching again.

11.5.1 Checking expiration date of the live patch

Make sure that the `lifecycle-data-sle-module-live-patching` is installed, then run the **`zypper lifecycle`** command. You should see expiration dates for live patches in the `Package end of support if different from product` section of the output.

Every live patch receives updates for one year from the release of the underlying kernel package. The Maintained kernels, patch updates and lifecycle (https://www.suse.com/products/live-patching/current-patches/) page allows you to check expiration dates based on the running kernel version without installing the product extension.

11.6 Troubleshooting Kernel Live Patching issues

11.6.1 Manual patch downgrade

If you find the latest live patch problematic, you can downgrade the currently installed live patch back to its previous version. We recommend performing patch downgrade before the system starts exhibiting issues. Keep in mind that a system with kernel warnings or kernel error traces in the system log may not be suitable for the patch downgrade procedure. If you are unsure whether the system meets the requirements for a patch downgrade, contact SUSE Technical Support for help.

PROCEDURE 11.1: MANUAL PATCH DOWNGRADE

1. Identify the running live patch using the **`klp -v patches`** command. You can see the currently running patch on the line starting with `RPM:`. For example:

   ```
   RPM: kernel-livepatch-6_4_0-150700_38-default-1-150700.1.23.x86_64
   ```

   The `6_4_0-150600_9-default` in the example above denotes the exact running kernel version.

2. Use the command **`zypper search -s kernel-livepatch-`***`RUNNING_KERNEL_VERSION`***`-default`** to search for previous versions of the patch. The command returns a list of available package versions. Keep in mind that for every new live patch package release, the version number increases by one. Make sure that you choose the version number one release lower than the current one.

3. Install the desired version with the command **`zypper in --oldpackage kernel-livepatch-`***`RUNNING_KERNEL_VERSION`***`-default=`***`DESIRED_VERSION`*.

12 User space live patching

Revision History

2025-02-25

This chapter describes the basic principles and usage of user space live patching.

12.1 About user space live patching

User space live patching (ULP) refers to the process of applying patches to the libraries used by a running process without interrupting them. Every time a security fix is available as a live patch, customer services will be secured after applying the live patch without restarting the processes.

ULP is supported on the following architectures:

- AMD64/Intel 64
- ppc64le (starting with SUSE Linux Enterprise Desktop 15 SP7)

Live patching operations are performed using the `ulp` tool that is part of `libpulp`. `libpulp` is a framework that consists of the `libpulp.so` library and the **`ulp`** binary that makes libraries live patchable and applies live patches.

> **Tip**
>
> You can run the **`ulp`** command either as a normal user or a privileged user via the **`sudo`** mechanism. The difference is that running **`ulp`** via **`sudo`** lets you view information of processes or patch processes that are running by `root`.

12.1.1 Prerequisites

For ULP to work, two requirements must be met.

- Install the ULP on your system by running:

```
> sudo zypper in libpulp0 libpulp-tools
```

- Applications with desired live patch support must be launched preloading the `libpulp.so.0` library. See *Section 12.1.3, "Using `libpulp`"* for more details.

12.1.2 Supported libraries

Currently, only `glibc` and `openssl` (`openssl1_1` and `openssl-3`) are supported. Additional packages will be available after they are prepared for live patching. To receive `glibc` and `openssl-1_1-` live patches, install the following packages:

```
> zypper install glibc-livepatches openssl-1_1-livepatches openssl-3-livepatches
```

12.1.3 Using `libpulp`

To enable live patching on an application, you need to preload the `libpulp.so.0` library when starting the application:

```
> LD_PRELOAD=/usr/lib64/libpulp.so.0 APPLICATION_CMD
```

12.1.3.1 Checking if a library is live patchable

To check whether a library is live patchable, use the following command:

```
> ulp livepatchable PATH_TO_LIBRARY
```

12.1.3.2 Checking if a `.so` file is a live patch container

A shared object (`.so`) is a live patch container if it contains the ULP patch description embedded into it. You can verify it with the following command:

```
> readelf -S SHARED_OBJECT | grep .ulp
```

If the output shows that there are both `.ulp` and `.ulp.rev` sections in the shared object, then it is a live patch container.

12.1.3.3 Applying live patches

Live patches provided by SUSE are shipped through usual updates if the following packages are installed: `glibc-livepatches` `openssl-livepatches` `openssl-3-livepatches`.

Custom live patches are applied using the **`ulp trigger`** command, for example:

```
> ulp trigger -p PID LIVEPATCH.so
```

Replace PID with the process ID of the running process that uses the library to be patched and LIVEPATCH.so with the actual live patch file. The command returns one of the following status messages:

SUCCESS

The live patching operation was successful.

SKIPPED

The patch was skipped because it was not designed for any library that is loaded in the process.

ERROR

An error occurred, and you can retrieve more information by inspecting the libpulp internal message buffer. See *Section 12.1.3.6, "View internal message queue"* for more information.

It is also possible to apply multiple live patches by using wildcards, for example:

```
> ulp trigger '*.so'
```

The command tries to apply every patch in the current folder to every process that have the libpulp library loaded. If the patch is not suitable for the process, it is automatically skipped. In the end, the tool shows how many patches it successfully applied to how many processes.

12.1.3.4 Reverting live patches

You can use the **ulp trigger** command to revert live patches. There are two ways to revert live patches. You can revert a live patch by using the --revert switch and passing the live patch container:

```
> ulp trigger -p PID --revert LIVEPATCH.so
```

Alternatively, it is possible to remove all patches associated with a particular library, for example:

```
> ulp trigger -p PID --revert-all=LIBRARY
```

In the example, *LIBRARY* refers to the actual library, such as libcrypto.so.1.1.

The latter approach can be useful when the source code of the original live patch is not available. Or you want to remove a specific old patch and apply a new one while the target application is still running a secure code, for example:

```
> ulp trigger -p PID  --revert-all=libcrypto.so.1.1 new_livepatch2.so
```

12.1.3.5 View applied patches

It is possible to verify which applications have live patches applied by running:

```
> ulp patches
```

The output shows which libraries are live patchable and patches loaded in programs, as well which bugs the patch addresses:

```
PID: 10636, name: test
  Livepatchable libraries:
    in /lib64/libc.so.6:
      livepatch: libc_livepatch1.so
        bug labels: jsc#SLE-0000
    in /usr/lib64/libpulp.so.0:
```

It is also possible to see which functions are patched by the live patch:

```
> ulp dump LIVEPATCH.so
```

12.1.3.6 View internal message queue

Log messages from `libpulp.so` are stored in a buffer inside the library and are not displayed unless requested by the user. To show these messages, run:

```
> ulp messages -p PID
```

12.2 ppc64le specific behavior of user space live patching

The system architecture can influence the behavior of user space live patching. On ppc64le, livepatching `systemd` (the process with PID = 1) results in the following message in `dmesg`:

```
unhandled trap code 1 in libpulp.so.0.0.0
```

This message is harmless and can be ignored, as the ppc64le kernel always display this if the process with PID = 1 executes a “trap”instruction. Refer https://bugzilla.suse.com/show_bug.cgi?id=1244263 for more information.

12.3 More information

Further information about `libpulp` is available in the project's Git repository (https://github.com/SUSE/libpulp).

13 Transactional updates

Revision History

2024-07-25

Transactional updates are available in SUSE Linux Enterprise Desktop as a technology preview for updating SLES when the root file system is read-only. Transactional updates are atomic—all updates are applied only if all succeed—and support rollbacks. It does not affect a running system as no changes are activated until after the system is rebooted. As reboots are disruptive, the administrator must decide if a reboot is more expensive than disturbing running services. If reboots are too expensive then do not use transactional updates.

Transactional updates are run daily by the **`transactional-update`** script. The script checks for available updates. If there are any updates, it creates a new snapshot of the root file system in the background, and then fetches updates from the release channels. After the new snapshot is updated, it is marked as active and will be the new default root file system after the next reboot of the system. When **`transactional-update`** is set to run automatically (which is the default behavior) it also reboots the system. Both the time that the update runs and the reboot maintenance window are configurable.

Only packages that are part of the snapshot of the root file system can be updated. If packages contain files that are not part of the snapshot, the update could fail or break the system.

RPMs that require a license to be accepted cannot be updated.

13.1 Limitations

Currently, there are certain limitations in the functionality of transactional updates. The following packages do not work with the **`transactional-update`** command:

- The `nginx` default `index.html` page may not be available
- `tomcat-webapps` and `tomcat-admin-webapps`

- `phpMyAdmin`
- `sca-appliance-*`
- `mpi-selector`
- `emacs` works except for Emacs games
- `bind` and `bind-chrootenv`
- `docbook*`
- `sblim-sfcb*`
- `texlive*`
- `iso_ent`
- `openjade`
- `opensp`
- `pcp`
- `plymouth`
- `postgresql-server-10`
- `pulseaudio-gdm-hooks`
- `smartmontools`

The updater component of the system installer does not work with a read-only file system as it has no support for transactional updates.

Further considerations:

- It is a good idea to minimize the time between updating the system and rebooting the machine.
- Only one update can be applied at a time. Be sure to reboot after an update, and before the next update is applied.
- **`update-alternatives`** should not be run after a transactional update until the machine has been rebooted.
- Do not create new system users or system groups after a transactional update until after reboot. It is acceptable to create normal users and groups (UID > 1000, GID > 1000).

- YaST is not yet aware of transactional updates. If a YaST module needs to install additional packages, this does not work. Normal system operations that modify configuration files in `/etc` work.
- For `php7-fastcgi`, you must manually create a symbolic link, `/srv/www/cgi-bin/php`, that points to `/usr/bin/php-cgi`.
- `ntp`is part of the Legacy Module for migration from older SLES versions. It is not supported on a new SUSE Linux Enterprise Desktop installation, and has been replaced by `chrony`. If you continue to use `ntp`, a fresh installation is required to work correctly with transactional updates.
- `sblim-sfcb`: the whole sblim ecosystem is incompatible with transactional update.
- **`btrfs-defrag`** from the `btrfsmaintenance` package does not work with a read-only root file system.
- For **`btrfs-balance`**, the variable `BTRFS_BALANCE_MOUNTPOINTS` in `/etc/sysconfig/btrfsmaintenance` must be changed from `/` to `/.snapshots`.
- For **`btrfs-scrub`**, the variable `BTRFS_SCRUB_MOUNTPOINTS` in `/etc/sysconfig/btrfs-maintenance` must be changed from `/` to `/.snapshots`.

13.2 Enabling `transactional-update`

You must enable the Transactional Server Module during system installation, and then select the Transactional Server System Role. Installing any package from the Transactional Server Module later in a running system is NOT supported and may break the system.

Changing the subvolume layout of the root partition, or putting subdirectories or subvolumes of the root partition on their own partitions (except `/home`, `/var`, `/srv`, and `/opt`) is not supported, and may break the system.

13.3 Managing automatic updates

Automatic updates are controlled by a **`systemd.timer`** that runs once per day. This applies all updates, and informs **`rebootmgrd`** that the machine should be rebooted. You may adjust the time when the update runs, see systemd.timer(5). To adjust the maintenance window, which is when **`rebootmgrd`** reboots the system, see rebootmgrd(8).

You can disable automatic transactional updates with this command:

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

13.4 The **transactional-update** command

The **transactional-update** command enables atomic installation or removal of updates. Updates are applied only if they all can be successfully installed. **transactional-update** creates a snapshot of your system before the update is applied, and you can restore this snapshot. All changes become active only after reboot.

`--continue`

The **--continue** option is for making multiple changes to an existing snapshot without rebooting.

The default **transactional-update** behavior is to create a new snapshot from the current root file system. If you forget something, such as installing a new package, you have to reboot to apply your previous changes, run **transactional-update** again to install the forgotten package, and reboot again. You cannot run the **transactional-update** command multiple times without rebooting to add more changes to the snapshot, because that creates separate independent snapshots that do not include changes from the previous snapshots.

Use the **--continue** option to make as many changes as you want without rebooting. A separate snapshot is made each time, and each snapshot contains all the changes you made in the previous snapshots, plus your new changes. Repeat this process as many times as you want, and when the final snapshot includes everything you want reboot the system, and your final snapshot becomes the new root file system.

Another useful feature of the **--continue** option is you may select any existing snapshot as the base for your new snapshot. The following example demonstrates running **transactional-update** to install a new package in a snapshot based on snapshot 13, and then running it again to install another package:

```
# transactional-update pkg install package_1
```

```
# transactional-update --continue 13 pkg install package_2
```

The **--continue [num]** option calls **snapper create --from**, see *Section 10.6.2, “Creating snapshots”*.

`cleanup`

If the current root file system is identical to the active root file system (after a reboot, before **`transactional-update`** creates a new snapshot with updates), all old snapshots without a cleanup algorithm get a cleanup algorithm set. This ensures that old snapshots are deleted by Snapper. (See the section about cleanup algorithms in snapper(8).) This also removes all unreferenced (and thus unused) `/etc` overlay directories in `/var/lib/overlay`:

```
# transactional-update cleanup
```

`pkg in/install`

Installs individual packages from the available channels using the **`zypper install`** command. This command can also be used to install Program Temporary Fix (PTF) RPM files.

```
# transactional-update pkg install package_name
```

or

```
# transactional-update pkg install rpm1 rpm2
```

`pkg rm/remove`

Removes individual packages from the active snapshot using the **`zypper remove`** command. This command can also be used to remove PTF RPM files.

```
# transactional-update pkg remove package_name
```

`pkg up/update`

Updates individual packages from the active snapshot using the **`zypper update`** command. Only packages that are part of the snapshot of the base file system can be updated.

```
# transactional-update pkg update package_name
```

`up/update`

If there are new updates available, a new snapshot is created and **`zypper up/update`** updates the snapshot.

```
# transactional-update up
```

`dup`

If there are new updates available, a new snapshot is created and **`zypper dup –no-allow-vendor-change`** updates the snapshot. The snapshot is activated afterwards and becomes the new root file system after reboot.

```
# transactional-update dup
```

`patch`

If there are new updates available, a new snapshot is created and **`zypper patch`** updates the snapshot.

```
# transactional-update patch
```

`rollback`

This sets the default subvolume. On systems with a read-write file system **`snapper rollback`** is called. On a read-only file system and without any argument, the current system is set to a new default root file system. If you specify a number, that snapshot is used as the default root file system. On a read-only file system, it does not create any additional snapshots.

```
# transactional-update rollback snapshot_number
```

`grub.cfg`

This creates a new GRUB2 configuration. Sometimes it is necessary to adjust the boot configuration, for example, adding additional kernel parameters. Edit *`/etc/default/grub`*, run **`transactional-update grub.cfg`**, and then reboot to activate the change. You must immediately reboot, or the new GRUB2 configuration gets overwritten with the default by the next **`transactional-update`** run.

```
# transactional-update grub.cfg
```

`reboot`

This parameter triggers a reboot after the action is completed.

```
# transactional-update dup reboot
```

`--help`

This prints a help screen with options and subcommands.

```
# transactional-update --help
```

13.5 Troubleshooting

If the upgrade fails, run **`supportconfig`** to collect log data. Provide the resulting files, including `/var/log/transactional-update.log` to SUSE Support.

14 Remote graphical sessions with VNC

Revision History

2024-05-13

Virtual Network Computing (VNC) enables you to access a remote computer via a graphical desktop, and run remote graphical applications. VNC is platform-independent and accesses the remote machine from any operating system. This chapter describes how to connect to a VNC server with the desktop clients vncviewer and Remmina, and how to operate a VNC server.

SUSE Linux Enterprise Desktop supports two different kinds of VNC sessions: one-time sessions that “live” While the VNC connection from the client is kept up, and persistent sessions that “live” until they are explicitly terminated.

A VNC server can offer both kinds of sessions simultaneously on different ports, but an open session cannot be converted from one type to the other.

14.1 The `vncviewer` client

To connect to a VNC service provided by a server, a client is needed. The default in SUSE Linux Enterprise Desktop is **`vncviewer`**, provided by the `tigervnc` package.

14.1.1 Connecting using the vncviewer CLI

To start your VNC viewer and initiate a session with the server, use the command:

```
> vncviewer jupiter.example.com:1
```

Instead of the VNC display number you can also specify the port number with two colons:

```
> vncviewer jupiter.example.com::5901
```

Note: Display and port number

The actual display or port number you specify in the VNC client must be the same as the display or port number selected when configuring a VNC server on the target machine. See *Section 14.4, “Configuring persistent VNC server sessions”* for further info.

14.1.2 Connecting using the vncviewer GUI

When running **`vncviewer`** without specifying **`--listen`** or a host to connect to, it shows a window asking for connection details. Enter the host into the *VNC server* field like in *Section 14.1.1, "Connecting using the vncviewer CLI"* and click *Connect*.

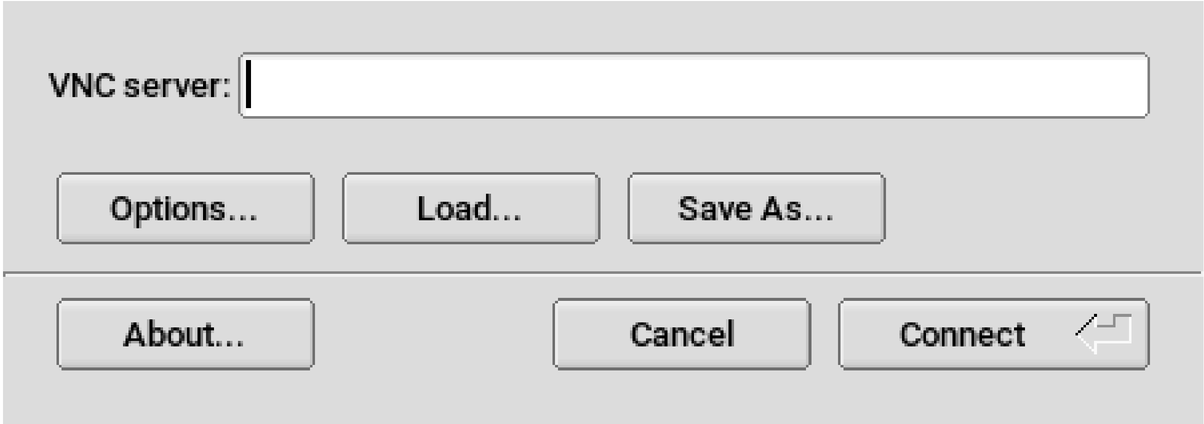


FIGURE 14.1: VNCVIEWER

14.1.3 Notification of unencrypted connections

The VNC protocol supports different kinds of encrypted connections, not to be confused with password authentication. If a connection does not use TLS, the text "(Connection not encrypted!)" can be seen in the window title of the VNC viewer.

14.2 Remmina: the remote desktop client

Remmina is a modern and feature-rich remote desktop client. It supports several access methods, for example, VNC, SSH, RDP and Spice.

14.2.1 Installation

To use Remmina, verify whether the `remmina` package is installed on your system, and install it if not. Remember to install the VNC plug-in for Remmina as well:

```
# zypper in remmina remmina-plugin-vnc
```

14.2.2 Main window

Run Remmina by entering the **`remmina`** command.

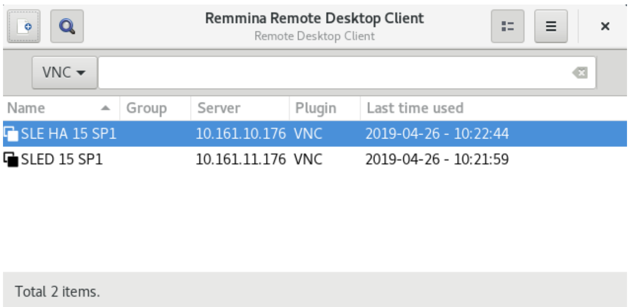


FIGURE 14.2: REMMINA'S MAIN WINDOW

The main application window shows the list of stored remote sessions. Here you can add and save a new remote session, quick-start a new session without saving it, start a previously saved session, or set Remmina's global preferences.

14.2.3 Adding remote sessions

To add and save a new remote session, click in the top left of the main window. The *Remote Desktop Preference* window opens.

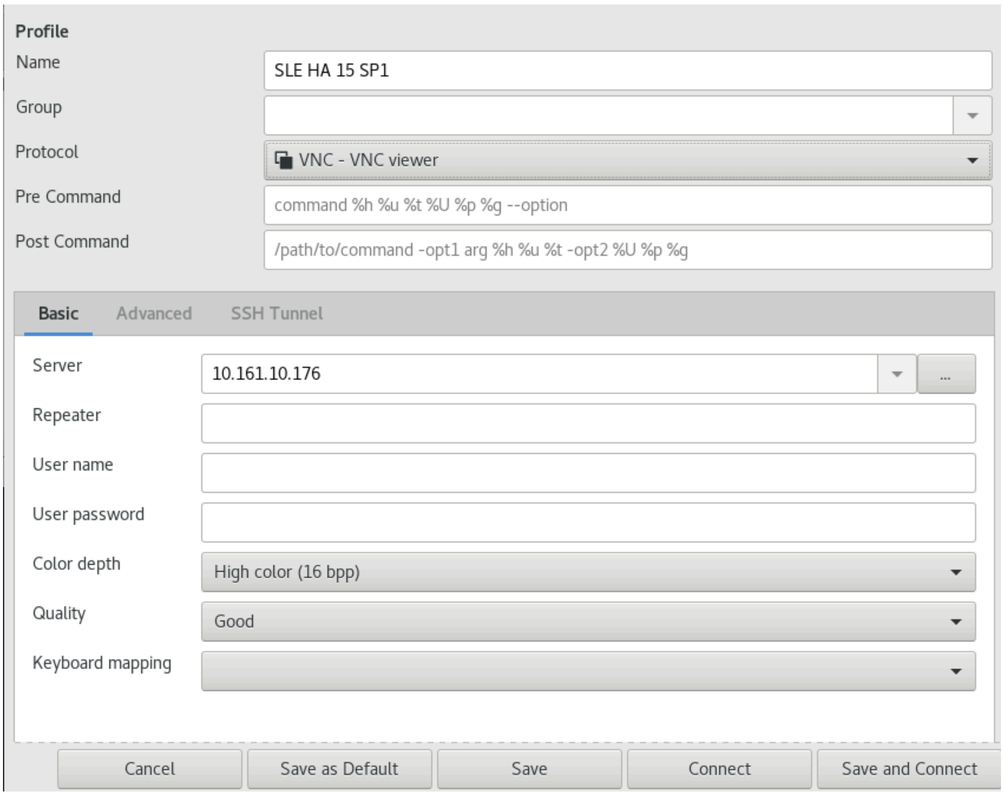


FIGURE 14.3: REMOTE DESKTOP PREFERENCE

Complete the fields that specify your newly added remote session profile. The most important are:

Name

Name of the profile. It will be listed in the main window.

Protocol

The protocol to use when connecting to the remote session, for example, VNC.

Server

The IP or DNS address and display number of the remote server.

User name, password

Credentials to use for remote authentication. Leave empty for no authentication.

Color depth, quality

Select the best options according to your connection speed and quality.

Select the *Advanced* tab to enter more specific settings.

Tip: Disable encryption

If the communication between the client and the remote server is not encrypted, activate *Disable encryption*, otherwise the connection fails.

Select the *SSH* tab for advanced SSH tunneling and authentication options.

Confirm with *Save*. Your new profile is now listed in the main window.

14.2.4 Starting remote sessions

You can either start a previously saved session, or quick-start a remote session without saving the connection details.

14.2.4.1 Quick-starting remote sessions

To start a remote session quickly without adding and saving connection details, use the drop-down box and text box at the top of the main window.

FIGURE 14.4: QUICK-STARTING

Select the communication protocol from the drop-down list, for example, “VNC”, then enter the VNC server DNS or IP address followed by a colon and a display number, and confirm with `Enter` .

14.2.4.2 Opening saved remote sessions

To open a specific remote session, double-click it from the list of sessions.

14.2.4.3 Remote sessions window

Remote sessions are opened in tabs of a separate window. Each tab hosts one session. The toolbar on the left of the window helps you manage the windows/sessions. For example, toggle full-screen mode, resize the window to match the display size of the session, send specific keystrokes to the session, take screenshots of the session, or set the image quality.

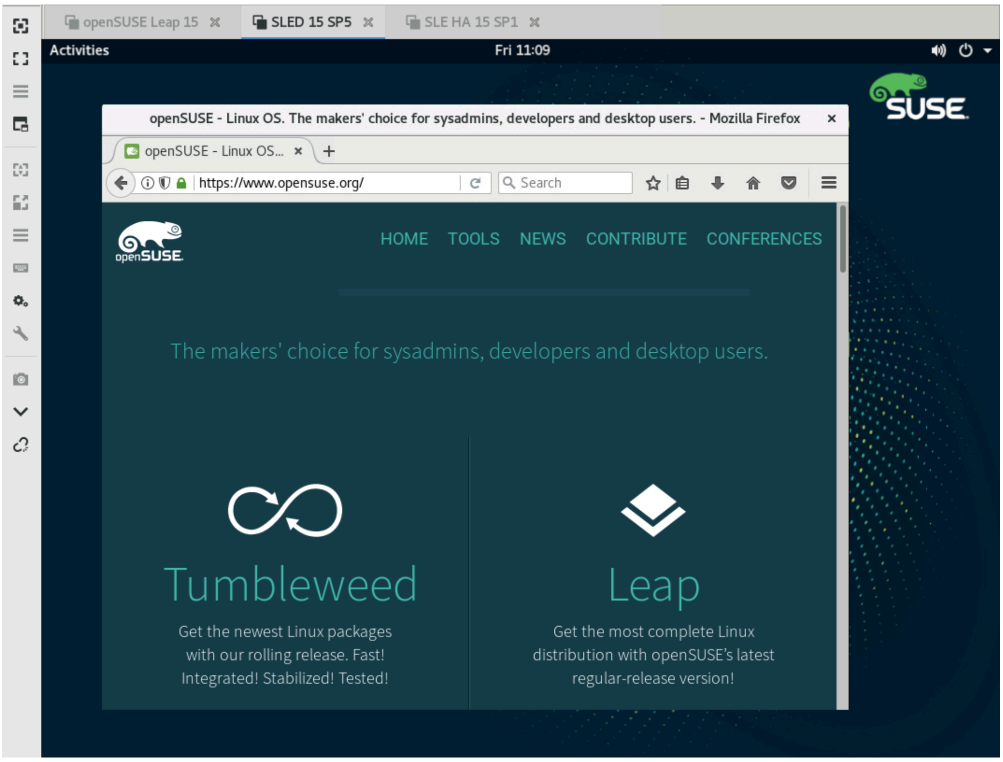


FIGURE 14.5: REMMINA VIEWING REMOTE SESSION

14.2.5 Editing, copying, and deleting saved sessions

To *edit* a saved remote session, right-click its name in Remmina's main window and select *Edit*. Refer to *Section 14.2.3, “Adding remote sessions”* for the description of the relevant fields.

To *copy* a saved remote session, right-click its name in Remmina's main window and select *Copy*. In the *Remote Desktop Preference* window, change the name of the profile, optionally adjust relevant options, and confirm with *Save*.

To *Delete* a saved remote session, right-click its name in Remmina's main window and select *Delete*. Confirm with *Yes* in the next dialog.

14.2.6 Running remote sessions from the command line

If you need to open a remote session from the command line or from a batch file without first opening the main application window, use the following syntax:

```
> remmina -c profile_name.remmina
```

Remmina's profile files are stored in the `.local/share/remmina/` directory in your home directory. To determine which profile file belongs to the session you want to open, run Remmina, click the session name in the main window, and read the path to the profile file in the window's status line at the bottom.

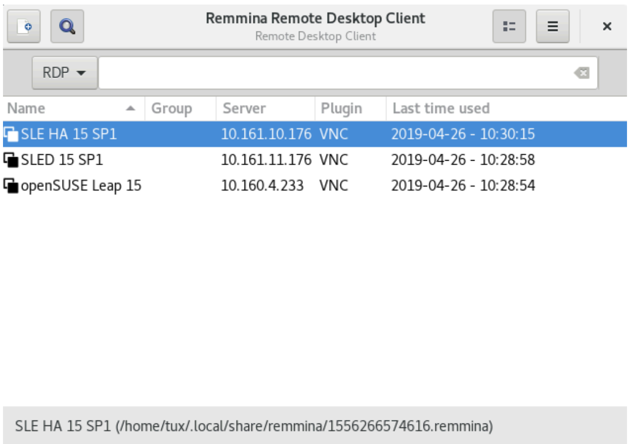


FIGURE 14.6: READING PATH TO THE PROFILE FILE

While Remmina is not running, you can rename the profile file to a more reasonable file name, such as `sle15.remmina`. You can even copy the profile file to your custom directory and run it using the **`remmina -c`** command from there.

14.3 Configuring one-time sessions on the VNC server

A one-time session is initiated by the remote client. It starts a graphical login screen on the server. This way you can choose the user which starts the session and, if supported by the login manager, the desktop environment. When you cancel the client connection to such a VNC session, all applications started within that session are terminated, too. One-time VNC sessions cannot be shared, but it is possible to have multiple sessions on a single host at the same time.

PROCEDURE 14.1: ENABLING ONE-TIME VNC SESSIONS

1. Start *YaST* › *Network Services* › *Remote Administration (VNC)*.
2. Check *Allow Remote Administration Without Session Management*.
3. Activate *Enable access using a web browser* if you plan to access the VNC session in a Web browser window.

4. If necessary, also check *Open Port in Firewall* (for example, when your network interface is configured to be in the External Zone). If you have more than one network interface, restrict opening the firewall ports to a specific interface via *Firewall Details*.
5. Confirm your settings with *Next*.
6. In case not all needed packages are available yet, you need to approve the installation of missing packages.

Tip: Restart the display manager

YaST makes changes to the display manager settings. You need to log out of your current graphical session and restart the display manager for the changes to take effect.

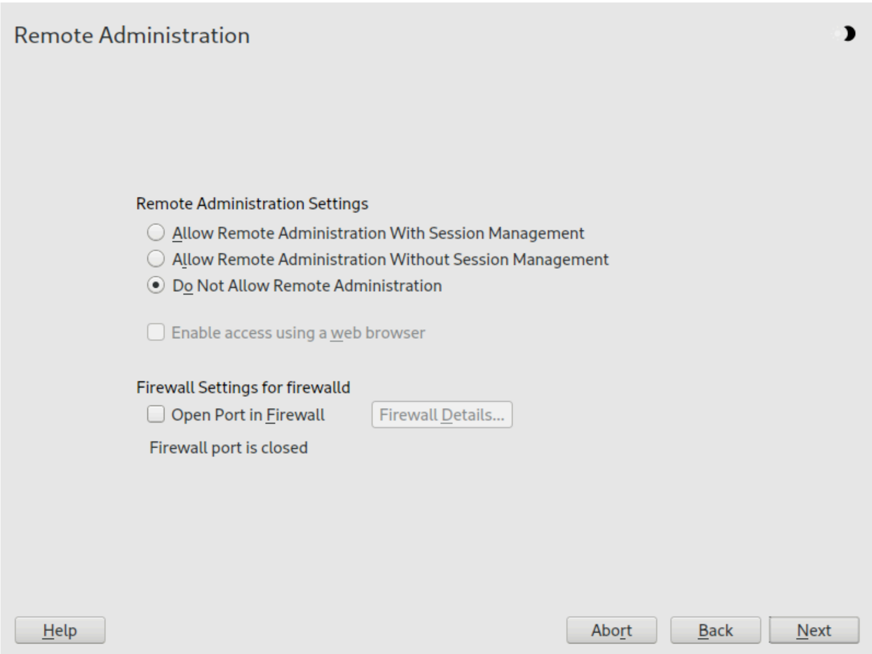


FIGURE 14.7: REMOTE ADMINISTRATION

14.3.1 Available configurations

The default configuration on SUSE Linux Enterprise Desktop serves sessions with a resolution of 1024x768 pixels at a color depth of 16-bit. The sessions are available on ports `5901` for “regular” VNC viewers (equivalent to VNC display `1`) and on port `5801` for Web browsers.

Other configurations can be made available on different ports. Ask your system administrator for details if you need to modify the configuration.

VNC display numbers and X display numbers are independent in one-time sessions. A VNC display number is manually assigned to every configuration that the server supports (:1 in the example above). Whenever a VNC session is initiated with one of the configurations, it automatically gets a free X display number.

By default, both the VNC client and server try to communicate securely via a self-signed SSL certificate, which is generated after installation. You can either use the default one, or replace it with your own. When using the self-signed certificate, you need to confirm its signature before the first connection—both in the VNC viewer and the Web browser.

> **Tip**
>
> Certain VNC clients refuse to establish a secure connection via the default self-signed certificate. For example, the Vinagre client verifies the certification against the GnuTLS global trust store and fails if the certificate is self-signed. In such a case, either use an encryption method other than `x509`, or generate a properly signed certificate for the VNC server and import it to the client's system trust store.

14.3.2 Initiating a one-time VNC session

To connect to a one-time VNC session, a VNC viewer must be installed, see also *Section 14.1, “The **`vncviewer`** client”*. Alternatively use a JavaScript-capable Web browser to view the VNC session by entering the following URL: `http://jupiter.example.com:5801`

14.3.3 Configuring one-time VNC sessions

You can skip this section, if you do not need or want to modify the default configuration.

One-time VNC sessions are started via the `systemd` socket `xvnc.socket`. By default it offers six configuration blocks: three for VNC viewers (`vnc1` to `vnc3`), and three serving a JavaScript client (`vnchttpd1` to `vnchttpd3`). By default only `vnc1` and `vnchttpd1` are active.

To activate the VNC server socket at boot time, run the following command:

```
> sudo  systemctl enable xvnc.socket
```

To start the socket immediately, run:

```
> sudo  systemctl start xvnc.socket
```

The **`Xvnc`** server can be configured via the `server_args` option. For a list of options, see **`Xvnc --help`**.

When adding custom configurations, make sure they are not using ports that are already in use by other configurations, other services, or existing persistent VNC sessions on the same host.

Activate configuration changes by entering the following command:

```
> sudo systemctl reload xvnc.socket
```

Important: Firewall and VNC ports

When activating Remote Administration as described in *Procedure 14.1, "Enabling one-time VNC sessions"*, the ports `5801` and `5901` are opened in the firewall. If the network interface serving the VNC sessions is protected by a firewall, you need to manually open the respective ports when activating additional ports for VNC sessions. See *Book "Security and Hardening Guide", Chapter 23 "Masquerading and firewalls"* for instructions.

14.4 Configuring persistent VNC server sessions

A persistent session can be accessed from multiple clients simultaneously. This is ideal for demonstration purposes where one client has full access and all other clients have view-only access. Another use case are training sessions where the trainer may need access to the trainee's desktop.

Tip: Connecting to a persistent VNC session

To connect to a persistent VNC session, a VNC viewer must be installed. Refer to *Section 14.1, "The **vncviewer** client"* for more details. Alternatively, use a JavaScript-capable Web browser to view the VNC session by entering the following URL: `http://jupiter.example.com:5801`.

14.4.1 VNC session initiated using vncmanager

PROCEDURE 14.2: ENABLING PERSISTENT VNC SESSIONS

1. Start *YaST* › *Network Services* › *Remote Administration (VNC)*.
2. Activate *Allow Remote Administration With Session Management*.
3. Activate *Enable access using a web browser* if you plan to access the VNC session in a Web browser window.
4. If necessary, also check *Open Port in Firewall* (for example, when your network interface is configured to be in the External Zone). If you have more than one network interface, restrict opening the firewall ports to a specific interface via *Firewall Details*.
5. Confirm your settings with *Next*.
6. In case not all needed packages are available yet, you need to approve the installation of missing packages.

> **Tip: Restart the display manager**
> YaST makes changes to the display manager settings. You need to log out of your current graphical session and restart the display manager for the changes to take effect.

14.4.1.1 Configuring persistent VNC sessions

After you enable the VNC session management as described in *Procedure 14.2, "Enabling persistent VNC sessions"*, you can normally connect to the remote session with your favorite VNC viewer, such as **`vncviewer`** or Remmina. After you log in, the "VNC" icon appears in the system tray of your desktop environment. Click the icon to open the *VNC Session* window. If your desktop environment does not support icons in the system tray, run **`vncmanager-controller`** manually.

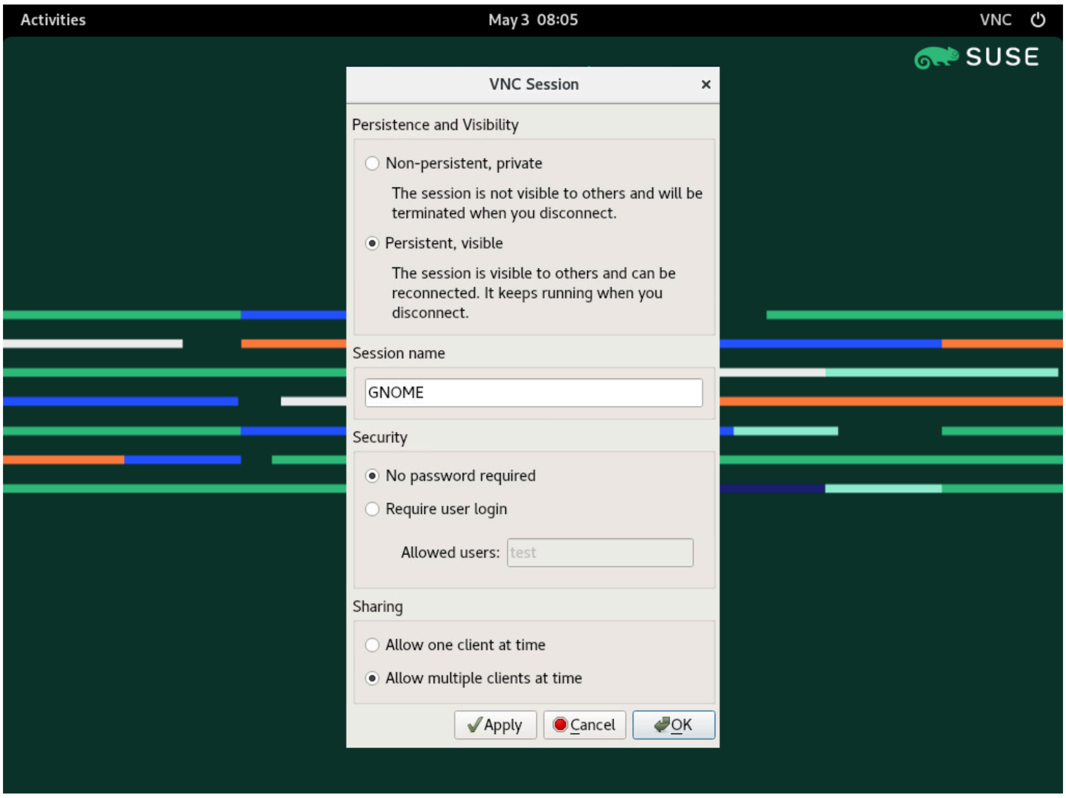


FIGURE 14.8: VNC SESSION SETTINGS

There are several settings that influence the VNC session's behavior:

Non-persistent, private

This is equivalent to a one-time session. It is not visible to others and is terminated after you disconnect from it. Refer to *Section 14.3, "Configuring one-time sessions on the VNC server"* for more information.

Persistent, visible

The session is visible to other users and keeps running even after you disconnect from it.

Session name

Specify the name of the persistent session so that it is easily identified when reconnecting.

No password required

The session is freely accessible without having to log in under user credentials.

Require user login

You need to log in with a valid user name and password to access the session. Lists the valid user names in the *Allowed users* text box.

Allow one client at a time

Prevents multiple users from joining the session at the same time.

Allow multiple clients at a time

Allows multiple users to join the persistent session at the same time. Useful for remote presentations or training sessions.

Confirm with *OK*.

14.4.1.2 Joining persistent VNC sessions

After you set up a persistent VNC session as described in *Section 14.4.1.1, "Configuring persistent VNC sessions"*, you can join it with your VNC viewer. After your VNC client connects to the server, you are prompted to choose whether you want to create a new session or join the existing one:

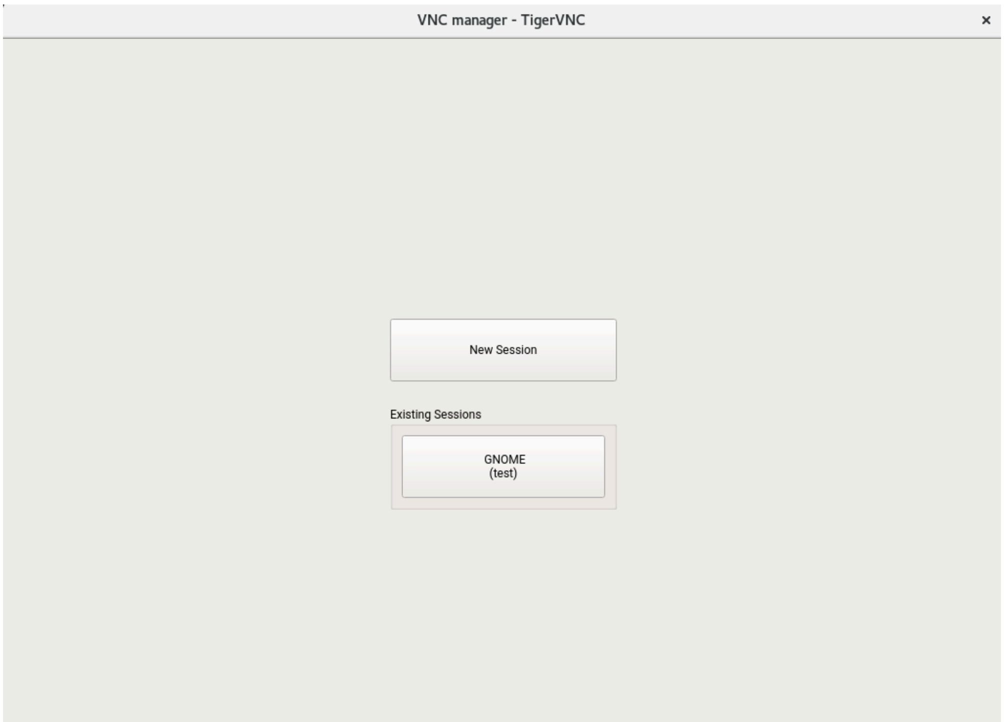


FIGURE 14.9: JOINING A PERSISTENT VNC SESSION

After you click the name of the existing session, you may be asked for login credentials, depending on the persistent session settings.

14.5 Configuring encryption on the VNC server

If the VNC server is set up properly, all communication between the VNC server and the client is encrypted. The authentication happens at the beginning of the session; the actual data transfer only begins afterward.

Whether for a one-time or a persistent VNC session, security options are configured via the `-securitytypes` parameter of the **`/usr/bin/Xvnc`** command located on the `server_args` line. The `-securitytypes` parameter selects both authentication method and encryption. It has the following options:

AUTHENTICATIONS

None, TLSNone, x509None

No authentication.

VncAuth, TLSVnc, x509Vnc

Authentication using custom password.

Plain, TLSPlain, x509Plain

Authentication using PAM to verify user's password.

ENCRYPTIONS

None, vncAuth, plain

No encryption.

TLSNone, TLSVnc, TLSPlain

Anonymous TLS encryption. Everything is encrypted, but there is no verification of the remote host. So you are protected against passive attackers, but not against man-in-the-middle attackers.

X509None, x509Vnc, x509Plain

TLS encryption with certificate. If you use a self-signed certificate, you will be asked to verify it on the first connection. On subsequent connections you will be warned only if the certificate changed. So you are protected against everything except man-in-the-middle on the first connection (similar to typical SSH usage). If you use a certificate signed by a certificate authority matching the machine name, then you get full security (similar to typical HTTPS usage).

Tip

Certain VNC clients refuse to establish a secure connection via the default self-signed certificate. For example, the Vinagre client verifies the certification against the GnuTLS global trust store and fails if the certificate is self-signed. In such a case, either use an encryption method other than `x509`, or generate a properly signed certificate for the VNC server and import it to the client's system trust store.

Tip: Path to certificate and key

With X509 based encryption, you need to specify the path to the X509 certificate and the key with `-X509Cert` and `-X509Key` options.

If you select multiple security types separated by comma, the first one supported and allowed by both client and server is used. That way, you can configure opportunistic encryption on the server. This is useful if you need to support VNC clients that do not support encryption.

On the client, you can also specify the allowed security types to prevent a downgrade attack if you are connecting to a server which you know has encryption enabled (although our vncviewer warns you with the `Connection not encrypted!` message in that case).

14.6 Compatibility with Wayland

The Remote Administration (VNC) feature relies on X11 and may result in an empty screen if Wayland is enabled. The display manager must be configured to use X11 instead of Wayland. For `gdm`, edit `/etc/gdm/custom.conf`. In the `[daemon]` section, add `WaylandEnable=false` to the configuration file. When logging in, the user must choose an X11-compatible session as well. If you wish to remove the Wayland option for GNOME, you can remove and lock the `gnome-session-wayland` package.

15 File copying with RSync

Revision History

2024-06-21

Today, a typical user has several computers: home and workplace machines, a laptop, a smartphone or a tablet. This makes the task of keeping files and documents in synchronization across multiple devices all the more important.

Warning: Risk of data loss

Before you start using a synchronization tool, you should familiarize yourself with its features and functionality. Make sure to back up your important files.

15.1 Conceptual overview

For synchronizing a large amount of data over a slow network connection, Rsync offers a reliable method of transmitting only changes within files. This applies not only to text files but also binary files. To detect the differences between files, Rsync subdivides the files into blocks and computes check sums over them.

Detecting changes requires certain computing power. So make sure that machines on both ends have enough resources, including RAM.

Rsync can be particularly useful when large amounts of data containing only minor changes need to be transmitted regularly. This is often the case when working with backups. Rsync can also be useful for mirroring staging servers that store complete directory trees of Web servers to a Web server in a DMZ.

Despite its name, Rsync is not a synchronization tool. Rsync is a tool that copies data only in one direction at a time. It does not and cannot do the reverse. If you need a bidirectional tool which can synchronize both source and destination, use Csync.

15.2 Basic syntax

Rsync is a command-line tool that has the following basic syntax:

```
rsync [OPTION] SOURCE [SOURCE]... DEST
```

You can use Rsync on any local or remote machine, provided you have access and write permissions. It is possible to have multiple *SOURCE* entries. The *SOURCE* and *DEST* placeholders can be paths, URLs or both.

Below are the most common Rsync options:

`-v`
: Outputs more verbose text

`-a`
: Archive mode; copies files recursively and preserves time stamps, user/group ownership, file permissions, and symbolic links

`-z`
: Compresses the transmitted data

Note: Trailing slashes count

When working with Rsync, you should pay particular attention to trailing slashes. A trailing slash after the directory denotes the *content* of the directory. No trailing slash denotes the *directory itself*.

15.3 Copying files and directories locally

The following description assumes that the current user has write permissions to the directory `/var/backup`. To copy a single file from one directory on your machine to another path, use the following command:

```
> rsync -avz backup.tar.xz /var/backup/
```

The file `backup.tar.xz` is copied to `/var/backup/`; the absolute path is `/var/backup/backup.tar.xz`.

Do not forget to add the *trailing slash* after the `/var/backup/` directory. If you do not insert the slash, the file `backup.tar.xz` is copied to `/var/backup` (file) *not* inside the directory `/var/backup/`!

Copying a directory is similar to copying a single file. The following example copies the directory `tux/` and its content into the directory `/var/backup/`:

```
> rsync -avz tux /var/backup/
```

Find the copy in the absolute path /var/backup/tux/.

15.4 Copying files and directories remotely

The Rsync tool is required on both machines. To copy files from or to remote directories requires an IP address or a domain name. A user name is optional if your current user names on the local and remote machine are the same.

To copy the file file.tar.xz from your local host to the remote host 192.168.1.1 with same users (being local and remote), use the following command:

```
> rsync -avz file.tar.xz  tux@192.168.1.1:
```

Depending on what you prefer, these commands are also possible and equivalent:

```
> rsync -avz file.tar.xz 192.168.1.1:~
> rsync -avz file.tar.xz 192.168.1.1:/home/tux
```

In all cases with standard configuration, you are prompted to enter your passphrase of the remote user. This command copies file.tar.xz to the home directory of user tux (normally /home/tux).

Copying a directory remotely is similar to copying a directory locally. The following example copies the directory tux/ and its content into the remote directory /var/backup/ on the 192.168.1.1 host:

```
> rsync -avz tux 192.168.1.1:/var/backup/
```

Assuming you have write permissions on the host 192.168.1.1, you can find the copy in the absolute path /var/backup/tux.

15.5 Configuring and using an rsync server

Rsync can run as a daemon (rsyncd) listening on default port 873 for incoming connections. This daemon can receive “copying targets”.

The following description explains how to create an Rsync server on a `jupiter` host with a *backup* target. This target can be used to store your backups. To create an Rsync server, do the following:

PROCEDURE 15.1: SETTING UP AN RSYNC SERVER

1. On jupiter, create a directory to store all your backup files. In this example, we use `/var/backup`:

   ```
   # mkdir /var/backup
   ```

2. Specify ownership. In this case, the directory is owned by user `tux` in group `users`:

   ```
   # chown tux.users /var/backup
   ```

3. Configure the rsyncd daemon.
   We separate the configuration file into a main file and certain “modules” which hold your backup target. This makes it easier to add additional targets later. Global values can be stored in `/etc/rsyncd.d/*.inc` files, whereas your modules are placed in `/etc/rsyncd.d/*.conf` files:

   a. Create a directory `/etc/rsyncd.d/`:

      ```
      # mkdir /etc/rsyncd.d/
      ```

   b. In the main configuration file `/etc/rsyncd.conf`, add the following lines:

      ```
      # rsyncd.conf main configuration file
      log file = /var/log/rsync.log
      pid file = /var/lock/rsync.lock

      &merge /etc/rsyncd.d 1
      &include /etc/rsyncd.d 2
      ```

      1 Merges global values from `/etc/rsyncd.d/*.inc` files into the main configuration file.

      2 Loads any modules (or targets) from `/etc/rsyncd.d/*.conf` files. These files should not contain any references to global values.

   c. Create your module (your backup target) in the file `/etc/rsyncd.d/backup.conf` with the following lines:

      ```
      # backup.conf: backup module
      ```

```
[backup] ❶
   uid = tux ❷
   gid = users ❷
   path = /var/backup ❸
   auth users = tux  ❹
   secrets file = /etc/rsyncd.secrets ❺
   comment = Our backup target
```

❶ The *backup* target. You can use any name you like. However, it is a good idea to name a target according to its purpose and use the same name in your `*.conf` file.

❷ Specifies the user name or group name that is used when the file transfer takes place.

❸ Defines the path to store your backups (from *Step 1*).

❹ Specifies a comma-separated list of allowed users. In its simplest form, it contains the user names that are allowed to connect to this module. In our case, only user `tux` is allowed.

❺ Specifies the path of a file that contains lines with user names and plain passwords.

d. Create the `/etc/rsyncd.secrets` file with the following content and replace *PASSPHRASE*:

```
# user:passwd
tux:PASSPHRASE
```

e. Make sure the file is only readable by `root`:

```
# chmod 0600 /etc/rsyncd.secrets
```

4. Start and enable the rsyncd daemon with:

```
# systemctl enable rsyncd
# systemctl start rsyncd
```

5. Test the access to your Rsync server:

```
> rsync jupiter::
```

You should see a response that looks like this:

```
backup          Our backup target
```

Otherwise, check your configuration file, firewall and network settings.

The above steps create an Rsync server that can now be used to store backups. The example also creates a log file listing all connections. This file is stored in `/var/log/rsyncd.log`. This is useful to debug your transfers.

To list the content of your backup target, use the following command:

```
> rsync -avz jupiter::backup
```

This command lists all files present in the directory `/var/backup` on the server. This request is also logged in the log file `/var/log/rsyncd.log`. To start an actual transfer, provide a source directory. Use `.` for the current directory. For example, the following command copies the current directory to your Rsync backup server:

```
> rsync -avz . jupiter::backup
```

By default, Rsync does not delete files and directories when it runs. To enable deletion, the additional option `--delete` must be stated. To ensure that no newer files are deleted, the option `--update` can be used instead. Any conflicts that arise must be resolved manually.

15.6 More information

Csync

Bidirectional file synchronization tool, see https://csync.org/.

RSnapshot

Creates incremental backups, see https://rsnapshot.org.

Unison

A file synchronization tool similar to CSync but with a graphical interface, see https://github.com/bcpierce00/unison.

Rear

A disaster recovery framework, see the *Administration Guide* of the SUSE Linux Enterprise High Availability, chapter *Disaster Recovery with Rear (Relax-and-Recover)* (https://documentation.suse.com/sle-ha/15/html/SLE-HA-all/cha-ha-rear.html).

II Booting a Linux system

16 Introduction to the boot process 220

17 UEFI (Unified Extensible Firmware Interface) 229

18 The boot loader GRUB 2 240

19 The `systemd` daemon 260

16 Introduction to the boot process

Revision History

2024-06-27

Booting a Linux system involves different components and tasks. After a firmware and hardware initialization process, which depends on the machine's architecture, the kernel is started by the boot loader GRUB 2. After this point, the boot process is controlled by the operating system and handled by `systemd`. `systemd` provides a set of “targets” that boot configurations for everyday usage, maintenance or emergencies.

16.1 Terminology

This chapter uses terms that can be interpreted ambiguously. To understand how they are used here, read the definitions below:

`init`

Two different processes are commonly named “init”:

- The `initramfs` process mounting the root file system
- The operating system process that starts all other processes that is executed from the real root file system

In both cases, the `systemd` program is taking care of this task. It is first executed from the `initramfs` to mount the root file system. When that has succeeded, it is re-executed from the root file system as the initial process. To avoid confusing these two `systemd` processes, we refer to the first process as *init on initramfs* and to the second one as *systemd*.

`initrd/initramfs`

An `initrd` (initial RAM disk) is an image file containing a root file system image which is loaded by the kernel and mounted from `/dev/ram` as the temporary root file system. Mounting this file system requires a file system driver.

Beginning with kernel 2.6.13, the initrd has been replaced by the `initramfs` (initial RAM file system), which does not require a file system driver to be mounted. SUSE Linux Enterprise Desktop exclusively uses an `initramfs`. However, since the `initramfs` is stored as `/boot/initrd`, it is often called "initrd". In this chapter we exclusively use the name `initramfs`.

16.2 The Linux boot process

The Linux boot process consists of several stages, each represented by a different component:

1. *Section 16.2.1, "The initialization and boot loader phase"*
2. *Section 16.2.2, "The kernel phase"*
3. *Section 16.2.3, "The init on initramfs phase"*
4. *Section 16.2.4, "The systemd phase"*

16.2.1 The initialization and boot loader phase

During the initialization phase, the machine's hardware is set up, and the devices are prepared. This process differs across hardware architectures.

SUSE Linux Enterprise Desktop uses the boot loader GRUB 2 on all architectures. Depending on the architecture and firmware, starting the GRUB 2 boot loader can be a multi-step process. The purpose of the boot loader is to load the kernel and the initial, RAM-based file system (initramfs). For more information about GRUB 2, refer to *Chapter 18, The boot loader GRUB 2*.

16.2.1.1 Initialization and boot loader phase on AArch64 and AMD64/Intel 64

After turning on the computer, the BIOS or the UEFI initializes the screen and keyboard, and tests the main memory. Up to this stage, the machine does not access any mass storage media. Subsequently, the information about the current date, time and the most important peripherals are loaded from the CMOS values. When the boot media and its geometry are recognized, the system control passes from the BIOS/UEFI to the boot loader.

On a machine equipped with a traditional BIOS, only code from the first physical 512-byte data sector (the Master Boot Record, MBR) of the boot disk can be loaded. Only a minimal GRUB 2 fits into the MBR. Its sole purpose is to load a GRUB 2 core image containing file system drivers from the gap between the MBR and the first partition (MBR partition table) or from the BIOS boot partition (GPT partition table). This image contains file system drivers and therefore is able to access `/boot` located on the root file system. `/boot` contains additional modules for GRUB 2 core as well as the kernel and the initramfs image. When it has access to this partition, GRUB 2 loads the kernel and the initramfs image into memory and hands control over to the kernel.

When booting a BIOS system from an encrypted file system that includes an encrypted `/boot` partition, you need to enter the password for decryption twice. It is first needed by GRUB 2 to decrypt `/boot` and then for `systemd` to mount the encrypted volumes.

On machines with UEFI the boot process is much simpler than on machines with a traditional BIOS. The firmware is able to read from a FAT formatted system partition of disks with a GPT partition table. This EFI system-partition (in the running system mounted as `/boot/efi`) holds enough space to host a fully fledged GRUB 2 which is directly loaded and executed by the firmware.

If the BIOS/UEFI supports network booting, it is also possible to configure a boot server that provides the boot loader. The system can then be booted via PXE. The BIOS/UEFI acts as the boot loader. It gets the boot image from the boot server and starts the system, independent of local hard disks.

16.2.1.2 Initialization and boot loader phase on IBM Z

On IBM Z the boot process must be initialized by a boot loader called **`zipl`** (z initial program load). Although **`zipl`** supports reading from multiple file systems, it does not support the SLE default file system (Btrfs) or booting from snapshots. SUSE Linux Enterprise Desktop therefore uses a two-stage boot process that ensures full Btrfs support at boot-time:

1. **`zipl`** boots from the partition `/boot/zipl`, which can be formatted with the Ext2, Ext3, Ext4, or XFS file system. This partition contains a minimal kernel and an initramfs that are loaded into memory. The initramfs contains a Btrfs driver (among others) and the boot loader GRUB 2. The kernel is started with a parameter `initgrub`, which tells it to start GRUB 2.

2. The kernel mounts the root file system, so `/boot` becomes accessible. Now GRUB 2 is started from the initramfs. It reads its configuration from `/boot/grub2/grub.cfg` and loads the final kernel and initramfs from `/boot`. The new kernel now gets loaded via Kexec.

16.2.2 The kernel phase

When the boot loader has passed on system control, the boot process is the same on all architectures. The boot loader loads both the kernel and an initial RAM-based file system (`initramfs`) into memory and the kernel takes over.

After the kernel has set up memory management and has detected the CPU type and its features, it initializes the hardware and mounts the temporary root file system from the memory that was loaded with the `initramfs`.

16.2.2.1 The `initramfs` file

`initramfs` (initial RAM file system) is a small cpio archive that the kernel can load into a RAM disk. It is located at `/boot/initrd`. It can be created with a tool called **`dracut`**—refer to **`man 8 dracut`** for details.

The `initramfs` provides a minimal Linux environment that enables the execution of programs before the actual root file system is mounted. This minimal Linux environment is loaded into memory by BIOS or UEFI routines and does not have specific hardware requirements other than sufficient memory. The `initramfs` archive must always provide an executable named `init` that executes the `systemd` daemon on the root file system for the boot process to proceed.

Before the root file system can be mounted and the operating system can be started, the kernel needs the corresponding drivers to access the device on which the root file system is located. These drivers may include special drivers for certain kinds of hard disks or even network drivers to access a network file system. The needed modules for the root file system are loaded by `init` on `initramfs`. After the modules are loaded, `udev` provides the `initramfs` with the needed devices. Later in the boot process, after changing the root file system, it is necessary to regenerate the devices. This is done by the `systemd` unit `systemd-udev-trigger.service`.

16.2.2.1.1 Regenerating the initramfs

Because the `initramfs` contains drivers, it needs to be updated whenever a new version of one of its drivers is available. This is done automatically when installing the package containing the driver update. YaST or zypper informs you about this by showing the output of the command that generates the `initramfs`. However, there are specific occasions when you need to regenerate an `initramfs` manually:

Adding drivers because of hardware changes

If you need to change hardware, for example, hard disks, and this hardware requires different drivers to be in the kernel at boot time, you must update the `initramfs` file.
Open or create `/etc/dracut.conf.d/10-DRIVER.conf` and add the following line (mind the leading blank space):

```
force_drivers+=" DRIVER1 "
```

Replace *`DRIVER1`* with the module name of the driver. If you need to add more than one driver, list them space-separated:

```
force_drivers+=" DRIVER1 DRIVER2 "
```

Proceed with *Procedure 16.1, "Generate an initramfs"*.

Moving system directories to a RAID or LVM

Whenever you move swap files, or system directories like `/usr` in a running system to a RAID or logical volume, you need to create an `initramfs` that contains support for software RAID or LVM drivers.
To do so, create the respective entries in `/etc/fstab` and mount the new entries (for example with **`mount -a`** and/or **`swapon -a`**).
Proceed with *Procedure 16.1, "Generate an initramfs"*.

Adding disks to an LVM group or Btrfs RAID containing the root file system

Whenever you add (or remove) a disk to a logical volume group or a Btrfs RAID containing the root file system, you need to create an `initramfs` that contains support for the enlarged volume. Follow the instructions at *Procedure 16.1, "Generate an initramfs"*.
Proceed with *Procedure 16.1, "Generate an initramfs"*.

Changing kernel variables

If you change the values of kernel variables via the **sysctl** interface by editing related files (/etc/sysctl.conf or /etc/sysctl.d/*.conf), the change will be lost on the next system reboot. Even if you load the values with **sysctl --system** at runtime, the changes are not saved into the initramfs file. You need to update it by proceeding as outlined in *Procedure 16.1, "Generate an initramfs"*.

Adding or removing swap devices, re-creating swap area

Whenever you add or remove a swap device, or re-create a swap area with a different UUID, update the initramfs as outlined in *Procedure 16.1, "Generate an initramfs"*. You may also need to update GRUB_CMDLINE_* variables that include the resume= option in /etc/default/grub, and then regenerate /boot/grub2/grub.cfg as outlined in *Section 18.2.1, "The file /boot/grub2/grub.cfg"*.

PROCEDURE 16.1: GENERATE AN INITRAMFS

Important

All commands in the following procedure need to be executed as the root user.

1. Enter your /boot directory:

```
# cd /boot
```

2. Generate a new initramfs file with **dracut**, replacing *MY_INITRAMFS* with a file name of your choice:

```
# dracut MY_INITRAMFS
```

   Alternatively, run **dracut -f** *FILENAME* to replace an existing init file.

3. (Skip this step if you ran **dracut -f** in the previous step.) Create a symbolic link from the initramfs file you created in the previous step to initrd:

```
#  ln -sf MY_INITRAMFS initrd
```

4. On the IBM Z architecture, additionally run **grub2-install**.

16.2.3 The init on initramfs phase

The temporary root file system mounted by the kernel from the `initramfs` contains the executable `systemd` (which is called `init` on `initramfs` in the following, also see *Section 16.1, “Terminology”*. This program performs all actions needed to mount the proper root file system. It provides kernel functionality for the needed file system and device drivers for mass storage controllers with `udev`.

The main purpose of `init` on `initramfs` is to prepare the mounting of and access to the real root file system. Depending on your system configuration, `init` on `initramfs` is responsible for the following tasks.

Loading kernel modules

Depending on your hardware configuration, special drivers may be needed to access the hardware components of your computer (the most important component being your hard disk). To access the final root file system, the kernel needs to load the proper file system drivers.

Providing block special files

The kernel generates device events depending on loaded modules. `udev` handles these events and generates the required special block files on a RAM file system in `/dev`. Without those special files, the file system and other devices would not be accessible.

Managing RAID and LVM setups

If you configured your system to hold the root file system under RAID or LVM, `init` on `initramfs` sets up LVM or RAID to enable access to the root file system later.

Managing the network configuration

If you configured your system to use a network-mounted root file system (mounted via NFS), `init` must make sure that the proper network drivers are loaded and that they are set up to allow access to the root file system.

If the file system resides on a network block device like iSCSI or SAN, the connection to the storage server is also set up by `init` on `initramfs`. SUSE Linux Enterprise Desktop supports booting from a secondary iSCSI target if the primary target is not available. .

Note: Handling of mount failures

If the root file system fails to mount from within the boot environment, it must be checked and repaired before the boot can continue. The file system checker will be automatically started for Ext3 and Ext4 file systems. The repair process is not automated for XFS and Btrfs file systems, and the user is presented with information describing the options available to repair the file system. When the file system has been successfully repaired, exiting the boot environment will cause the system to retry mounting the root file system. If successful, the boot will continue normally.

16.2.3.1 The init on initramfs phase in the installation process

When `init` on `initramfs` is called during the initial boot as part of the installation process, its tasks differ from those mentioned above. The installation system also does not start `systemd` from `initramfs`—these tasks are performed by **`linuxrc`**.

Finding the installation medium

When starting the installation process, your machine loads an installation kernel and a special `init` containing the YaST installer. The YaST installer is running in a RAM file system and needs to have information about the location of the installation medium to access it for installing the operating system.

Initiating hardware recognition and loading appropriate kernel modules

As mentioned in *Section 16.2.2.1, "The `initramfs` file"*, the boot process starts with a minimum set of drivers that can be used with most hardware configurations. On AArch64, POWER, and AMD64/Intel 64 machines, **`linuxrc`** starts an initial hardware scanning process that determines the set of drivers suitable for your hardware configuration. On IBM Z, a list of drivers and their parameters needs to be provided, for example, via linuxrc or a parmfile. These drivers are used to generate a custom `initramfs` that is needed to boot the system. If the modules are not needed for boot but for coldplug, the modules can be loaded with `systemd`; for more information, see *Section 19.6.4, "Loading kernel modules"*.

Loading the installation system

When the hardware is properly recognized, the appropriate drivers are loaded. The `udev` program creates the special device files and **`linuxrc`** starts the installation system with the YaST installer.

Starting YaST

Finally, **`linuxrc`** starts YaST, which starts the package installation and the system configuration.

16.2.4 The systemd phase

After the "real" root file system has been found, it is checked for errors and mounted. If this is successful, the `initramfs` is cleaned and the `systemd` daemon on the root file system is executed. `systemd` is Linux's system and service manager. It is the parent process that is started as PID 1 and acts as an init system which brings up and maintains user space services. See *Chapter 19, The `systemd` daemon* for details.

17 UEFI (Unified Extensible Firmware Interface)

Revision History

2025-06-06

UEFI (Unified Extensible Firmware Interface) is the interface between the firmware that comes with the system hardware, all the hardware components of the system, and the operating system.

UEFI is becoming more and more available on PC systems and thus is replacing the traditional PC-BIOS. UEFI, for example, properly supports 64-bit systems and offers secure booting (“Secure Boot”, firmware version 2.3.1c or better required), which is one of its most important features. Lastly, with UEFI a standard firmware will become available on all x86 platforms.

UEFI additionally offers the following advantages:

- Booting from large disks (over 2 TiB) with a GUID Partition Table (GPT).
- CPU-independent architecture and drivers.
- Flexible pre-OS environment with network capabilities.
- CSM (Compatibility Support Module) to support booting legacy operating systems via a PC-BIOS-like emulation.

For more information, see https://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface. The following sections are not meant as a general UEFI overview; these are only hints about how certain features are implemented in SUSE Linux Enterprise Desktop.

17.1 Secure boot

In the world of UEFI, securing the bootstrapping process means establishing a chain of trust. The “platform” is the root of this chain of trust; in the context of SUSE Linux Enterprise Desktop, the mainboard and the on-board firmware could be considered the “platform”. In other words, it is the hardware vendor, and the chain of trust flows from that hardware vendor to the component manufacturers, the OS vendors, etc.

The trust is expressed via public key cryptography. The hardware vendor puts a so-called Platform Key (PK) into the firmware, representing the root of trust. The trust relationship with operating system vendors and others is documented by signing their keys with the Platform Key.

Finally, security is established by requiring that no code will be executed by the firmware unless it has been signed by one of these “trusted” keys—be it an OS boot loader, a driver located in the flash memory of certain PCI Express card or on disk, or be it an update of the firmware itself.

To use Secure Boot, you need to have your OS loader signed with a key trusted by the firmware, and you need the OS loader to verify that the kernel it loads can be trusted.

Key Exchange Keys (KEK) can be added to the UEFI key database. This way, you can use other certificates, if they are signed with the private part of the PK.

17.1.1 Implementation on SUSE Linux Enterprise Desktop

Microsoft’s Key Exchange Key (KEK) is installed by default.

Note: GUID partitioning table (GPT) required

The Secure Boot feature is enabled by default on UEFI/x86_64 installations. You can find the *Enable Secure Boot Support* option in the *Boot Code Options* tab of the *Boot Loader Settings* dialog. It supports booting when the secure boot is activated in the firmware, while making it possible to boot when it is deactivated.

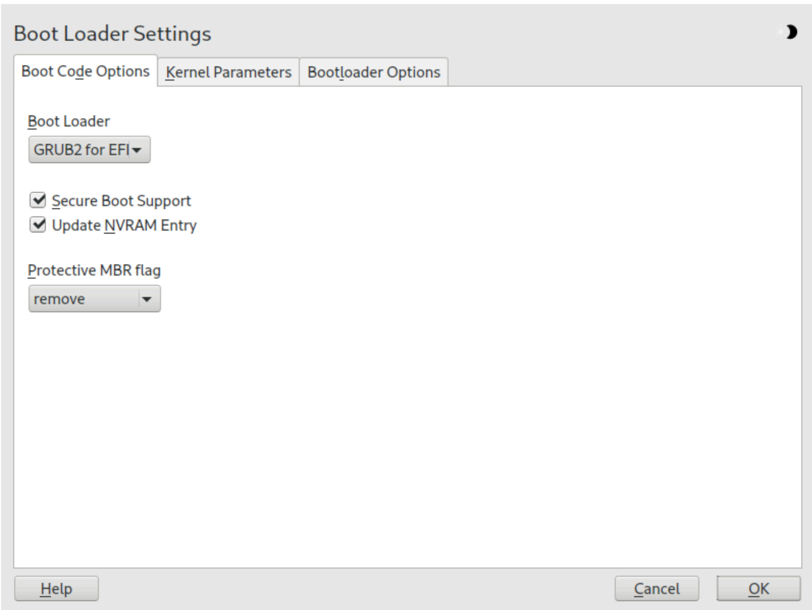


FIGURE 17.1: SECURE BOOT SUPPORT

The Secure Boot feature requires that a GUID Partitioning Table (GPT) replaces the old partitioning with a Master Boot Record (MBR). If YaST detects EFI mode during the installation, it will try to create a GPT partition. UEFI expects to find the EFI programs on a FAT-formatted EFI System Partition (ESP).

Supporting UEFI Secure Boot requires having a boot loader with a digital signature that the firmware recognizes as a trusted key. That key is trusted by the firmware a priori, without requiring any manual intervention.

There are two ways of getting there. One is to work with hardware vendors to have them endorse a SUSE key, which SUSE then signs the boot loader with. The other way is to go through Microsoft's Windows Logo Certification program to have the boot loader certified and have Microsoft recognize the SUSE signing key (that is, have it signed with their KEK). By now, SUSE got the loader signed by UEFI Signing Service (that is Microsoft in this case).

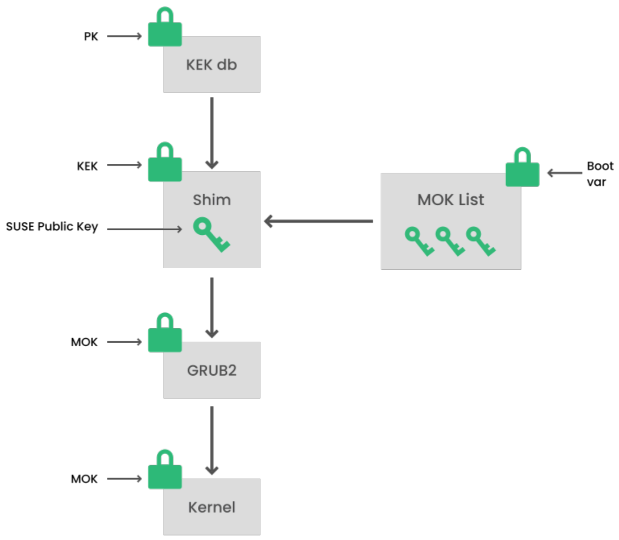


FIGURE 17.2: UEFI: SECURE BOOT PROCESS

At the implementation layer, SUSE uses the `shim` loader which is installed by default. It is a smart solution that avoids legal issues, and simplifies the certification and signing step considerably. The `shim` loader's job is to load a boot loader such as GRUB 2 and verify it; this boot loader in turn will load kernels signed by a SUSE key only.

There are two types of trusted users:

- First, those who hold the keys. The Platform Key (PK) allows almost everything. The Key Exchange Key (KEK) allows all a PK can except changing the PK.
- Second, anyone with physical access to the machine. A user with physical access can reboot the machine, and configure UEFI.

UEFI offers two types of variables to fulfill the needs of those users:

- The first is the so-called “Authenticated Variables”, which can be updated from both within the boot process (the so-called Boot Services Environment) and the running OS. This can be done only when the new value of the variable is signed with the same key that the old value of the variable was signed with. And they can only be appended to or changed to a value with a higher serial number.
- The second is the so-called “Boot Services Only Variables”. These variables are accessible to any code that runs during the boot process. After the boot process ends and before the OS starts, the boot loader must call the `ExitBootServices` call. After that, these variables are no longer accessible, and the OS cannot touch them.

UEFI key lists are of the first type, as this allows online updating, adding and blacklisting of keys, drivers and firmware fingerprints. It is the second type of variable, the “Boot Services Only Variable”, that helps to implement Secure Boot in a secure and open source-friendly manner, and thus compatible with GPLv3.

SUSE starts with `shim`—a small and simple EFI boot loader signed by SUSE and Microsoft.

This allows `shim` to load and execute.

`shim` then goes on to verify that the boot loader it wants to load is trusted. In a default situation `shim` will use an independent SUSE certificate embedded in its body. In addition, `shim` will allow to “enroll” additional keys, overriding the default SUSE key. In the following, we call them “Machine Owner Keys” or MOKs for short.

Next the boot loader will verify and then boot the kernel, and the kernel will do the same on the modules.

17.1.2 MOK (Machine Owner Key)

To replace specific kernels, drivers or other components that are part of the boot process, you need to use Machine Owner Keys (MOKs). The `mokutil` tool can help you to manage MOKs.

You can create a MOK enrollment request with `mokutil`. The request is stored in a UEFI runtime (RT) variable called `MokNew`. During the next boot, the `shim` boot loader detects `MokNew` and loads `MokManager`, which presents you with several options. You can use the *Enroll key from disk* and *Enroll hash from disk* options to add the key to the MokList. Use the *Enroll MOK* option to copy the key from the `MokNew` variable.

Enrolling a key from disk is normally done when the shim fails to load `grub2` and falls back to loading MokManager. As `MokNew` does not exist yet, you have the option of locating the key on the UEFI partition.

17.1.3 Booting a custom kernel

The following is based on https://en.opensuse.org/openSUSE:UEFI#Booting_a_custom_kernel.

Secure Boot does not prevent you from using a self-compiled kernel. You must sign it with your own certificate and make that certificate known to the firmware or MOK.

1. Create a custom X.509 key and certificate used for signing:

```
openssl req -new -x509 -newkey rsa:2048 -keyout key.asc \
  -out cert.pem -nodes -days 666 -subj "/CN=$USER/"
```

   For more information about creating certificates, see https://en.opensuse.org/openSUSE:UEFI_Image_File_Sign_Tools#Create_Your_Own_Certificate.

2. Package the key and the certificate as a PKCS#12 structure:

```
> openssl pkcs12 -export -inkey key.asc -in cert.pem \
  -name kernel_cert -out cert.p12
```

3. Generate an NSS database for use with **`pesign`**:

```
> certutil -d . -N
```

4. Import the key and the certificate contained in PKCS#12 into the NSS database:

```
> pk12util -d . -i cert.p12
```

5. “Bless” the kernel with the new signature using **`pesign`**:

```
> pesign -n . -c kernel_cert -i arch/x86/boot/bzImage \
  -o vmlinuz.signed -s
```

6. List the signatures on the kernel image:

```
> pesign -n . -S -i vmlinuz.signed
```

At that point, you can install the kernel in `/boot` as usual. Because the kernel now has a custom signature the certificate used for signing needs to be imported into the UEFI firmware or MOK.

7. Convert the certificate to the DER format for import into the firmware or MOK:

```
> openssl x509 -in cert.pem -outform der -out cert.der
```

8. Copy the certificate to the ESP for easier access:

```
> sudo cp cert.der /boot/efi/
```

9. Use **`mokutil`** to launch the MOK list automatically.

- a. Import the certificate to MOK:

  ```
  > mokutil --root-pw --import cert.der
  ```

  The `--root-pw` option enables usage of the `root` user directly.

  b. Check the list of certificates that are prepared to be enrolled:

  ```
  > mokutil --list-new
  ```

  c. Reboot the system; `shim` should launch MokManager. You need to enter the `root` password to confirm the import of the certificate to the MOK list.

  d. Check if the newly imported key was enrolled:

  ```
  > mokutil --list-enrolled
  ```

- a. Alternatively, this is the procedure to launch MOK manually:
  Reboot

  b. In the GRUB 2 menu press the '`c`' key.

  c. Type:

  ```
  chainloader $efibootdir/MokManager.efi
  boot
  ```

d. Select *Enroll key from disk*.

e. Navigate to the `cert.der` file and press **Enter** .

f. Follow the instructions to enroll the key. Normally this should be pressing "0" and then "y" to confirm.
Alternatively, the firmware menu may provide ways to add a new key to the Signature Database.

17.1.4 Using non-inbox drivers

There is no support for adding non-inbox drivers (that is, drivers that do not come with SUSE Linux Enterprise Desktop) during installation with Secure Boot enabled. The signing key used for SolidDriver/PLDP is not trusted by default.

It is possible to install third party drivers during installation with Secure Boot enabled in two different ways. In both cases:

- Add the needed keys to the firmware database via firmware/system management tools before the installation. This option depends on the specific hardware you are using. Consult your hardware vendor for more information.
- Use a bootable driver ISO from https://drivers.suse.com/ or your hardware vendor to enroll the needed keys in the MOK list at first boot.

To use the bootable driver ISO to enroll the driver keys to the MOK list, follow these steps:

1. Burn the ISO image above to an empty CD/DVD medium.
2. Start the installation using the new CD/DVD medium, having the standard installation media at hand or a URL to a network installation server.
If doing a network installation, enter the URL of the network installation source on the boot command line using the `install=` option.
If doing installation from optical media, the installer will first boot from the driver kit and then ask to insert the first installation disk of the product.
3. An initrd containing updated drivers will be used for installation.

For more information, see https://drivers.suse.com/doc/Usage/Secure_Boot_Certificate.html .

17.1.5 Features and limitations

When booting in Secure Boot mode, the following features apply:

- Installation to UEFI default boot loader location, a mechanism to keep or restore the EFI boot entry.
- Reboot via UEFI.
- Xen hypervisor will boot with UEFI when there is no legacy BIOS to fall back to.
- UEFI IPv6 PXE boot support.
- UEFI video mode support, the kernel can retrieve video mode from UEFI to configure KMS mode with the same parameters.
- UEFI booting from USB devices is supported.
- Since SUSE Linux Enterprise Server 15 SP3, Kexec and Kdump are supported in Secure Boot mode.

When booting in Secure Boot mode, the following limitations apply:

- To ensure that Secure Boot cannot be easily circumvented, certain kernel features are disabled when running under Secure Boot.
- Boot loader, kernel, and kernel modules must be signed.
- Hibernation (suspend on disk) is disabled.
- Access to `/dev/kmem` and `/dev/mem` is not possible, not even as root user.
- Access to the I/O port is not possible, not even as root user. All X11 graphical drivers must use a kernel driver.
- PCI BAR access through sysfs is not possible.
- `custom_method` in ACPI is not available.
- debugfs for asus-wmi module is not available.
- the `acpi_rsdp` parameter does not have any effect on the kernel.

17.2 The Secure Boot Revocation List

The UEFI Secure Boot Revocation List, also known as `dbx` (Secure Boot Forbidden Signature Database), is a critical security component of a computer's UEFI firmware. It enhances the system security by preventing the loading and execution of untrusted software during the boot process. `dbx` is important because it does the following:

- **Prevents boot-time malware:** `dbx` stops malicious code from being loaded and executed before the operating system even starts.
- **Maintains a chain of trust:** Each component verifies the next one in the boot sequence. `dbx` ensures that any component in this chain that has been compromised is immediately blocked.
- **Protects against rollback attacks:** Helps prevent attackers from rolling back firmware or boot-loaders to older, vulnerable versions.
- **Enhanced security:** Updating `dbx` is important to keep your system protected as new vulnerabilities are discovered. Failing to update it can leave your system exposed to known exploits.

17.2.1 How to apply an online Revocation List update

PREREQUISITES

- Secure boot is enabled on your system.
- Your system can access the Internet for updates.

1. Check the current version of the Revocation List:

```
> fwupdmgr get-devices
      LENOVO 21AAS05L00
      │
      ├─11th Gen Intel Core™ i7-11800H @ 2.30GHz:
      │     Device ID:          4bde70ba4e39b28f9eab1628f9dd6e6244c03027
      │     Current version:    0x00000052
      │     Vendor:             Intel
      │     GUIDs:              a6bd4ca5-75a6-5796-b564-66b5cab1b11b ← CPUID
\PRO_0&FAM_06&MOD_8D
      │                         d9dd5e77-df17-5bab-b5ec-22827598bfed ← CPUID
\PRO_0&FAM_06&MOD_8D&STP_1
      │     Device Flags:       • Internal device
```

```
      |     Device Requests:     • Message
```

2. Enable the LVFS (Linux Vendor Firmware Service) Revocation List repository:

```
> fwupdmgr enable-remote lvfs
```

3. Refresh the repository metadata:

```
> fwupdmgr refresh
```

4. Apply the Revocation List update:

```
> fwupdmgr update
```

5. To verify after reboot, check the current version:

```
> fwupdmgr get-devices
```

17.2.2 How to apply an online Revocation List update

For an offline revocation list update, you can update the secure Boot revocation list from SUSE Linux Enterprise Desktop so that secure boot prevents known security issues. This procedure is safe and ensures that the update does not prevent your system from booting.

1. Check the current version of the Revocation List:

```
> fwupdmgr get-devices
        LENOVO 21AAS05L00
        |
        ├─11th Gen Intel Core™ i7-11800H @ 2.30GHz:
        |     Device ID:          4bde70ba4e39b28f9eab1628f9dd6e6244c03027
        |     Current version:    0x00000052
        |     Vendor:             Intel
        |     GUIDs:              a6bd4ca5-75a6-5796-b564-66b5cab1b11b ← CPUID
\PRO_0&FAM_06&MOD_8D
        |                         d9dd5e77-df17-5bab-b5ec-22827598bfed ← CPUID
\PRO_0&FAM_06&MOD_8D&STP_1
        |     Device Flags:       • Internal device
        |     Device Requests:    • Message
```

2. List the updates available from SUSE Linux Enterprise Desktop:

```
> ls /usr/share/dbxtool/
```

3. Choose the most recent update file for your architecture. For example, `DBXUpdate-date-architecture.cab`.

4. Install the selected update file:

```
> fwupdmgr install /usr/share/dbxtool/DBXUpdate-date-architecture.cab
```

5. To verify after reboot, check the current version:

```
> fwupdmgr get-devices
```

17.3 More information

- https://uefi.org — UEFI home page where you can find the current UEFI specifications.
- Blog posts by Olaf Kirch and Vojtěch Pavlík (the chapter above is heavily based on these posts):
  - https://www.suse.com/c/uefi-secure-boot-plan/
  - https://www.suse.com/c/uefi-secure-boot-overview/
  - https://www.suse.com/c/uefi-secure-boot-details/
- https://en.opensuse.org/openSUSE:UEFI —UEFI with openSUSE.

18 The boot loader GRUB 2

Revision History

2025-05-07

This chapter describes how to configure GRUB 2, the boot loader used in SUSE® Linux Enterprise Desktop. It is the successor to the traditional GRUB boot loader—now called “GRUB Legacy”. GRUB 2 has been the default boot loader in SUSE® Linux Enterprise Desktop since version 12. A YaST module is available for configuring the most important settings. The boot procedure as a whole is outlined in *Chapter 16, Introduction to the boot process*. For details on Secure Boot support for UEFI machines, see *Chapter 17, UEFI (Unified Extensible Firmware Interface)*.

18.1 Main differences between GRUB legacy and GRUB 2

- The configuration is stored in different files.
- More file systems are supported (for example, Btrfs).
- Can directly read files stored on LVM or RAID devices.
- The user interface can be translated and altered with themes.
- Includes a mechanism for loading modules to support additional features, such as file systems, etc.
- Automatically searches for and generates boot entries for other kernels and operating systems, such as Windows.
- Includes a minimal Bash-like console.

18.2 Configuration file structure

The configuration of GRUB 2 is based on the following files:

`/boot/grub2/grub.cfg`

: This file contains the configuration of the GRUB 2 menu items. It replaces `menu.lst` used in GRUB Legacy. `grub.cfg` should not be edited—it is automatically generated by the command **`grub2-mkconfig -o /boot/grub2/grub.cfg`**.

`/boot/grub2/custom.cfg`

: This optional file is directly sourced by `grub.cfg` at boot time and can be used to add custom items to the boot menu. Starting with SUSE Linux Enterprise Desktop 12 SP2 these entries are also parsed when using **`grub2-once`**.

`/etc/default/grub`

: This file controls the user settings of GRUB 2 and normally includes additional environmental settings such as backgrounds and themes.

Scripts under `/etc/grub.d/`

: The scripts in this directory are read during execution of the command **`grub2-mkconfig -o /boot/grub2/grub.cfg`**. Their instructions are integrated into the main configuration file `/boot/grub/grub.cfg`.

`/etc/sysconfig/bootloader`

: This configuration file holds certain basic settings like the boot loader type and whether to enable UEFI Secure Boot support.

`/boot/grub2/x86_64-efi`, `/boot/grub2/power-ieee1275`

: These configuration files contain architecture-specific options.

GRUB 2 can be controlled in multiple ways. Boot entries from an existing configuration can be selected from the graphical menu (splash screen). The configuration is loaded from the file `/boot/grub2/grub.cfg` which is compiled from other configuration files (see below). All GRUB 2 configuration files are considered system files, and you need `root` privileges to edit them.

Note: Activating configuration changes

After having manually edited GRUB 2 configuration files, you need to run **`grub2-mkconfig -o /boot/grub2/grub.cfg`** to activate the changes. However, this is not necessary when changing the configuration with YaST, because YaST automatically runs this command.

18.2.1 The file `/boot/grub2/grub.cfg`

The graphical splash screen with the boot menu is based on the GRUB 2 configuration file `/boot/grub2/grub.cfg`, which contains information about all partitions or operating systems that can be booted by the menu.

Every time the system is booted, GRUB 2 loads the menu file directly from the file system. For this reason, GRUB 2 does not need to be re-installed after changes to the configuration file. `grub.cfg` is automatically rebuilt with kernel installations or removals.

`grub.cfg` is compiled from the file `/etc/default/grub` and scripts found in the `/etc/grub.d/` directory when running the command **`grub2-mkconfig -o /boot/grub2/grub.cfg`**. Therefore you should never edit the file manually. Instead, edit the related source files or use the YaST *Boot Loader* module to modify the configuration as described in *Section 18.3, "Configuring the boot loader with YaST"*.

18.2.2 The file `/etc/default/grub`

More general options of GRUB 2 belong in this file, such as the time the menu is displayed, or the default OS to boot. To list all available options, see the output of the following command:

```
> grep "export GRUB_DEFAULT" -A50 /usr/sbin/grub2-mkconfig | grep GRUB_
```

You can introduce custom variables and use them later in the scripts found in the `/etc/grub.d` directory.

After having edited `/etc/default/grub`, update the main configuration file with **`grub2-mkconfig -o /boot/grub2/grub.cfg`**.

Note: Scope

All options specified in this file are general options that affect all boot entries. Options specific to a Xen hypervisor include the `_XEN_` substring.

Important: Escaping inner quotes

More complex options with spaces require quoting so that they are processed as one option. Such inner quotes need to be correctly escaped, for example:

```
GRUB_CMDLINE_LINUX_XEN="debug loglevel=9 log_buf_len=5M \"ddebug_query=file
 drivers/xen/xen-acpi-processor.c +p\""
```

`GRUB_DEFAULT`

Sets the boot menu entry that is booted by default. Its value can be a numeric value, the complete name of a menu entry, or “saved”.

`GRUB_DEFAULT=2` boots the third (counted from zero) boot menu entry.

`GRUB_DEFAULT="2>0"` boots the first submenu entry of the third top-level menu entry.

`GRUB_DEFAULT="Example boot menu entry"` boots the menu entry with the title “Example boot menu entry”.

`GRUB_DEFAULT=saved` boots the entry specified by the **`grub2-once`** or **`grub2-set-default`** commands. While **`grub2-reboot`** sets the default boot entry for the next reboot only, **`grub2-set-default`** sets the default boot entry until changed. **`grub2-editenv list`** lists the next boot entry.

`GRUB_HIDDEN_TIMEOUT`

Waits the specified number of seconds for the user to press a key. During the period no menu is shown unless the user presses a key. If no key is pressed during the time specified, the control is passed to `GRUB_TIMEOUT`. `GRUB_HIDDEN_TIMEOUT=0` first checks whether **Shift** is pressed and shows the boot menu if yes, otherwise immediately boots the default menu entry. This is the default when only one bootable OS is identified by GRUB 2.

`GRUB_HIDDEN_TIMEOUT_QUIET`

If `false` is specified, a countdown timer is displayed on a blank screen when the `GRUB_HIDDEN_TIMEOUT` feature is active.

`GRUB_TIMEOUT`

Time period in seconds the boot menu is displayed before automatically booting the default boot entry. If you press a key, the timeout is cancelled and GRUB 2 waits for you to make the selection manually. `GRUB_TIMEOUT=-1` causes the menu to be displayed until you select the boot entry manually.

`GRUB_CMDLINE_LINUX`

Entries on this line are added at the end of the boot entries for normal and recovery modes. Use it to add kernel parameters to the boot entry.

`GRUB_CMDLINE_LINUX_DEFAULT`

Same as `GRUB_CMDLINE_LINUX` but the entries are appended in the normal mode only.

`GRUB_CMDLINE_LINUX_RECOVERY`

Same as `GRUB_CMDLINE_LINUX` but the entries are appended in the recovery mode only.

`GRUB_CMDLINE_LINUX_XEN_REPLACE`

This entry replaces the `GRUB_CMDLINE_LINUX` parameters for all Xen boot entries.

`GRUB_CMDLINE_LINUX_XEN_REPLACE_DEFAULT`

Same as `GRUB_CMDLINE_LINUX_XEN_REPLACE` but it only replaces parameters of `GRUB_CMDLINE_LINUX_DEFAULT`.

`GRUB_CMDLINE_XEN`

These entries are passed to the Xen hypervisor Xen menu entries for normal and recovery modes. For example:

```
GRUB_CMDLINE_XEN="loglvl=all guest_loglvl=all"
```

Tip: Xen hypervisor options

Find a complete list of Xen hypervisor options in https://xenbits.xen.org/docs/unstable/misc/xen-command-line.html

`GRUB_CMDLINE_XEN_DEFAULT`

Same as `GRUB_CMDLINE_XEN` but the entries are appended in the normal mode only.

`GRUB_TERMINAL`

Enables and specifies an input/output terminal device. Can be `console` (PC BIOS and EFI consoles), `serial` (serial terminal), `ofconsole` (Open Firmware console), or the default `gfxterm` (graphics-mode output). It is also possible to enable more than one device by quoting the required options, for example, `GRUB_TERMINAL="console serial"`.

`GRUB_GFXMODE`

The resolution used for the `gfxterm` graphical terminal. You can only use modes supported by your graphics card (VBE). The default is 'auto', which tries to select a preferred resolution. You can display the screen resolutions available to GRUB 2 by typing **`videoinfo`** in the GRUB 2 command line. The command line is accessed by typing `C` when the GRUB 2 boot menu screen is displayed.

You can also specify a color depth by appending it to the resolution setting, for example, `GRUB_GFXMODE=1280x1024x24`.

`GRUB_BACKGROUND`

Set a background image for the `gfxterm` graphical terminal. The image must be a file readable by GRUB 2 at boot time, and it must end with the `.png`, `.tga`, `.jpg`, or `.jpeg` suffix. If necessary, the image is scaled to fit the screen.

`GRUB_DISABLE_OS_PROBER`

If this option is set to `true`, automatic searching for other operating systems is disabled. Only the kernel images in `/boot/` and the options from your own scripts in `/etc/grub.d/` are detected.

`SUSE_BTRFS_SNAPSHOT_BOOTING`

If this option is set to `true`, GRUB 2 can boot directly into Snapper snapshots. For more information, see *Section 10.3, "System rollback by booting from snapshots"*.

For a complete list of options, see the GNU GRUB manual (https://www.gnu.org/software/grub/manual/grub/grub.html#Simple-configuration).

18.2.3 Scripts in `/etc/grub.d`

The scripts in this directory are read during execution of the command **`grub2-mkconfig -o /boot/grub2/grub.cfg`**. Their instructions are incorporated into `/boot/grub2/grub.cfg`. The order of menu items in `grub.cfg` is determined by the order in which the files in this directory are run. Files with a leading numeral are executed first, beginning with the lowest number.

`00_header` is run before `10_linux`, which would run before `40_custom`. If files with alphabetic names are present, they are executed after the numerically named files. Only executable files generate output to `grub.cfg` during execution of **`grub2-mkconfig`**. By default all files in the `/etc/grub.d` directory are executable.

Tip: Persistent custom content in `grub.cfg`

Because `/boot/grub2/grub.cfg` is recompiled each time **`grub2-mkconfig`** is run, any custom content is lost. To insert your lines directly into `/boot/grub2/grub.cfg` without losing them after **`grub2-mkconfig`** is run, insert them between

```
### BEGIN /etc/grub.d/90_persistent ###
```

and

```
### END /etc/grub.d/90_persistent ###
```

The `90_persistent` script ensures that such content is preserved.

A list of the most important scripts follows:

`00_header`
: Sets environmental variables such as system file locations, display settings, themes and previously saved entries. It also imports preferences stored in the `/etc/default/grub`. Normally you do not need to make changes to this file.

`10_linux`
: Identifies Linux kernels on the root device and creates relevant menu entries. This includes the associated recovery mode option if enabled. Only the latest kernel is displayed on the main menu page, with additional kernels included in a submenu.

`30_os-prober`
: This script uses **`os-prober`** to search for Linux and other operating systems and places the results in the GRUB 2 menu. There are sections to identify specific other operating systems, such as Windows or macOS.

`40_custom`
: This file provides a simple way to include custom boot entries into `grub.cfg`. Make sure that you do not change the `exec tail -n +3 $0` part at the beginning.

The processing sequence is set by the preceding numbers with the lowest number being executed first. If scripts are preceded by the same number the alphabetical order of the complete name decides the order.

Tip: `/boot/grub2/custom.cfg`

If you create `/boot/grub2/custom.cfg` and fill it with content, it is automatically included into `/boot/grub2/grub.cfg` right after `40_custom` at boot time.

18.2.4 Mapping between BIOS drives and Linux devices

In GRUB Legacy, the `device.map` configuration file was used to derive Linux device names from BIOS drive numbers. The mapping between BIOS drives and Linux devices cannot always be guessed correctly. For example, GRUB Legacy would get a wrong order if the boot sequence of IDE and SCSI drives is exchanged in the BIOS configuration.

GRUB 2 avoids this problem by using device ID strings (UUIDs) or file system labels when generating `grub.cfg`. GRUB 2 utilities create a temporary device map on the fly, which is normally sufficient, particularly for single-disk systems.

However, if you need to override the GRUB 2's automatic device mapping mechanism, create your custom mapping file `/boot/grub2/device.map`. The following example changes the mapping to make `DISK 3` the boot disk. GRUB 2 partition numbers start with `1` and not with `0` as in GRUB 2 Legacy.

```
(hd0)  /dev/disk-by-id/DISK3 ID
(hd1)  /dev/disk-by-id/DISK1 ID
(hd2)  /dev/disk-by-id/DISK2 ID
```

18.2.5 Editing menu entries during the boot procedure

Being able to directly edit menu entries is useful when the system does not boot anymore because of a faulty configuration. It can also be used to test new settings without altering the system configuration.

1. In the graphical boot menu, select the entry you want to edit with the arrow keys.

2. Press E to open the text-based editor.

3. Use the arrow keys to move to the line you want to edit.

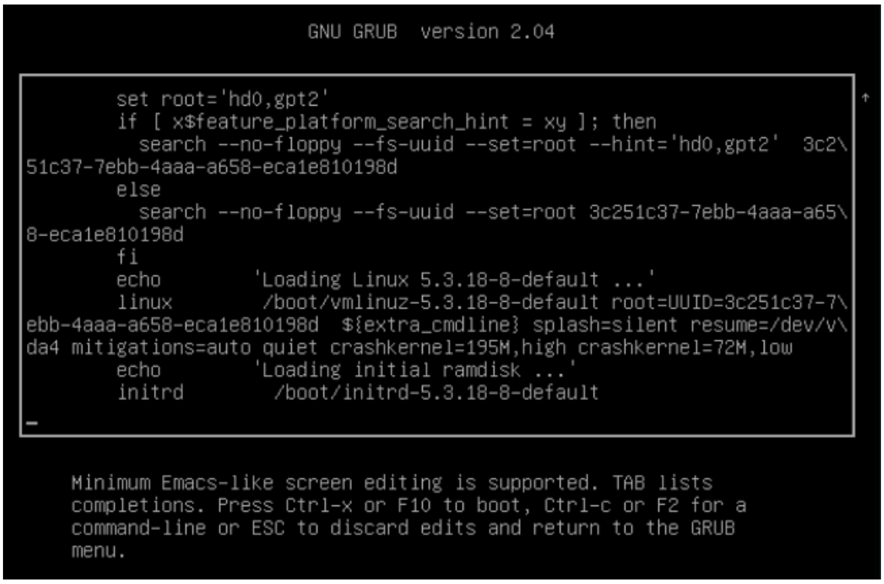


FIGURE 18.1: GRUB 2 BOOT EDITOR

Now you have two options:

a. Add space-separated parameters to the end of the line starting with `linux` or `linuxefi` to edit the kernel parameters. A complete list of parameters is available at https://en.opensuse.org/Linuxrc.

b. Or edit the general options to change, for example, the kernel version. The →| key suggests all possible completions.

4. Press F10 to boot the system with the changes you made or press Esc to discard your edits and return to the GRUB 2 menu.

Changes made this way only apply to the current boot process and are not saved permanently.

Important: Keyboard layout during the boot procedure

The US keyboard layout is the only one available when booting. See *Book “Deployment Guide”, Chapter 9 “Troubleshooting”, Section 9.3 “Booting from installation media fails”, US keyboard layout*.

Note: Boot loader on the installation media

The Boot Loader of the installation media on systems with a traditional BIOS is still GRUB Legacy. To add boot parameters, select an entry and start typing. Additions you make to the installation boot entry are permanently saved in the installed system.

18.2.6 Setting a boot password

Even before the operating system is booted, GRUB 2 enables access to file systems. Users without root permissions can access files in your Linux system to which they have no access after the system is booted. To block this kind of access or to prevent users from booting certain menu entries, set a boot password.

Important: Booting requires a password

If set, the boot password is required on every boot, which means the system does not boot automatically.

Proceed as follows to set a boot password. Alternatively use YaST (*Protect Boot Loader with Password*).

1. Encrypt the password using **`grub2-mkpasswd-pbkdf2`**:

```
> sudo grub2-mkpasswd-pbkdf2
Password: ****
Reenter password: ****
PBKDF2 hash of your password is grub.pbkdf2.sha512.10000.9CA4611006FE96BC77A...
```

2. Paste the resulting string into the file `/etc/grub.d/40_custom` together with the **`set superusers`** command.

```
set superusers="root"
password_pbkdf2 root grub.pbkdf2.sha512.10000.9CA4611006FE96BC77A...
```

3. To import the changes into the main configuration file, run:

```
> sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

After you reboot, GRUB 2 prompts you for a user name and a password when trying to boot a menu entry. Enter `root` and the password you typed during the **`grub2-mkpasswd-pbkdf2`** command. If the credentials are correct, the system boots the selected boot entry.

For more information, see https://www.gnu.org/software/grub/manual/grub/grub.html#Security.

18.2.7 Authorized access to boot menu entries

You can configure GRUB 2 to allow access to boot menu entries depending on the level of authorization. You can configure multiple user accounts protected with passwords and assign them access to different menu entries. To configure authorization in GRUB 2, follow these steps:

1. Create and encrypt one password for each user account you want to use in GRUB 2. Use the **`grub2-mkpasswd-pbkdf2`** command as described in *Section 18.2.6, “Setting a boot password”*.
2. Delete the file `/etc/grub.d/10_linux`. This prevents outputting the default GRUB 2 menu entries.
3. Edit the `/boot/grub2/custom.cfg` file and add custom menu entries manually. The following template is an example, adjust it to better match your use case:

```
set superusers=admin
password admin ADMIN_PASSWORD
password maintainer MAINTAINER_PASSWORD

menuentry 'Operational mode' {
  insmod ext2
  set root=hd0,1
  echo 'Loading Linux ...'
  linux /boot/vmlinuz root=/dev/vda1 $GRUB_CMDLINE_LINUX_DEFAULT $GRUB_CMDLINE_LINUX
 mode=operation
  echo 'Loading Initrd ...'
  initrd /boot/initrd
}

menuentry 'Maintenance mode' --users maintainer {
  insmod ext2
  set root=hd0,1
  echo 'Loading Linux ...'
  linux /boot/vmlinuz root=/dev/vda1 $GRUB_CMDLINE_LINUX_DEFAULT $GRUB_CMDLINE_LINUX
 mode=maintenance
  echo 'Loading Initrd ...'
```

```
  initrd /boot/initrd
}
```

4. Import the changes into the main configuration file:

```
> sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

In the above example:

- The GRUB 2 menu has two entries, *Operational mode* and *Maintenance mode*.
- If no user is specified, both boot menu entries are accessible, but no one can access GRUB 2 command line or edit existing menu entries.
- `admin` user can access GRUB 2 command line and edit existing menu entries.
- `maintenance` user can select the recovery menu item.

18.3 Configuring the boot loader with YaST

The easiest way to configure general options of the boot loader in your SUSE Linux Enterprise Desktop system is to use the YaST module. In the *YaST Control Center*, select *System* › *Boot Loader*. The module shows the current boot loader configuration of your system and allows you to make changes.

Use the *Boot Code Options* tab to view and change settings related to type, location and advanced loader settings. You can choose whether to use GRUB 2 in standard or EFI mode.

Important: EFI systems require GRUB2-EFI

If you have an EFI system you can only install GRUB2-EFI, otherwise your system is no longer bootable.

Important: Reinstalling the boot loader

To reinstall the boot loader, make sure to change a setting in YaST and then change it back. For example, to reinstall GRUB2-EFI, select *GRUB2* first and then immediately switch back to *GRUB2-EFI*.

Otherwise, the boot loader may only be partially reinstalled.

Note: Custom boot loader

To use a boot loader other than the ones listed, select *Do Not Install Any Boot Loader*. Read the documentation of your boot loader carefully before choosing this option.

18.3.1 Boot loader location and boot code options

The default location of the boot loader depends on the partition setup and is either the Master Boot Record (MBR) or the boot sector of the `/` partition. To modify the location of the boot loader, follow these steps:

PROCEDURE 18.1: CHANGING THE BOOT LOADER LOCATION

1. Select the *Boot Code Options* tab and then choose one of the following options for *Boot Loader Location*:

   Boot from Master Boot Record

   This installs the boot loader in the MBR of the disk containing the directory `/boot`. Usually this will be the disk mounted to `/`, but if `/boot` is mounted to a separate partition on a different disk, the MBR of that disk will be used.

   Boot from Root Partition

   This installs the boot loader in the boot sector of the `/` partition.

   Custom Root Partition

   Use this option to specify the location of the boot loader manually.

2. Click *OK* to apply the changes.

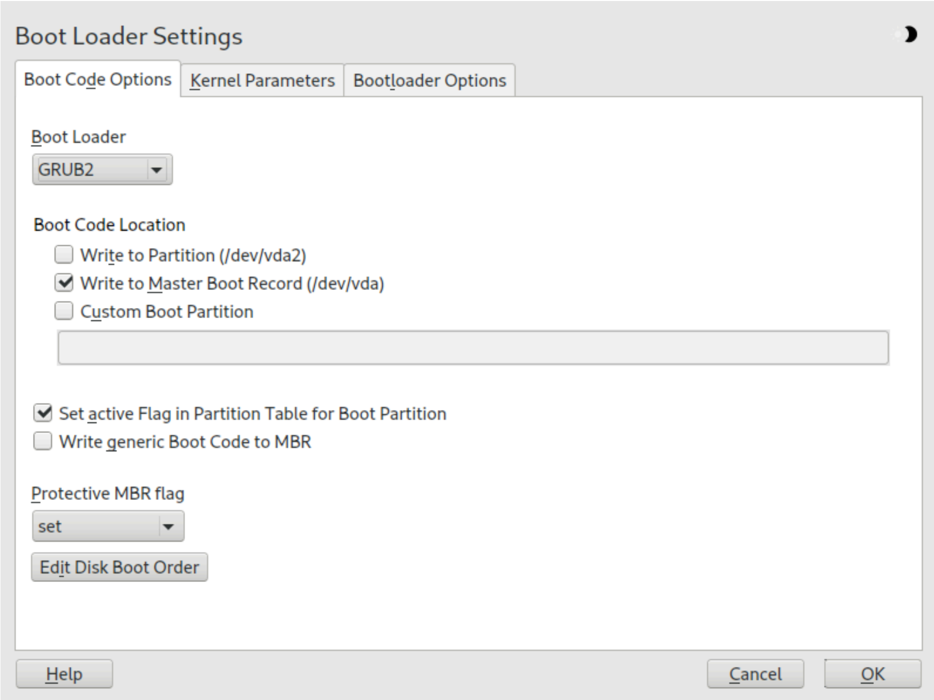


FIGURE 18.2: BOOT CODE OPTIONS

The *Boot Code Options* tab includes the following additional options:

Set Active Flag in Partition Table for Boot Partition

Activates the partition that contains the `/boot` directory. For POWER systems it activates the PReP partition. Use this option on systems with old BIOS and/or legacy operating systems because they may fail to boot from a non-active partition. It is safe to leave this option active.

Write Generic Boot Code to MBR

If MBR contains a custom 'non-GRUB' code, this option replaces it with a generic, operating system independent code. If you deactivate this option, the system may become unbootable.

Enable Trusted Boot Support

Starts TrustedGRUB2, which supports trusted computing functionality (Trusted Platform Module (TPM)). For more information refer to https://github.com/Sirrix-AG/TrustedGRUB2 ↗.

The *Protective MBR flag* section includes the following options:

set

This is appropriate for traditional legacy BIOS booting.

remove

This is appropriate for UEFI booting.

do not change

This is usually the best choice if you have an already working system.

In most cases YaST defaults to the appropriate choice.

18.3.2 Adjusting the disk order

If your computer has more than one hard disk, you can specify the boot sequence of the disks. The first disk in the list is where GRUB 2 will be installed in the case of booting from MBR. It is the disk where SUSE Linux Enterprise Desktop is installed by default. The rest of the list is a hint for GRUB 2's device mapper (see *Section 18.2.4, "Mapping between BIOS drives and Linux devices"*).

Warning: Unbootable system

The default value is usually valid for almost all deployments. If you change the boot order of disks wrongly, the system may become unbootable on the next reboot. For example, if the first disk in the list is not part of the BIOS boot order, and the other disks in the list have empty MBRs.

PROCEDURE 18.2: SETTING THE DISK ORDER

1. Open the *Boot Code Options* tab.
2. Click *Edit Disk Boot Order*.
3. If more than one disk is listed, select a disk and click *Up* or *Down* to reorder the displayed disks.
4. Click *OK* two times to save the changes.

18.3.3 Configuring advanced options

Advanced boot parameters can be configured via the *Boot Loader Options* tab.

18.3.3.1 *Boot Loader Options* tab

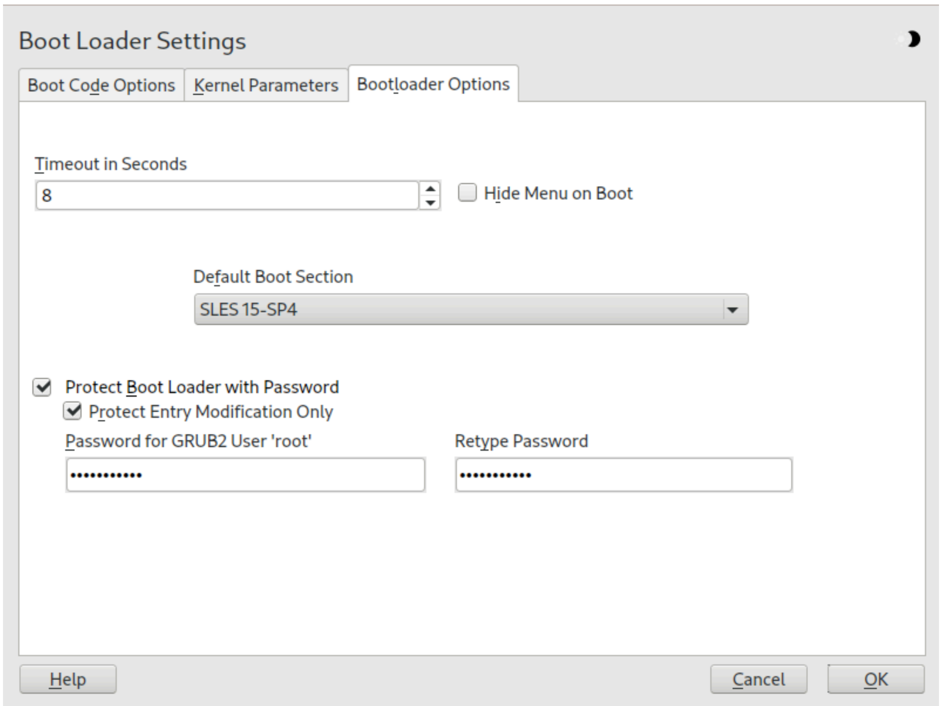


FIGURE 18.3: BOOT LOADER OPTIONS

Boot Loader Time-Out

Change the value of *Time-Out in Seconds* by typing in a new value and clicking the appropriate arrow key with your mouse.

Probe Foreign OS

When selected, the boot loader searches for other systems like Windows or other Linux installations.

Hide Menu on Boot

Hides the boot menu and boots the default entry.

Adjusting the Default Boot Entry

Select the desired entry from the "Default Boot Section" list. Note that the ">" sign in the boot entry name delimits the boot section and its subsection.

Protect Boot Loader with Password

Protects the boot loader and the system with an additional password. For details on manual configuration, see *Section 18.2.6, "Setting a boot password"*. If this option is activated, the boot password is required on every boot, which means the system does not boot automatically. However, if you prefer the behavior of GRUB 1, additionally enable *Protect Entry Modification Only*. With this setting, anybody is allowed to select a boot entry and boot the system, whereas the password for the GRUB 2 `root` user is only required for modifying boot entries.

18.3.3.2 *Kernel Parameters* tab

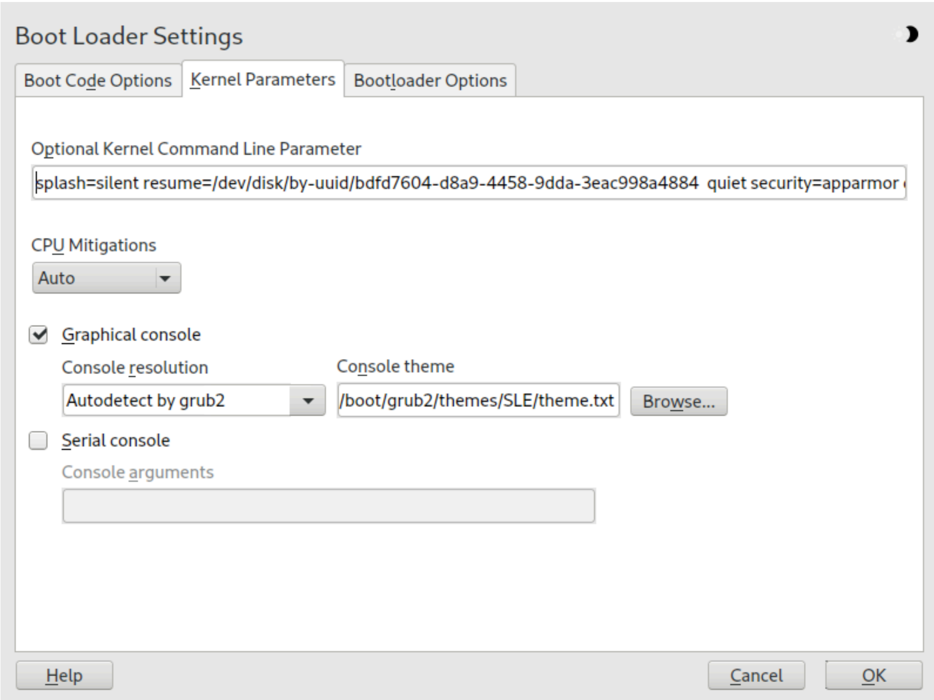


FIGURE 18.4: KERNEL PARAMETERS

Optional Kernel Command Line Parameter

Specify optional kernel parameters here to enable/disable system features, add drivers, etc.

CPU Mitigations

SUSE has released one or more kernel boot command line parameters for all software mitigations that have been deployed to prevent CPU side-channel attacks. Some of those may result in performance loss. Choose one the following options to strike a balance between security and performance, depending on your setting:

Auto. Enables all mitigations required for your CPU model, but does not protect against cross-CPU thread attacks. This setting may impact performance to some degree, depending on the workload.

Auto + No SMT. Provides the full set of available security mitigations. Enables all mitigations required for your CPU model. In addition, it disables Simultaneous Multithreading (SMT) to avoid side-channel attacks across multiple CPU threads. This setting may further impact performance, depending on the workload.

Off. Disables all mitigations. Side-channel attacks against your CPU are possible, depending on the CPU model. This setting has no impact on performance.

Manual. Does not set any mitigation level. Specify your CPU mitigations manually by using the kernel command line options.

Use Graphical Console

When checked, the boot menu appears on a graphical splash screen rather than in a text mode. The resolution of the boot screen is set automatically by default, but you can manually set it via *Console resolution*. The graphical theme definition file can be specified with the *Console theme* file chooser. Only change this if you want to apply your own, custom-made theme.

Use Serial Console

If your machine is controlled via a serial console, activate this option and specify which COM port to use at which speed. See **`info grub`** or https://www.gnu.org/software/grub/manual/grub.html#Serial-terminal

18.4 Helpful GRUB 2 commands

`grub2-mkconfig`

Generates a new `/boot/grub2/grub.cfg` based on `/etc/default/grub` and the scripts from `/etc/grub.d/`.

EXAMPLE 18.1: USAGE OF GRUB2-MKCONFIG

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

Tip: Syntax check

Running **`grub2-mkconfig`** without any parameters prints the configuration to STDOUT where it can be reviewed. Use **`grub2-script-check`** after `/boot/grub2/grub.cfg` has been written to check its syntax.

> **Important: `grub2-mkconfig` cannot repair UEFI Secure Boot tables**
>
> If you are using UEFI Secure Boot and your system is not reaching GRUB 2 correctly anymore, you may need to additionally reinstall the Shim and regenerate the UEFI boot table. To do so, use:
>
> ```
> # shim-install --config-file=/boot/grub2/grub.cfg
> ```

`grub2-mkrescue`

Creates a bootable rescue image of your installed GRUB 2 configuration.

EXAMPLE 18.2: USAGE OF GRUB2-MKRESCUE

```
grub2-mkrescue -o save_path/name.iso iso
```

`grub2-script-check`

Checks the given file for syntax errors.

EXAMPLE 18.3: USAGE OF GRUB2-SCRIPT-CHECK

```
grub2-script-check /boot/grub2/grub.cfg
```

`grub2-once`

Set the default boot entry for the next boot only. To get the list of available boot entries use the `--list` option.

EXAMPLE 18.4: USAGE OF GRUB2-ONCE

```
grub2-once number_of_the_boot_entry
```

> **Tip: `grub2-once` help**
>
> Call the program without any option to get a full list of all possible options.

18.5 Rescue mode

Rescue mode is a specific `root` user session for troubleshooting and repairing systems where the booting process fails. It offers a single-user environment with local file systems and core system services active. Network interfaces are not activated. To enter the rescue mode, follow these steps.

PROCEDURE 18.3: ENTERING RESCUE MODE

1. Reboot the system. The boot screen appears, offering the GRUB 2 boot menu.
2. Select the menu entry to boot and press `e` to edit the boot line.
3. Append the following parameter to the line containing the kernel parameters:

```
systemd.unit=rescue.target
```

4. Press `Ctrl` + `X` to boot with these settings.
5. Enter the password for `root`.
6. Make all the necessary changes.
7. Enter normal operating target again by entering **`systemctl isolate multi-user.target`** or **`systemctl isolate graphical.target`** at the command line.

18.6 More information

Extensive information about GRUB 2 is available at https://www.gnu.org/software/grub/. Also refer to the **`grub`** info page. You can also search for the keyword "GRUB 2" in the Technical Information Search at https://www.suse.com/support to get information about special issues.

19 The systemd daemon

Revision History

2024-05-13

systemd initializes the system. It has the process ID 1. systemd is started directly by the kernel and resists signal 9, which normally terminates processes. All other programs are started directly by systemd or by one of its child processes. systemd is a replacement for the System V init daemon and is fully compatible with System V init (by supporting init scripts).

The main advantage of systemd is that it considerably speeds up boot time by parallelizing service starts. Furthermore, systemd only starts a service when it is really needed. Daemons are not started unconditionally at boot time, but when being required for the first time. systemd also supports Kernel Control Groups (cgroups), creating snapshots, and restoring the system state. For more details see https://www.freedesktop.org/wiki/Software/systemd/.

Tip: systemd inside WSL

Windows Subsystem for Linux (WSL) enables running Linux applications and distributions under the Microsoft Windows operating system. WSL uses its init process instead of systemd. To enable systemd in SLED running in WSL, install the wsl_systemd pattern that automates the process:

```
> sudo zypper in -t pattern wsl_systemd
```

Alternatively, you can edit /etc/wsl.conf and add the following lines manually:

```
[boot]
systemd=true
```

Keep in mind that the support for systemd in WSL is partial—systemd unit files must have reasonable process management behavior.

19.1 The systemd concept

The following section explains the concept behind systemd.

`systemd` is a system and session manager for Linux, compatible with System V and LSB init scripts. The main features of `systemd` include:

- parallelization capabilities
- socket and D-Bus activation for starting services
- on-demand starting of daemons
- tracking of processes using Linux cgroups
- creating snapshots and restoring of the system state
- maintains mount and automount points
- implements an elaborate transactional dependency-based service control logic

19.1.1 Unit file

A unit configuration file contains information about a service, a socket, a device, a mount point, an automount point, a swap file or partition, a start-up target, a watched file system path, a timer controlled and supervised by `systemd`, a temporary system state snapshot, a resource management slice or a group of externally created processes.

"Unit file" is a generic term used by `systemd` for the following:

- **Service.** Information about a process (for example, running a daemon); file ends with .service
- **Targets.** Used for grouping units and as synchronization points during start-up; file ends with .target
- **Sockets.** Information about an IPC or network socket or a file system FIFO, for socket-based activation (like `inetd`); file ends with .socket
- **Path.** Used to trigger other units (for example, running a service when files change); file ends with .path
- **Timer.** Information about a timer controlled, for timer-based activation; file ends with .timer
- **Mount point.** Normally auto-generated by the fstab generator; file ends with .mount
- **Automount point.** Information about a file system automount point; file ends with .automount

- **Swap.** Information about a swap device or file for memory paging; file ends with .swap
- **Device.** Information about a device unit as exposed in the sysfs/udev(7) device tree; file ends with .device
- **Scope / slice.** A concept for hierarchically managing resources of a group of processes; file ends with .scope/.slice

For more information about `systemd` unit files, see https://www.freedesktop.org/software/systemd/man/latest/systemd.unit.html

19.2 Basic usage

The System V init system uses several commands to handle services—the init scripts, **`insserv`**, **`telinit`** and others. `systemd` makes it easier to manage services, because there is only one command to handle most service related tasks: **`systemctl`**. It uses the “command plus subcommand” notation like **`git`** or **`zypper`**:

```
systemctl GENERAL OPTIONS SUBCOMMAND SUBCOMMAND OPTIONS
```

See **`man 1 systemctl`** for a complete manual.

Tip: Terminal output and Bash completion

If the output goes to a terminal (and not to a pipe or a file, for example), `systemd` commands send long output to a pager by default. Use the `--no-pager` option to turn off paging mode.

`systemd` also supports bash-completion, allowing you to enter the first letters of a subcommand and then press →| . This feature is only available in the `bash` shell and requires the installation of the package `bash-completion`.

19.2.1 Managing services in a running system

Subcommands for managing services are the same as for managing a service with System V init (**`start`**, **`stop`**, ...). The general syntax for service management commands is as follows:

`systemd`

```
systemctl reload|restart|start|status|stop|... MY_SERVICE(S)
```

System V init

```
rcMY_SERVICE(S) reload|restart|start|status|stop|...
```

`systemd` allows you to manage several services in one go. Instead of executing init scripts one after the other as with System V init, execute a command like the following:

```
> sudo systemctl start MY_1ST_SERVICE MY_2ND_SERVICE
```

To list all services available on the system:

```
> sudo systemctl list-unit-files --type=service
```

The following table lists the most important service management commands for `systemd` and System V init:

TABLE 19.1: SERVICE MANAGEMENT COMMANDS

Task	`systemd` **Command**	**System V init Command**
Starting.	`start`	`start`
Stopping.	`stop`	`stop`
Restarting. Shuts down services and starts them afterward. If a service is not yet running, it is started.	`restart`	`restart`
Restarting conditionally. Restarts services if they are currently running. Does nothing for services that are not running.	`try-restart`	`try-restart`
Reloading. Tells services to reload their configuration files without interrupting operation. Use case: tell Apache to reload a modified `httpd.conf` configuration file. Not all services support reloading.	`reload`	`reload`
Reloading or restarting. Reloads services if reloading is supported, otherwise restarts	`reload-or-restart`	`n/a`

Task	systemd Command	System V init Command
them. If a service is not yet running, it is started.		
Reloading or restarting conditionally. Reloads services if reloading is supported, otherwise restarts them if currently running. Does nothing for services that are not running.	`reload-or-try-restart`	`n/a`
Getting detailed status information. Lists information about the status of services. The `systemd` command shows details such as description, executable, status, cgroup, and messages last issued by a service (see *Section 19.6.9, "Debugging services"*). The level of details displayed with the System V init differs from service to service.	`status`	`status`
Getting short status information. Shows whether services are active or not.	`is-active`	`status`

19.2.2 Permanently enabling/disabling services

The service management commands mentioned in the previous section let you manipulate services for the current session. `systemd` also lets you permanently enable or disable services, so they are automatically started when requested or are always unavailable. You can either do this by using YaST, or on the command line.

19.2.2.1 Enabling/disabling services on the command line

The following table lists enabling and disabling commands for `systemd` and System V init:

> **Important: Service start**
>
> When enabling a service on the command line, it is not started automatically. It is scheduled to be started with the next system start-up or runlevel/target change. To immediately start a service after having enabled it, explicitly run **`systemctl start`** *`MY_SERVICE`* or **`rc`** *`MY_SERVICE`* **`start`**.

TABLE 19.2: COMMANDS FOR ENABLING AND DISABLING SERVICES

Task	`systemd` Command	System V init Command
Enabling.	**`systemctl enable`** *`MY_SERVICE(S)`*	**`insserv`** *`MY_SERVICE(S)`*, **`chkconfig -a`** *`MY_SERVICE(S)`*
Disabling.	**`systemctl disable`** *`MY_SERVICE(S)`***`.service`**	**`insserv -r`** *`MY_SERVICE(S)`*, **`chkconfig -d`** *`MY_SERVICE(S)`*
Checking. Shows whether a service is enabled or not.	**`systemctl is-enabled`** *`MY_SERVICE`*	**`chkconfig`** *`MY_SERVICE`*
Re-enabling. Similar to restarting a service, this command first disables and then enables a service. Useful to re-enable a service with its defaults.	**`systemctl reenable`** *`MY_SERVICE`*	n/a
Masking. After “disabling” a service, it can still be started manually. To disable	**`systemctl mask`** *`MY_SERVICE`*	n/a

Task	`systemd` **Command**	**System V init Command**
a service, you need to mask it. Use with care.		
Unmasking. A service that has been masked can only be used again after it has been unmasked.	**`systemctl unmask`** *`MY_SERVICE`*	n/a

19.3 System start and target management

The entire process of starting the system and shutting it down is maintained by `systemd`. From this point of view, the kernel can be considered a background process to maintain all other processes and adjust CPU time and hardware access according to requests from other programs.

19.3.1 Targets compared to runlevels

With System V init the system was booted into a so-called “Runlevel”. A runlevel defines how the system is started and what services are available in the running system. Runlevels are numbered; the most commonly known ones are `0` (shutting down the system), `3` (multiuser with network) and `5` (multiuser with network and display manager).

`systemd` introduces a new concept by using so-called “target units”. However, it remains fully compatible with the runlevel concept. Target units are named rather than numbered and serve specific purposes. For example, the targets `local-fs.target` and `swap.target` mount local file systems and swap spaces.

The target `graphical.target` provides a multiuser system with network and display manager capabilities and is equivalent to runlevel 5. Complex targets, such as `graphical.target` act as “meta” targets by combining a subset of other targets. Since `systemd` makes it easy to create custom targets by combining existing targets, it offers great flexibility.

The following list shows the most important `systemd` target units. For a full list refer to **`man 7 systemd.special`**.

SELECTED `systemd` TARGET UNITS

`default.target`
: The target that is booted by default. Not a "real" target, but rather a symbolic link to another target like `graphic.target`. Can be permanently changed via YaST (see *Section 19.4, "Managing services with YaST"*). To change it for a session, use the kernel parameter `systemd.unit=MY_TARGET.target` at the boot prompt.

`emergency.target`
: Starts a minimal emergency `root` shell on the console. Only use it at the boot prompt as `systemd.unit=emergency.target`.

`graphical.target`
: Starts a system with network, multiuser support and a display manager.

`halt.target`
: Shuts down the system.

`mail-transfer-agent.target`
: Starts all services necessary for sending and receiving mails.

`multi-user.target`
: Starts a multiuser system with network.

`reboot.target`
: Reboots the system.

`rescue.target`
: Starts a single-user `root` session without network. Basic tools for system administration are available. The `rescue` target is suitable for solving multiple system problems, for example, failing logins or fixing issues with a display driver.

To remain compatible with the System V init runlevel system, `systemd` provides special targets named `runlevelX.target` mapping the corresponding runlevels numbered *X*.

To inspect the current target, use the command: **`systemctl get-default`**

TABLE 19.3: SYSTEM V RUNLEVELS AND `systemd` TARGET UNITS

System V runlevel	`systemd` target	Purpose
0	`runlevel0.target`, `halt.target`, `poweroff.target`	System shutdown

System V runlevel	`systemd` target	Purpose
1, S	`runlevel1.target`, `rescue.target`,	Single-user mode
2	`runlevel2.target`, `multi-user.target`,	Local multiuser without remote network
3	`runlevel3.target`, `multi-user.target`,	Full multiuser with network
4	`runlevel4.target`	Unused/User-defined
5	`runlevel5.target`, `graphical.target`,	Full multiuser with network and display manager
6	`runlevel6.target`, `reboot.target`,	System reboot

Important: `systemd` ignores `/etc/inittab`

The runlevels in a System V init system are configured in `/etc/inittab`. `systemd` does *not* use this configuration. Refer to *Section 19.5.5, "Creating custom targets"* for instructions on how to create your own bootable target.

19.3.1.1 Commands to change targets

Use the following commands to operate with target units:

Task	`systemd` Command	System V init Command
Change the current target/runlevel	**`systemctl isolate`** *`MY_TARGET`*.target	**`telinit`** *`X`*

Task	`systemd` Command	System V init Command
Change to the default target/runlevel	**`systemctl default`**	n/a
Get the current target/runlevel	**`systemctl list-units --type=target`** With `systemd`, there is usually more than one active target. The command lists all currently active targets.	**`who -r`** or **`runlevel`**
persistently change the default runlevel	Use the Services Manager or run the following command: **`ln -sf /usr/lib/systemd/system/`** *`MY_TARGET`*.target /etc/systemd/system/default.target	Use the Services Manager or change the line **`id:`** *`X`*:initdefault: in `/etc/inittab`
Change the default runlevel for the current boot process	Enter the following option at the boot prompt **`systemd.unit=`** *`MY_TARGET`*.target	Enter the desired runlevel number at the boot prompt.
Show a target's/runlevel's dependencies	**`systemctl show -p "Requires"`** *`MY_TARGET`*.target **`systemctl show -p "Wants"`** *`MY_TARGET`*.target “Requires” lists the hard dependencies (the ones that must be resolved), whereas “Wants” lists the soft dependencies (the ones that get resolved if possible).	n/a

19.3.2 Debugging system start-up

`systemd` offers the means to analyze the system start-up process. You can review the list of all services and their status (rather than having to parse `/var/log/`). `systemd` also allows you to scan the start-up procedure to find out how much time each service start-up consumes.

19.3.2.1 Review start-up of services

To review the complete list of services that have been started since booting the system, enter the command **systemctl**. It lists all active services like shown below (shortened). To get more information on a specific service, use **systemctl status** *MY_SERVICE*.

EXAMPLE 19.1: LIST ACTIVE SERVICES

```
# systemctl
UNIT                        LOAD   ACTIVE SUB       JOB DESCRIPTION
[...]
iscsi.service               loaded active exited    Login and scanning of iSC+
kmod-static-nodes.service   loaded active exited    Create list of required s+
libvirtd.service            loaded active running   Virtualization daemon
nscd.service                loaded active running   Name Service Cache Daemon
chronyd.service             loaded active running   NTP Server Daemon
polkit.service              loaded active running   Authorization Manager
postfix.service             loaded active running   Postfix Mail Transport Ag+
rc-local.service            loaded active exited    /etc/init.d/boot.local Co+
rsyslog.service             loaded active running   System Logging Service
[...]
LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

161 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

To restrict the output to services that failed to start, use the `--failed` option:

EXAMPLE 19.2: LIST FAILED SERVICES

```
# systemctl --failed
UNIT                   LOAD   ACTIVE SUB    JOB DESCRIPTION
apache2.service        loaded failed failed     apache
NetworkManager.service loaded failed failed     Network Manager
plymouth-start.service loaded failed failed     Show Plymouth Boot Screen

[...]
```

19.3.2.2 Debug start-up time

To debug system start-up time, `systemd` offers the **systemd-analyze** command. It shows the total start-up time, a list of services ordered by start-up time and can also generate an SVG graphic showing the time services took to start in relation to the other services.

Listing the system start-up time

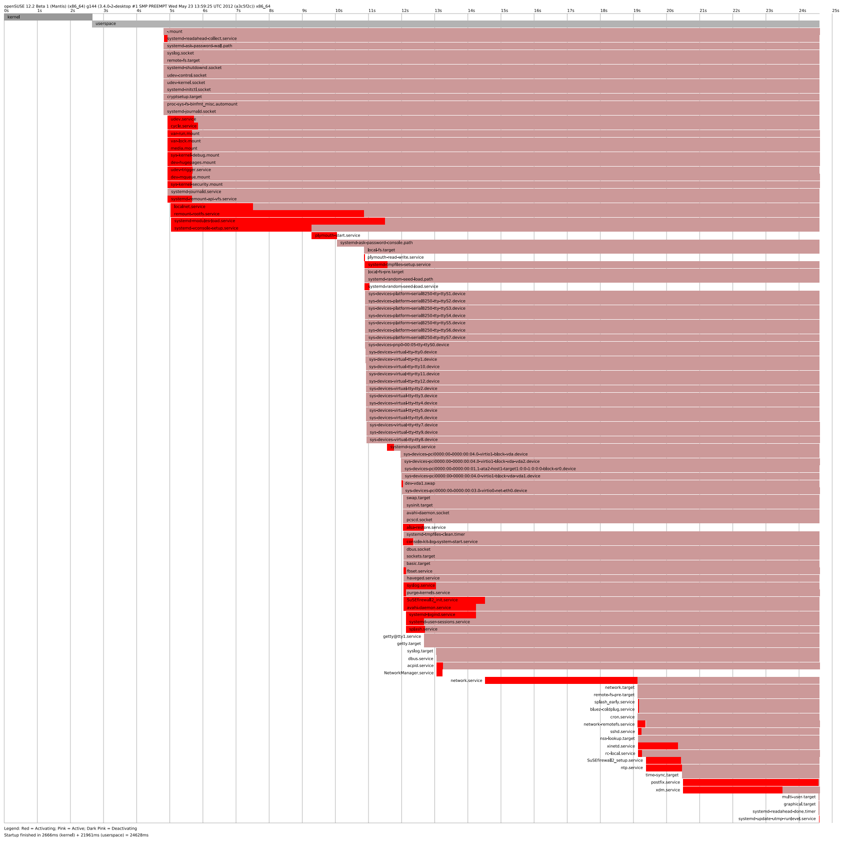
```
# systemd-analyze
Startup finished in 2666ms (kernel) + 21961ms (userspace) = 24628ms
```

Listing the services start-up time

```
# systemd-analyze blame
    15.000s backup-rpmdb.service
    14.879s mandb.service
     7.646s backup-sysconfig.service
     4.940s postfix.service
     4.921s logrotate.service
     4.640s libvirtd.service
     4.519s display-manager.service
     3.921s btrfsmaintenance-refresh.service
     3.466s lvm2-monitor.service
     2.774s plymouth-quit-wait.service
     2.591s firewalld.service
     2.137s initrd-switch-root.service
     1.954s ModemManager.service
     1.528s rsyslog.service
     1.378s apparmor.service
    [...]
```

Services start-up time graphics

```
# systemd-analyze plot > jupiter.example.com-startup.svg
```



19.3.2.3 Review the complete start-up process

The commands above list the services that are started and their start-up times. For a more detailed overview, specify the following parameters at the boot prompt to instruct `systemd` to create a verbose log of the complete start-up procedure.

```
systemd.log_level=debug systemd.log_target=kmsg
```

Now `systemd` writes its log messages into the kernel ring buffer. View that buffer with **`dmesg`**:

```
> dmesg -T | less
```

19.3.3 System V compatibility

`systemd` is compatible with System V, allowing you to still use existing System V init scripts. However, there is at least one known issue where a System V init script does not work with `systemd` out of the box: starting a service as a different user via **`su`** or **`sudo`** in init scripts results in a failure of the script, producing an “Access denied” error.

When changing the user with **`su`** or **`sudo`**, a PAM session is started. This session will be terminated after the init script is finished. As a consequence, the service that has been started by the init script is also terminated. To work around this error, proceed as follows:

1. Create a service file wrapper with the same name as the init script plus the file name extension `.service`:

```
[Unit]
Description=DESCRIPTION
After=network.target

[Service]
User=USER
Type=forking ❶
PIDFile=PATH TO PID FILE ❶
ExecStart=PATH TO INIT SCRIPT start
ExecStop=PATH TO INIT SCRIPT stop
ExecStopPost=/usr/bin/rm -f PATH TO PID FILE ❶

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

   Replace all values written in *`UPPERCASE LETTERS`* with appropriate values.

   ❶ Optional—only use if the init script starts a daemon.

   ❷ `multi-user.target` also starts the init script when booting into `graphical.target`. If it should only be started when booting into the display manager, use `graphical.target`.

2. Start the daemon with **`systemctl start`** *`APPLICATION`*.

19.4 Managing services with YaST

Basic service management can also be done with the YaST Services Manager module. It supports starting, stopping, enabling and disabling services. It also lets you show a service's status and change the default target. Start the YaST module with *YaST* › *System* › *Services Manager*.

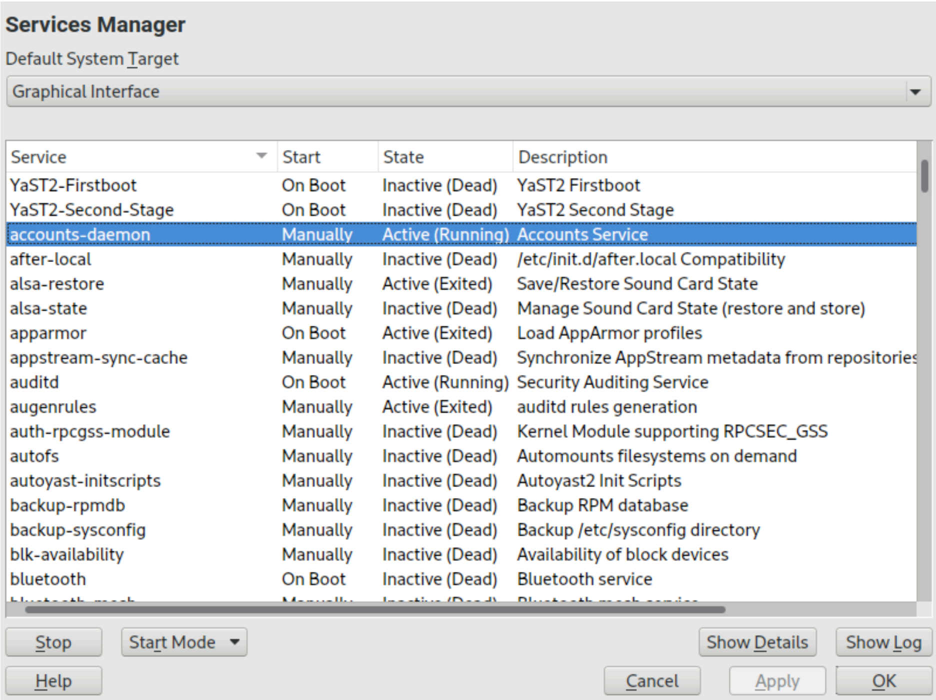


FIGURE 19.1: SERVICES MANAGER

Changing the *Default system target*

To change the target the system boots into, choose a target from the *Default System Target* drop-down box. The most often used targets are *Graphical Interface* (starting a graphical login screen) and *Multi-User* (starting the system in command line mode).

Starting or stopping a service

Select a service from the table. The *State* column shows whether it is currently running (*Active*) or not (*Inactive*). Toggle its status by choosing *Start* or *Stop*.

Starting or stopping a service changes its status for the currently running session. To change its status throughout a reboot, you need to enable or disable it.

Defining service start-up behavior

Services can either be started automatically at boot time or manually. Select a service from the table. The *Start* column shows whether it is currently started *Manually* or *On Boot*. Toggle its status by choosing *Start Mode*.

To change a service status in the current session, you need to start or stop it as described above.

View a status messages

To view the status message of a service, select it from the list and choose *Show Details*. The output is identical to the one generated by the command **`systemctl`** `-l` status *`MY_SERVICE`*.

19.5 Customizing `systemd`

The following sections describe how to customize `systemd` unit files.

19.5.1 Where are unit files stored?

`systemd` unit files shipped by SUSE are stored in `/usr/lib/systemd/`. Customized unit files and unit file *drop-ins* are stored in `/etc/systemd/`.

Warning: Preventing your customization from being overwritten

When customizing `systemd`, always use the directory `/etc/systemd/` instead of `/usr/lib/systemd/`. Otherwise your changes will be overwritten by the next update of `systemd`.

19.5.2 Override with drop-in files

Drop-in files (or *drop-ins*) are partial unit files that override only specific settings of the unit file. Drop-ins have higher precedence over main configuration files. The command **`systemctl edit`** *`SERVICE`* starts the default text editor and creates a directory with an empty `override.conf` file in `/etc/systemd/system/`*`NAME`*`.service.d/`. The command also ensures that the running `systemd` process is notified about the changes.

For example, to change the amount of time that the system waits for MariaDB to start, run **`sudo systemctl edit mariadb.service`** and edit the opened file to include the modified lines only:

```
# Configures the time to wait for start-up/stop
TimeoutSec=300
```

Adjust the `TimeoutSec` value and save the changes. To enable the changes, run **`sudo systemctl daemon-reload`**.

For further information, refer to the man pages that can be evoked with the **`man 1 systemctl`** command.

Warning: Creating a copy of a full unit file

If you use the `--full` option in the **`systemctl edit --full`** *`SERVICE`* command, a copy of the original unit file is created where you can modify specific options. We do not recommend such customization because when the unit file is updated by SUSE, its changes are overridden by the customized copy in the `/etc/systemd/system/` directory. Moreover, if SUSE provides updates to distribution drop-ins, they override the copy of the unit file created with `--full`. To prevent this confusion and always have your customization valid, use drop-ins.

19.5.3 Creating drop-in files manually

Apart from using the **`systemctl edit`** command, you can create drop-ins manually to have more control over their priority. Such drop-ins let you extend both unit and daemon configuration files without having to edit or override the files themselves. They are stored in the following directories:

`/etc/systemd/*.conf.d/`, `/etc/systemd/system/*.service.d/`
: Drop-ins added and customized by system administrators.

`/usr/lib/systemd/*.conf.d/`, `/usr/lib/systemd/system/*.service.d/`
: Drop-ins installed by customization packages to override upstream settings. For example, SUSE ships `systemd-default-settings`.

Tip

See the man page **`man 5 systemd.unit`** for the full list of unit search paths.

For example, to disable the rate limiting that is enforced by the default setting of `systemd-journald`, follow these steps:

1. Create a directory called `/etc/systemd/journald.conf.d`.

```
> sudo mkdir /etc/systemd/journald.conf.d
```

Note

The directory name must follow the service name that you want to patch with the drop-in file.

2. In that directory, create a file `/etc/systemd/journald.conf.d/60-rate-limit.conf` with the option that you want to override, for example:

```
> cat /etc/systemd/journald.conf.d/60-rate-limit.conf
# Disable rate limiting
RateLimitIntervalSec=0
```

3. Save your changes and restart the service of the corresponding `systemd` daemon.

```
> sudo systemctl restart systemd-journald
```

Note: Avoiding name conflicts

To avoid name conflicts between your drop-ins and files shipped by SUSE, it is recommended to prefix all drop-ins with a two-digit number and a dash, for example, `80-override.conf`.

The following ranges are reserved:

- `0-19` is reserved for `systemd` upstream.
- `20-29` is reserved for `systemd` shipped by SUSE.
- `30-39` is reserved for SUSE packages other than `systemd`.
- `40-49` is reserved for third-party packages.
- `50` is reserved for unit drop-in files created with **`systemctl set-property`**.

Use a two-digit number above this range to ensure that none of the drop-ins shipped by SUSE can override your own drop-ins.

Tip

You can use **`systemctl cat $UNIT`** to list and verify which files are taken into account in the units configuration.

Tip

Because the configuration of `systemd` components can be scattered across different places on the file system, it might be hard to get a global overview. To inspect the configuration of a `systemd` component, use the following commands:

- **`systemctl cat`** *`UNIT_PATTERN`* prints configuration files related to one or more `systemd` units, for example:

  ```
  > systemctl cat atd.service
  ```

- **`systemd-analyze cat-config`** *`DAEMON_NAME_OR_PATH`* copies the contents of a configuration file and drop-ins for a `systemd` daemon, for example:

  ```
  > systemd-analyze cat-config systemd/journald.conf
  ```

19.5.4 Converting `xinetd` services to `systemd`

Since the release of SUSE Linux Enterprise Desktop 15, the `xinetd` infrastructure has been removed. This section outlines how to convert existing custom `xinetd` service files to `systemd` sockets.

For each `xinetd` service file, you need at least two `systemd` unit files: the socket file (`*.socket`) and an associated service file (`*.service`). The socket file tells `systemd` which socket to create, and the service file tells `systemd` which executable to start.

Consider the following example `xinetd` service file:

```
# cat /etc/xinetd.d/example
service example
{
  socket_type = stream
  protocol = tcp
  port = 10085
  wait = no
  user = user
  group = users
  groups = yes
  server = /usr/libexec/example/exampled
  server_args = -auth=bsdtcp exampledump
  disable = no
```

```
}
```

To convert it to `systemd`, you need the following two matching files:

```
# cat /usr/lib/systemd/system/example.socket
[Socket]
ListenStream=0.0.0.0:10085
Accept=false

[Install]
WantedBy=sockets.target
```

```
# cat /usr/lib/systemd/system/example.service
[Unit]
Description=example

[Service]
ExecStart=/usr/libexec/example/exampled -auth=bsdtcp exampledump
User=user
Group=users
StandardInput=socket
```

For a complete list of the `systemd` "socket" and "service" file options, refer to the systemd.socket and systemd.service manual pages (**`man 5 systemd.socket`**, **`man 5 systemd.service`**).

19.5.5 Creating custom targets

On System V init SUSE systems, runlevel 4 is unused to allow administrators to create their own runlevel configuration. `systemd` allows you to create any number of custom targets. It is suggested to start by adapting an existing target such as `graphical.target`.

1. Copy the configuration file `/usr/lib/systemd/system/graphical.target` to `/etc/systemd/system/MY_TARGET.target` and adjust it according to your needs.

2. The configuration file copied in the previous step already covers the required ("hard") dependencies for the target. To also cover the wanted ("soft") dependencies, create a directory `/etc/systemd/system/MY_TARGET.target.wants`.

3. For each wanted service, create a symbolic link from `/usr/lib/systemd/system` into `/etc/systemd/system/MY_TARGET.target.wants`.

4. When you have finished setting up the target, reload the `systemd` configuration to make the new target available:

```
> sudo systemctl daemon-reload
```

19.6 Advanced usage

The following sections cover advanced topics for system administrators. For even more advanced `systemd` documentation, refer to Lennart Pöttering's series about `systemd` for administrators at https://0pointer.de/blog/projects/.

19.6.1 Cleaning temporary directories

`systemd` supports cleaning temporary directories regularly. The configuration from the previous system version is automatically migrated and active. `tmpfiles.d`—which is responsible for managing temporary files—reads its configuration from `/etc/tmpfiles.d/*.conf`, `/run/tmpfiles.d/*.conf`, and `/usr/lib/tmpfiles.d/*.conf` files. Configuration placed in `/etc/tmpfiles.d/*.conf` overrides related configurations from the other two directories (`/usr/lib/tmpfiles.d/*.conf` is where packages store their configuration files).

The configuration format is one line per path containing action and path, and optionally mode, ownership, age and argument fields, depending on the action. The following example unlinks the X11 lock files:

```
Type Path               Mode UID  GID  Age Argument
r    /tmp/.X[0-9]*-lock
```

To get the status the tmpfile timer:

```
> sudo systemctl status systemd-tmpfiles-clean.timer
systemd-tmpfiles-clean.timer - Daily Cleanup of Temporary Directories
 Loaded: loaded (/usr/lib/systemd/system/systemd-tmpfiles-clean.timer; static)
 Active: active (waiting) since Tue 2018-04-09 15:30:36 CEST; 1 weeks 6 days ago
   Docs: man:tmpfiles.d(5)
         man:systemd-tmpfiles(8)

Apr 09 15:30:36 jupiter systemd[1]: Starting Daily Cleanup of Temporary Directories.
Apr 09 15:30:36 jupiter systemd[1]: Started Daily Cleanup of Temporary Directories.
```

For more information on temporary files handling, see **`man 5 tmpfiles.d`**.

19.6.2 System log

Section 19.6.9, "Debugging services" explains how to view log messages for a given service. However, displaying log messages is not restricted to service logs. You can also access and query the complete log messages written by `systemd`—the so-called "Journal". Use the command **`journalctl`** to display the complete log messages starting with the oldest entries. Refer to **`man 1 journalctl`** for options such as applying filters or changing the output format.

19.6.3 Snapshots

You can save the current state of `systemd` to a named snapshot and later revert to it with the **`isolate`** subcommand. This is useful when testing services or custom targets, because it allows you to return to a defined state at any time. A snapshot is only available in the current session and will automatically be deleted on reboot. A snapshot name must end in `.snapshot`.

Create a snapshot

```
> sudo systemctl snapshot MY_SNAPSHOT.snapshot
```

Delete a snapshot

```
> sudo systemctl delete MY_SNAPSHOT.snapshot
```

View a snapshot

```
> sudo systemctl show MY_SNAPSHOT.snapshot
```

Activate a snapshot

```
> sudo systemctl isolate MY_SNAPSHOT.snapshot
```

19.6.4 Loading kernel modules

With `systemd`, kernel modules can automatically be loaded at boot time via a configuration file in `/etc/modules-load.d`. The file should be named *`MODULE`*.conf and have the following content:

```
# load module MODULE at boot time
MODULE
```

In case a package installs a configuration file for loading a kernel module, the file gets installed to `/usr/lib/modules-load.d`. If two configuration files with the same name exist, the one in `/etc/modules-load.d` tales precedence.

For more information, see the `modules-load.d(5)` man page.

19.6.5 Performing actions before loading a service

With System V init actions that need to be performed before loading a service, needed to be specified in `/etc/init.d/before.local` . This procedure is no longer supported with `systemd`. If you need to do actions before starting services, do the following:

Loading kernel modules

Create a drop-in file in `/etc/modules-load.d` directory (see **`man modules-load.d`** for the syntax)

Creating Files or Directories, Cleaning-up Directories, Changing Ownership

Create a drop-in file in `/etc/tmpfiles.d` (see **`man tmpfiles.d`** for the syntax)

Other tasks

Create a system service file, for example, `/etc/systemd/system/before.service`, from the following template:

```
[Unit]
Before=NAME OF THE SERVICE YOU WANT THIS SERVICE TO BE STARTED BEFORE
[Service]
Type=oneshot
RemainAfterExit=true
ExecStart=YOUR_COMMAND
# beware, executable is run directly, not through a shell, check the man pages
# systemd.service and systemd.unit for full syntax
[Install]
# target in which to start the service
WantedBy=multi-user.target
#WantedBy=graphical.target
```

When the service file is created, you should run the following commands (as `root`):

```
> sudo systemctl daemon-reload
> sudo systemctl enable before
```

Every time you modify the service file, you need to run:

```
> sudo systemctl daemon-reload
```

19.6.6 Kernel control groups (cgroups)

On a traditional System V init system, it is not always possible to match a process to the service that spawned it. Certain services, such as Apache, spawn a lot of third-party processes (for example, CGI or Java processes), which themselves spawn more processes. This makes a clear assignment difficult or even impossible. Additionally, a service may not finish correctly, leaving certain children alive.

`systemd` solves this problem by placing each service into its own cgroup. cgroups are a kernel feature that allows aggregating processes and all their children into hierarchical organized groups. `systemd` names each cgroup after its service. Since a non-privileged process is not allowed to “leave” its cgroup, this provides an effective way to label all processes spawned by a service with the name of the service.

To list all processes belonging to a service, use the command **`systemd-cgls`**, for example:

EXAMPLE 19.3: LIST ALL PROCESSES BELONGING TO A SERVICE

```
# systemd-cgls --no-pager
├─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 20
├─user.slice
│ └─user-1000.slice
│   ├─session-102.scope
│   │ ├─12426 gdm-session-worker [pam/gdm-password]
│   │ ├─15831 gdm-session-worker [pam/gdm-password]
│   │ ├─15839 gdm-session-worker [pam/gdm-password]
│   │ ├─15858 /usr/lib/gnome-terminal-server

[...]

└─system.slice
  ├─systemd-hostnamed.service
  │ └─17616 /usr/lib/systemd/systemd-hostnamed
  ├─cron.service
  │ └─1689 /usr/sbin/cron -n
  ├─postfix.service
  │ ├─ 1676 /usr/lib/postfix/master -w
  │ ├─ 1679 qmgr -l -t fifo -u
  │ └─15590 pickup -l -t fifo -u
  ├─sshd.service
  │ └─1436 /usr/sbin/sshd -D

[...]
```

See *Book “System Analysis and Tuning Guide”, Chapter 10 “Kernel control groups”* for more information about cgroups.

19.6.7 Terminating services (sending signals)

As explained in *Section 19.6.6, "Kernel control groups (cgroups)"*, it is not always possible to assign a process to its parent service process in a System V init system. This makes it difficult to stop a service and its children. Child processes that have not been terminated remain as zombie processes.

`systemd`'s concept of confining each service into a cgroup makes it possible to identify all child processes of a service and therefore allows you to send a signal to each of these processes. Use **`systemctl kill`** to send signals to services. For a list of available signals refer to **`man 7 signals`**.

Sending `SIGTERM` to a service

`SIGTERM` is the default signal that is sent.

```
> sudo systemctl kill MY_SERVICE
```

Sending *`SIGNAL`* to a service

Use the `-s` option to specify the signal that should be sent.

```
> sudo systemctl kill -s SIGNAL MY_SERVICE
```

Selecting processes

By default the **`kill`** command sends the signal to `all` processes of the specified cgroup. You can restrict it to the `control` or the `main` process. The latter is, for example, useful to force a service to reload its configuration by sending `SIGHUP`:

```
> sudo systemctl kill -s SIGHUP --kill-who=main MY_SERVICE
```

19.6.8 Important notes on the D-Bus service

The D-Bus service is the message bus for communication between `systemd` clients and the systemd manager that is running as pid 1. Even though `dbus` is a stand-alone daemon, it is an integral part of the init infrastructure.

Stopping `dbus` or restarting it in the running system is similar to an attempt to stop or restart PID 1. It breaks the `systemd` client/server communication and makes most `systemd` functions unusable.

Therefore, terminating or restarting `dbus` is neither recommended nor supported.

Updating the `dbus` or `dbus`-related packages requires a reboot. When in doubt whether a reboot is necessary, run the **`sudo zypper ps -s`**. If `dbus` appears among the listed services, you need to reboot the system.

Keep in mind that `dbus` is updated even when automatic updates are configured to skip the packages that require reboot.

19.6.9 Debugging services

By default, `systemd` is not overly verbose. If a service was started successfully, no output is produced. In case of a failure, a short error message is displayed. However, **`systemctl status`** provides a means to debug the start-up and operation of a service.

`systemd` comes with its own logging mechanism ("The Journal") that logs system messages. This allows you to display the service messages together with status messages. The **`status`** command works similar to **`tail`** and can also display the log messages in different formats, making it a powerful debugging tool.

Show service start-up failure

Whenever a service fails to start, use **`systemctl status`** *`MY_SERVICE`* to get a detailed error message:

```
# systemctl start apache2
Job failed. See system journal and 'systemctl status' for details.
# systemctl status apache2
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
   Active: failed (Result: exit-code) since Mon, 04 Apr 2018 16:52:26 +0200; 29s ago
   Process: 3088 ExecStart=/usr/sbin/start_apache2 -D SYSTEMD -k start (code=exited,
 status=1/FAILURE)
   CGroup: name=systemd:/system/apache2.service

Apr 04 16:52:26 g144 start_apache2[3088]: httpd2-prefork: Syntax error on line
205 of /etc/apache2/httpd.conf: Syntax error on li...alHost>
```

Show last *N* service messages

The default behavior of the **`status`** subcommand is to display the last ten messages a service issued. To change the number of messages to show, use the `--lines=`*`N`* parameter:

```
> sudo systemctl status chronyd
> sudo systemctl --lines=20 status chronyd
```

Show service messages in append mode

To display a "live stream" of service messages, use the `--follow` option, which works like **`tail`** `-f`:

```
> sudo systemctl --follow status chronyd
```

Messages output format

The `--output=MODE` parameter allows you to change the output format of service messages. The most important modes available are:

`short`

The default format. Shows the log messages with a human readable time stamp.

`verbose`

Full output with all fields.

`cat`

Terse output without time stamps.

19.7 systemd timer units

Similar to cron, `systemd` timer units provide a mechanism for scheduling jobs on Linux. Although `systemd` timer units serve the same purpose as cron, they offer several advantages.

- Jobs scheduled using a timer unit can depend on other `systemd` services.
- Timer units are treated as regular `systemd` services, so can be managed with **`systemctl`**.
- Timers can be realtime and monotonic.
- Time units are logged to the `systemd` journal, which makes it easier to monitor and troubleshoot them.

`systemd` timer units are identified by the `.timer` file name extension.

19.7.1 systemd timer types

Timer units can use monotonic and realtime timers.

- Similar to cronjobs, realtime timers are triggered on calendar events. Realtime timers are defined using the option `OnCalendar`.
- Monotonic timers are triggered at a specified time elapsed from a certain starting point. The latter could be a system boot or system unit activation event. There are several options for defining monotonic timers including `OnBootSec`, `OnUnitActiveSec`, and `OnTypeSec`. Monotonic timers are not persistent, and they are reset after each reboot.

19.7.2 systemd timers and service units

Every timer unit must have a corresponding `systemd` unit file it controls. In other words, a `.timer` file activates and manages the corresponding `.service` file. When used with a timer, the `.service` file does not require an `[Install]` section, as the service is managed by the timer.

19.7.3 Practical example

To understand the basics of `systemd` timer units, we set up a timer that triggers the `foo.sh` shell script.

First step is to create a `systemd` service unit that controls the shell script. To do this, open a new text file for editing and add the following service unit definition:

```
[Unit]
Description="Foo shell script"

[Service]
ExecStart=/usr/local/bin/foo.sh
```

Save the file under the name `foo.service` in the directory `/etc/systemd/system/`.

Next, open a new text file for editing and add the following timer definition:

```
[Unit]
Description="Run foo shell script"

[Timer]
OnBootSec=5min
OnUnitActiveSec=24h
Unit=foo.service

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

The `[Timer]` section in the example above specifies what service to trigger (`foo.service`) and when to trigger it. In this case, the option `OnBootSec` specifies a monotonic timer that triggers the service five minutes after the system boot, while the option `OnUnitActiveSec` triggers the service 24 hours after the service has been activated (that is, the timer triggers the service once a day). Finally, the option `WantedBy` specifies that the timer should start when the system has reached the multiuser target.

Instead of a monotonic timer, you can specify a real-time one using the option `OnCalendar`. The following realtime timer definition triggers the related service unit once a week, starting on Monday at 12:00.

```
[Timer]
OnCalendar=weekly
Persistent=true
```

The option `Persistent=true` indicates that the service is triggered immediately after the timer activation if the timer missed the last start time (for example, because of the system being powered off).

The option `OnCalendar` can also be used to define specific dates times for triggering a service using the following format: `DayOfWeek Year-Month-Day Hour:Minute:Second`. The example below triggers a service at 5am every day:

```
OnCalendar=*-*-* 5:00:00
```

You can use an asterisk to specify any value, and commas to list possible values. Use two values separated by .. to indicate a contiguous range. The following example triggers a service at 6pm on Friday of every month:

```
OnCalendar=Fri *-*-1..7 18:00:00
```

To trigger a service at different times, you can specify several `OnCalendar` entries:

```
OnCalendar=Mon..Fri 10:00
OnCalendar=Sat,Sun 22:00
```

In the example above, a service is triggered at 10am on week days and at 10pm on weekends.

When you are done editing the timer unit file, save it under the name `foo.timer` in the `/etc/systemd/system/` directory. To check the correctness of the created unit files, run the following command:

```
> sudo  systemd-analyze verify /etc/systemd/system/foo.*
```

If the command returns no output, the files have passed the verification successfully.

To start the timer, use the command **`sudo systemctl start foo.timer`**. To enable the timer on boot, run the command **`sudo systemctl enable foo.timer`**.

19.7.4 Managing systemd timers

Since timers are treated as regular `systemd` units, you can manage them using **`systemctl`**. You can start a timer with **`systemctl start`**, enable a timer with **`systemctl enable`**, and so on. Additionally, you can list all active timers using the command **`systemctl list-timers`**. To list all timers, including inactive ones, run the command **`systemctl list-timers --all`**.

19.8 More information

For more information on `systemd` refer to the following online resources:

Homepage

https://systemd.io/ ↗

`systemd` for administrators

Lennart Pöttering, one of the `systemd` authors, has written a series of blog entries (13 at the time of writing this chapter). Find them at https://0pointer.de/blog/projects/ ↗.

III System

20 32-bit and 64-bit applications in a 64-bit system environment 291

21 **`journalctl`**: query the `systemd` journal 293

22 **`update-alternatives`**: managing multiple versions of commands and files 301

23 Basic networking 309

24 Printer operation 374

25 Graphical user interface 388

26 Accessing file systems with FUSE 407

27 Installing multiple kernel versions 409

28 Managing kernel modules 416

29 Dynamic kernel device management with `udev` 420

30 Special system features 432

31 Using NetworkManager 444

20 32-bit and 64-bit applications in a 64-bit system environment

Revision History

2024-05-13

SUSE® Linux Enterprise Desktop is available for 64-bit platforms. The developers have not ported all 32-bit applications to 64-bit systems. This chapter offers a brief overview of 32-bit support implementation on 64-bit SUSE Linux Enterprise Desktop platforms.

SUSE Linux Enterprise Desktop for the 64-bit platforms AMD64 and Intel 64 is designed so that existing 32-bit applications run in the 64-bit environment “out-of-the-box.” This support means that you can continue to use your preferred 32-bit applications without waiting for a corresponding 64-bit port to become available.

Note: No support for building 32-bit applications

SUSE Linux Enterprise Desktop does not support compilation of 32-bit applications. It only offers runtime support for 32-bit binaries.

20.1 Runtime support

Important: Conflicts between application versions

If an application is available for both 32-bit and 64-bit environments, installing both versions may cause problems. In such cases, decide on one version to install to avoid potential runtime errors.

An exception to this rule is PAM (pluggable authentication modules). SUSE Linux Enterprise Desktop uses PAM in the authentication process as a layer that mediates between user and application. Always install both PAM versions on 64-bit operating systems that also run 32-bit applications.

For correct execution, every application requires a range of libraries. Because the names are identical for the 32-bit and 64-bit versions of these libraries, they must be differentiated from each other in another way.

To retain compatibility with 32-bit versions, 64-bit and 32-bit libraries are stored in the same location. The 32-bit version of `libc.so.6` is located under `/lib/libc.so.6` in both 32-bit and 64-bit environments.

All 64-bit libraries and object files are located in directories called `lib64`. The 64-bit object files normally found under `/lib` and `/usr/lib` are now found under `/lib64` and `/usr/lib64`. This means that space is available for 32-bit libraries under `/lib` and `/usr/lib`, so the file name for both versions can remain unchanged.

If the data content of 32-bit subdirectories under `/lib` does not depend on word size, they are not moved. This scheme conforms to LSB (Linux Standards Base) and FHS (Filesystem Hierarchy Standard).

20.2 Kernel specifications

The 64-bit kernels for AMD64/Intel 64 offer both a 64-bit and a 32-bit kernel ABI (application binary interface). The latter is identical to the ABI for the corresponding 32-bit kernel. This means that communication between both 32-bit and 64-bit applications with 64-bit kernels are identical.

The 32-bit system call emulation for 64-bit kernels does not support all the APIs used by system programs. This depends on the platform. For this reason, few applications, like **`lspci`**, must be compiled.

A 64-bit kernel can only load 64-bit kernel modules. You must compile 64-bit modules specifically for 64-bit kernels. It is not possible to use 32-bit kernel modules with 64-bit kernels.

Tip: Kernel-loadable modules

Certain applications require separate kernel-loadable modules. If you intend to use a 32-bit application in a 64-bit system environment, contact the provider of the application and SUSE. Make sure that the 64-bit version of the kernel-loadable module and the 32-bit compiled version of the kernel API are available for this module.

21 `journalctl`: query the systemd journal

Revision History

2024-05-13

`systemd` features its own logging system called *journal*. There is no need to run a `syslog`-based service, as all system events are written to the journal.

The journal itself is a system service managed by `systemd`. Its full name is `systemd-journald.service`. It collects and stores logging data by maintaining structured indexed journals based on logging information received from the kernel, user processes, standard input, and system service errors. The `systemd-journald` service is on by default:

```
> sudo systemctl status systemd-journald
systemd-journald.service - Journal Service
   Loaded: loaded (/usr/lib/systemd/system/systemd-journald.service; static)
   Active: active (running) since Mon 2014-05-26 08:36:59 EDT; 3 days ago
     Docs: man:systemd-journald.service(8)
           man:journald.conf(5)
 Main PID: 413 (systemd-journal)
   Status: "Processing requests..."
   CGroup: /system.slice/systemd-journald.service
           └─413 /usr/lib/systemd/systemd-journald
[...]
```

21.1 Making the journal persistent

The journal stores log data in `/run/log/journal/` by default. Because the `/run/` directory is volatile by nature, log data is lost at reboot. To make the log data persistent, create the directory `/var/log/journal/` and make sure it has the correct access modes and ownership, so the systemd-journald service can store its data. To switch to persistent logging, execute the following commands:

```
> sudo  mkdir /var/log/journal
> sudo  systemd-tmpfiles --create --prefix=/var/log/journal
> sudo  journalctl --flush
```

Any log data stored in `/run/log/journal/` will be flushed into `/var/log/journal/`.

21.2 **journalctl**: useful switches

This section introduces several common useful options to enhance the default **journalctl** behavior. All switches are described in the **journalctl** man page, **man 1 journalctl**.

Tip: Messages related to a specific executable

To show all journal messages related to a specific executable, specify the full path to the executable:

```
> sudo journalctl /usr/lib/systemd/systemd
```

-f

Shows only the most recent journal messages, and prints new log entries as they are added to the journal.

Prints the messages and jumps to the end of the journal, so that the latest entries are visible within the pager.

-r

Prints the messages of the journal in reverse order, so that the latest entries are listed first.

-k

Shows only kernel messages. This is equivalent to the field match `_TRANSPORT=kernel` (see *Section 21.3.3, "Filtering based on fields"*).

-u

Shows only messages for the specified `systemd` unit. This is equivalent to the field match `_SYSTEMD_UNIT=UNIT` (see *Section 21.3.3, "Filtering based on fields"*).

```
> sudo journalctl -u apache2
[...]
Jun 03 10:07:11 pinkiepie systemd[1]: Starting The Apache Webserver...
Jun 03 10:07:12 pinkiepie systemd[1]: Started The Apache Webserver.
```

21.3 Filtering the journal output

When called without switches, **journalctl** shows the full content of the journal, the oldest entries listed first. The output can be filtered by specific switches and fields.

21.3.1 Filtering based on a boot number

`journalctl` can filter messages based on a specific system boot. To list all available boots, run

```
> sudo journalctl --list-boots
-1 097ed2cd99124a2391d2cffab1b566f0 Mon 2014-05-26 08:36:56 EDT—Fri 2014-05-30 05:33:44
 EDT
 0 156019a44a774a0bb0148a92df4af81b Fri 2014-05-30 05:34:09 EDT—Fri 2014-05-30 06:15:01
 EDT
```

The first column lists the boot offset: `0` for the current boot, `-1` for the previous one, `-2` for the one before that, etc. The second column contains the boot ID followed by the limiting time stamps of the specific boot.

Show all messages from the current boot:

```
> sudo journalctl -b
```

If you need to see journal messages from the previous boot, add an offset parameter. The following example outputs the previous boot messages:

```
> sudo journalctl -b -1
```

Another way is to list boot messages based on the boot ID. For this purpose, use the _BOOT_ID field:

```
> sudo journalctl _BOOT_ID=156019a44a774a0bb0148a92df4af81b
```

21.3.2 Filtering based on time interval

You can filter the output of `journalctl` by specifying the starting and/or ending date. The date specification should be of the format `2014-06-30 9:17:16`. If the time part is omitted, midnight is assumed. If seconds are omitted, `:00` is assumed. If the date part is omitted, the current day is assumed. Instead of numeric expression, you can specify the keywords `yesterday`, `today` or `tomorrow`. They refer to midnight of the day before the current day, of the current day, or of the day after the current day. If you specify `now`, it refers to the current time. You can also specify relative times prefixed with `-` or `+`, referring to times before or after the current time.

Show only new messages since now, and update the output continuously:

```
> sudo journalctl --since "now" -f
```

Show all messages since last midnight till 3:20am:

```
> sudo journalctl --since "today" --until "3:20"
```

21.3.3 Filtering based on fields

You can filter the output of the journal by specific fields. The syntax of a field to be matched is `FIELD_NAME=MATCHED_VALUE`, such as `_SYSTEMD_UNIT=httpd.service`. You can specify multiple matches in a single query to filter the output messages even more. See **`man 7 systemd.journal-fields`** for a list of default fields.

Show messages produced by a specific process ID:

```
> sudo journalctl _PID=1039
```

Show messages belonging to a specific user ID:

```
# journalctl _UID=1000
```

Show messages from the kernel ring buffer (the same as **`dmesg`** produces):

```
> sudo journalctl _TRANSPORT=kernel
```

Show messages from the service's standard or error output:

```
> sudo journalctl _TRANSPORT=stdout
```

Show messages produced by a specified service only:

```
> sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service
```

If two different fields are specified, only entries that match both expressions at the same time are shown:

```
> sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=1488
```

If two matches refer to the same field, all entries matching either expression are shown:

```
> sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _SYSTEMD_UNIT=dbus.service
```

You can use the `+` separator to combine two expressions in a logical `OR`. The following example shows all messages from the Avahi service process with the process ID 1480 together with all messages from the D-Bus service:

```
> sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=1480 +
 _SYSTEMD_UNIT=dbus.service
```

21.4 Investigating systemd errors

This section introduces a simple example to illustrate how to find and fix the error reported by `systemd` during **`apache2`** start-up.

1. Try to start the apache2 service:

```
# systemctl start apache2
Job for apache2.service failed. See 'systemctl status apache2' and 'journalctl -xn'
 for details.
```

2. Let us see what the service's status says:

```
> sudo systemctl status apache2
apache2.service - The Apache Webserver
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
   Active: failed (Result: exit-code) since Tue 2014-06-03 11:08:13 CEST; 7min ago
  Process: 11026 ExecStop=/usr/sbin/start_apache2 -D SYSTEMD -DFOREGROUND \
           -k graceful-stop (code=exited, status=1/FAILURE)
```

The ID of the process causing the failure is 11026.

3. Show the verbose version of messages related to process ID 11026:

```
> sudo journalctl -o verbose _PID=11026
[...]
MESSAGE=AH00526: Syntax error on line 6 of /etc/apache2/default-server.conf:
[...]
MESSAGE=Invalid command 'DocumenttRoot', perhaps misspelled or defined by a module
[...]
```

4. Fix the typo inside `/etc/apache2/default-server.conf`, start the apache2 service, and print its status:

```
> sudo systemctl start apache2 && systemctl status apache2
apache2.service - The Apache Webserver
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
   Active: active (running) since Tue 2014-06-03 11:26:24 CEST; 4ms ago
  Process: 11026 ExecStop=/usr/sbin/start_apache2 -D SYSTEMD -DFOREGROUND
           -k graceful-stop (code=exited, status=1/FAILURE)
 Main PID: 11263 (httpd2-prefork)
   Status: "Processing requests..."
   CGroup: /system.slice/apache2.service
           ├─11263 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11280 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11281 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
```

```
            ├─11282 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
            ├─11283 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
            └─11285 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
```

21.5 Journald configuration

The behavior of the systemd-journald service can be adjusted by modifying `/etc/systemd/journald.conf`. This section introduces only basic option settings. For a complete file description, see **`man 5 journald.conf`**. You need to restart the journal for the changes to take effect with

```
> sudo systemctl restart systemd-journald
```

21.5.1 Changing the journal size limit

If the journal log data is saved to a persistent location (see *Section 21.1, "Making the journal persistent"*), it uses up to 10% of the file system the `/var/log/journal` resides on. For example, if `/var/log/journal` is located on a 30 GB `/var` partition, the journal may use up to 3 GB of the disk space. To change this limit, change (and uncomment) the `SystemMaxUse` option:

```
SystemMaxUse=50M
```

21.5.2 Forwarding the journal to `/dev/ttyX`

You can forward the journal to a terminal device to inform you about system messages on a preferred terminal screen, for example, `/dev/tty12`. Change the following journald options to

```
ForwardToConsole=yes
TTYPath=/dev/tty12
```

21.5.3 Forwarding the journal to syslog facility

Journald is backward compatible with traditional syslog implementations such as `rsyslog`. Make sure the following is valid:

- rsyslog is installed.

```
> sudo rpm -q rsyslog
rsyslog-7.4.8-2.16.x86_64
```

- rsyslog service is enabled.

```
> sudo systemctl is-enabled rsyslog
enabled
```

- Forwarding to syslog is enabled in `/etc/systemd/journald.conf`.

```
ForwardToSyslog=yes
```

21.6 Using YaST to filter the systemd journal

For an easy way of filtering the systemd journal (without dealing with the journalctl syntax), you can use the YaST journal module. After installing it with **`sudo zypper in yast2-journal`**, start it from YaST by selecting *System* › *Systemd Journal*. Alternatively, start it from command line by entering **`sudo yast2 journal`**.

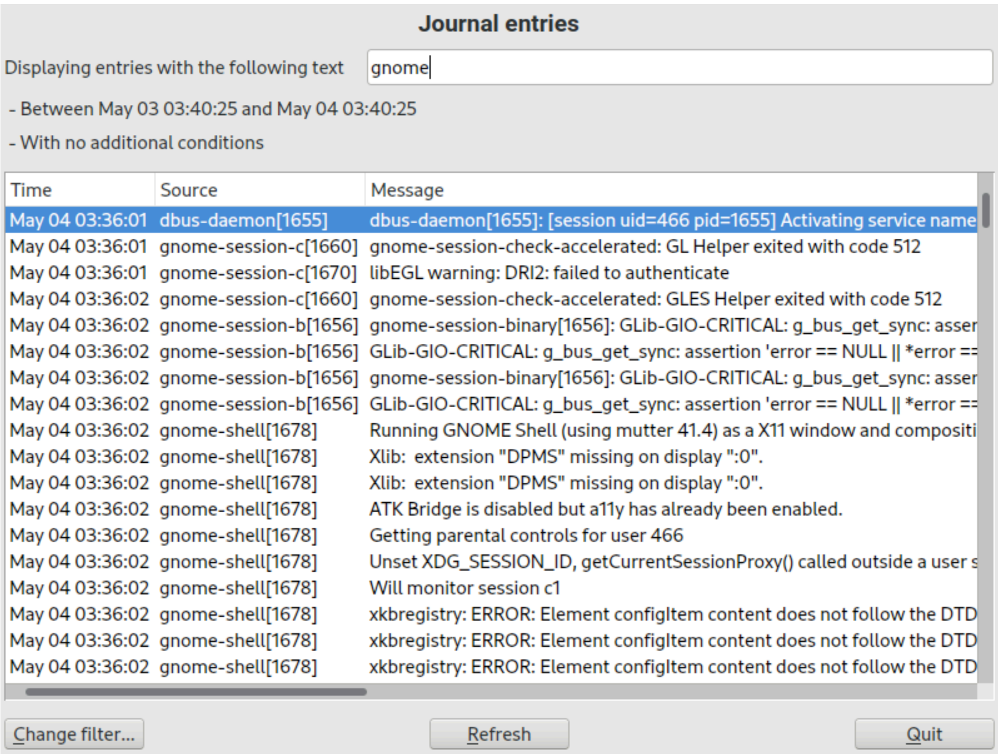


FIGURE 21.1: YAST SYSTEMD JOURNAL

The module displays the log entries in a table. The search box on top allows you to search for entries that contain certain characters, similar to using **grep**. To filter the entries by date and time, unit, file or priority, click *Change filters* and set the respective options.

21.7 Viewing logs in GNOME

You can view the journal with *GNOME Logs*. Start it from the application menu. To view system log messages, it needs to be run as root, for example, with **xdg-su gnome-logs**. This command can be executed when pressing `Alt`–`F2`.

22 **`update-alternatives`**: managing multiple versions of commands and files

Revision History

2024-05-13

Often, there are several versions of the same tool installed on a system. To give administrators a choice and to make it possible to install and use different versions side by side, the alternatives system allows managing such versions consistently.

22.1 Overview

On SUSE Linux Enterprise Desktop, several programs perform the same or similar tasks. For example, if Java 1.7 and Java 1.8 are both installed on the system, the alternatives system script (**`update-alternatives`**) is called from inside the RPM package. By default, the alternatives system will refer to version 1.8: higher versions also have a higher priority. However, the administrator can change the default and can point the generic name to version 1.7.

The following terminology is used in this chapter:

TERMINOLOGY

Administrative directory
: The default `/var/lib/rpm/alternatives` directory contains information about the current state of alternatives.

Alternative
: The name of a specific file in the file system, which can be made accessible via a generic name using the alternatives system.

Alternatives directory
: The default `/etc/alternatives` directory containing symbolic links.

Generic name
: A name (for example, **`/usr/bin/edit`**) that refers to one file out of several available using the alternatives system.

Link group
: A set of related symbolic links that can be updated as a group.

Master link

The link in a link group that determines how the other links in the group are configured.

Slave link

A link in a link group controlled by the master link.

Symbolic link (symlink)

A file that is a reference to another file in the same file system. The alternatives system uses symbolic links in the alternatives directory to switch between versions of a file.

Symbolic links in the alternatives directory can be modified by the administrator through the **update-alternatives** command.

The alternatives system provides the **update-alternatives** command to create, remove, maintain and show information about symbolic links. While these symbolic links normally point to commands, they can also point to JAR archives, man pages, and other files. Examples in this chapter use commands and man pages, but they are also applicable to other file types.

The alternatives system uses the alternatives directory to collect links to possible alternatives. When a new package with an alternative is installed, the new alternative is added to the system. Whether the new package's alternative is selected as the default depends on its priority and on the mode that is set. Packages with a higher version also have a higher priority. The alternatives system can operate in two modes:

- **Automatic mode.** In this mode, the alternatives system ensures that the links in the group point to the highest priority alternatives appropriate for the group.
- **Manual mode.** In this mode, the alternatives system does not make any changes to the system administrator's settings.

For example, the **java** command has the following link hierarchy in the alternatives system:

EXAMPLE 22.1: ALTERNATIVES SYSTEM OF THE java COMMAND

```
/usr/bin/java ❶
 -> /etc/alternatives/java ❷
     -> /usr/lib64/jvm/jre-10-openjdk/bin/java ❸
```

❶ The generic name.

❷ The symbolic link in the alternatives directory.

❸ One of the alternatives.

22.2 Use cases

By default, the **update-alternatives** script is called from inside an RPM package. When a package is installed or removed, the script takes care of all its symbolic links. But you can run it manually from the command line for:

- displaying the current alternatives for a generic name.
- changing the defaults of an alternative.
- creating a set of related files for an alternative.

22.3 Getting an overview of alternatives

To retrieve the names of all configured alternatives, use:

```
> ls /var/lib/alternatives
```

To get an overview of all configured alternatives and their values, use

```
> sudo update-alternatives --get-selections
asadmin                        auto     /usr/bin/asadmin-2.7
awk                            auto     /usr/bin/gawk
chardetect                     auto     /usr/bin/chardetect-3.6
dbus-launch                    auto     /usr/bin/dbus-launch.x11
default-displaymanager         auto     /usr/lib/X11/displaymanagers/gdm
[...]
```

22.4 Viewing details on specific alternatives

The easiest way to check the alternatives is to follow the symbolic links of your command. For example, to find out what the **java** command is referring to, use the following command:

```
> readlink --canonicalize /usr/bin/java
/usr/lib64/jvm/jre-10-openjdk/bin/java
```

If you see the same path (in our example, it is `/usr/bin/java`), there are no alternatives available for this command.

To see the full alternatives (including slaves), use the `--display` option:

```
> sudo update-alternatives --display java
```

```
java - auto mode
  link best version is /usr/lib64/jvm/jre-1.8.0-openjdk/bin/java
  link currently points to /usr/lib64/jvm/jre-1.8.0-openjdk/bin/java
  link java is /usr/bin/java
  slave java.1.gz is /usr/share/man/man1/java.1.gz
  slave jre is /usr/lib64/jvm/jre
  slave jre_exports is /usr/lib64/jvm-exports/jre
  slave keytool is /usr/bin/keytool
  slave keytool.1.gz is /usr/share/man/man1/keytool.1.gz
  slave orbd is /usr/bin/orbd
  slave orbd.1.gz is /usr/share/man/man1/orbd.1.gz
[...]
```

22.5 Setting the default version of alternatives

By default, commands in `/usr/bin` refer to the alternatives directory with the highest priority. For example, by default, the command **`java`** shows the following version number:

```
> java -version
openjdk version "10.0.1" 2018-04-17
OpenJDK Runtime Environment (build 10.0.1+10-suse-lp150.1.11-x8664)
OpenJDK 64-Bit Server VM (build 10.0.1+10-suse-lp150.1.11-x8664, mixed mode)
```

To change the default **`java`** command to refer to a previous version, run:

```
> sudo update-alternatives --config java
root's password:
There are 2 choices for the alternative java (providing /usr/bin/java).

  Selection    Path                                       Priority   Status
------------------------------------------------------------
* 0            /usr/lib64/jvm/jre-10-openjdk/bin/java      2005      auto mode
  1            /usr/lib64/jvm/jre-1.8.0-openjdk/bin/java   1805      manual mode
  2            /usr/lib64/jvm/jre-10-openjdk/bin/java      2005      manual mode
  3            /usr/lib64/jvm/jre-11-openjdk/bin/java      0         manual mode

Press <enter> to keep the current choice[*], or type selection number:
```

Depending on your system and installed versions, the exact Java version number will be different. After you have selected `1`, **`java`** shows the following version number:

```
> java -version
java version "1.8.0_171"
OpenJDK Runtime Environment (IcedTea 3.8.0) (build 1.8.0_171-b11 suse-lp150.2.3.1-x86_64)
```

```
OpenJDK 64-Bit Server VM (build 25.171-b11, mixed mode)
```

Also, keep in mind the following points:

- When working in manual mode and installing another Java version, the alternatives system neither touches the links nor changes the generic name.
- When working in automatic mode and installing another Java version, the alternatives system changes the Java master link and all slave links (as you can see in *Section 22.4, "Viewing details on specific alternatives"*). To check the master-slave relationships, use:

  ```
  > sudo update-alternatives --display java
  ```

22.6 Installing custom alternatives

This section describes how to set up custom alternatives on a system.

Warning: No custom alternatives for python3

Do not install custom alternatives for python3. **`/usr/bin/python3`** does not have update alternatives and always points to specific tested versions. Creating a custom python3 alternative pointing to a different version—such as python 3.11—breaks dependent system tools.

The example makes the following assumptions:

- There are two scripts, **`foo-2`** and **`foo-3`**, with similar functionality.
- The scripts are stored in the `/usr/local/bin` directory to avoid any conflicts with the system tools in `/usr/bin`.
- There is a master link **`foo`** that points to either **`foo-2`** or **`foo-3`**.

To provide alternatives on your system, follow these steps:

1. Copy your scripts into the `/usr/local/bin` directory.
2. Make the scripts executable:

   ```
   > sudo chmod +x /usr/local/bin/foo-{2,3}
   ```

3. Run **update-alternatives** for both scripts:

```
> sudo update-alternatives --install \
   /usr/local/bin/foo ❶\
   foo ❷\
   /usr/local/bin/foo-2 ❸\
   200 ❹
> sudo update-alternatives --install \
   /usr/local/bin/foo ❶\
   foo ❷\
   /usr/local/bin/foo-3 ❸\
   300 ❹
```

The options after `--install` have the following meanings:

❶ The generic name. To avoid confusion, this is normally the script name without any version numbers.

❷ The name of the master link. Must be the same.

❸ The path to the original scripts located in `/usr/local/bin`.

❹ The priority. We give **foo-2** a lower priority than **foo-3**. It is good practice to use a significant number increase to separate priorities. For example, a priority of 200 for **foo-2** and 300 for **foo-3**.

4. Check the master link:

```
> sudo update-alternatives --display foo
foo - auto mode
  link best version is /usr/local/bin/foo-3
  link currently points to /usr/local/bin/foo-3
  link foo is /usr/local/bin/foo
/usr/local/bin/foo-2 - priority 200
/usr/local/bin/foo-3 - priority 300
```

After you completed the described steps, you can use the master link **/usr/local/bin/foo**.

If needed, you can install additional alternatives. To remove an alternative, use the following command:

```
> sudo update-alternatives --remove foo /usr/local/bin/foo-2
```

After this script has been removed, the alternatives system for the foo group looks like this:

```
> sudo update-alternatives --display foo
foo - auto mode
```

```
  link best version is /usr/local/bin/foo-3
  link currently points to /usr/local/bin/foo-3
  link foo is /usr/local/bin/foo
/usr/local/bin/foo-3 - priority 300
```

22.7 Defining dependent alternatives

If you have alternatives, the script itself is not enough. Most commands are not stand-alone —they ship with additional files, such as extensions, configurations or man pages. To create alternatives which are dependent on a master link, use *slave alternatives*.

Let us assume we want to extend our example in *Section 22.6, "Installing custom alternatives"* and provide man pages and configuration files:

- Two man pages, `foo-2.1.gz` and `foo-3.1.gz` stored in the `/usr/local/man/man1` directory.
- Two configuration files, `foo-2.conf` and `foo-3.conf`, stored in `/etc`.

Follow these steps to add the additional files to your alternatives:

1. Copy the configuration files into `/etc`:

   ```
   > sudo cp foo-{2,3}.conf /etc
   ```

2. Copy the man pages into the `/usr/local/man/man1` directory:

   ```
   > sudo cp foo-{2,3}.1.gz /usr/local/man/man1/
   ```

3. Add the slave links to the main scripts with the `--slave` option:

   ```
   > sudo update-alternatives --install \
      /usr/local/bin/foo foo /usr/local/bin/foo-2 200 \
      --slave /usr/local/man/man1/foo.1.gz \
      foo.1.gz \
      /usr/local/man/man1/foo-2.1.gz \
      --slave /etc/foo.conf \
      foo.conf \
      /etc/foo-2.conf
   > sudo update-alternatives --install \
      /usr/local/bin/foo foo /usr/local/bin/foo-3 300 \
      --slave /usr/local/man/man1/foo.1.gz \
      foo.1.gz \
      /usr/local/man/man1/foo-3.1.gz \
   ```

```
    --slave /etc/foo.conf \
    foo.conf \
    /etc/foo-3.conf
```

4. Check the master link:

```
foo - auto mode
  link best version is /usr/local/bin/foo-3
  link currently points to /usr/local/bin/foo-3
  link foo is /usr/local/bin/foo
  slave foo.1.gz is /usr/local/man/man1/foo.1.gz
  slave foo.conf is /etc/foo.conf
/usr/local/bin/foo-2 - priority 200
  slave foo.1.gz: /usr/local/man/man1/foo-2.1.gz
  slave foo.conf: /etc/foo-2.conf
/usr/local/bin/foo-3 - priority 300
  slave foo.1.gz: /usr/local/man/man1/foo-3.1.gz
  slave foo.conf: /etc/foo-3.conf
```

If you change the links with **`update-alternatives --config foo`** to **`foo-2`**, then all slave links will change as well.

23 Basic networking

Revision History

2024-06-21

Linux offers the necessary networking tools and features for integration into all types of network structures. Network access using a network card can be configured with YaST. Manual configuration is also possible. In this chapter, only the fundamental mechanisms and the relevant network configuration files are covered.

Linux and other Unix operating systems use the TCP/IP protocol. It is not a single network protocol, but a family of network protocols that offer multiple services. The protocols listed in *Several protocols in the TCP/IP protocol family* are provided for exchanging data between two machines via TCP/IP. Networks combined by TCP/IP, comprising a worldwide network, are also called “the Internet.”

RFC stands for *Request for Comments*. RFCs are documents that describe Internet protocols and implementation procedures for the operating system and its applications. The RFC documents describe the setup of Internet protocols. For more information about RFCs, see https://datatracker.ietf.org/ ↗.

SEVERAL PROTOCOLS IN THE TCP/IP PROTOCOL FAMILY

TCP

Transmission Control Protocol: a connection-oriented secure protocol. The data to transmit is first sent by the application as a stream of data and converted into the appropriate format by the operating system. The data arrives at the respective application on the destination host in the original data stream format it was initially sent. TCP determines whether any data has been lost or jumbled during the transmission. TCP is implemented wherever the data sequence matters.

UDP

User Datagram Protocol: a connectionless, insecure protocol. The data to transmit is sent in the form of packets generated by the application. The order in which the data arrives at the recipient is not guaranteed and data loss is possible. UDP is suitable for record-oriented applications. It features a smaller latency period than TCP.

ICMP

Internet Control Message Protocol: this is not a protocol for the end user, but a special control protocol that issues error reports and can control the behavior of machines participating in TCP/IP data transfer. In addition, it provides a special echo mode that can be viewed using the program ping.

IGMP

Internet Group Management Protocol: this protocol controls machine behavior when implementing IP multicast.

As shown in *Figure 23.1, “Simplified layer model for TCP/IP”*, data exchange takes place in different layers. The actual network layer is the insecure data transfer via IP (Internet protocol). On top of IP, TCP (transmission control protocol) guarantees, to a certain extent, security of the data transfer. The IP layer is supported by the underlying hardware-dependent protocol, such as Ethernet.

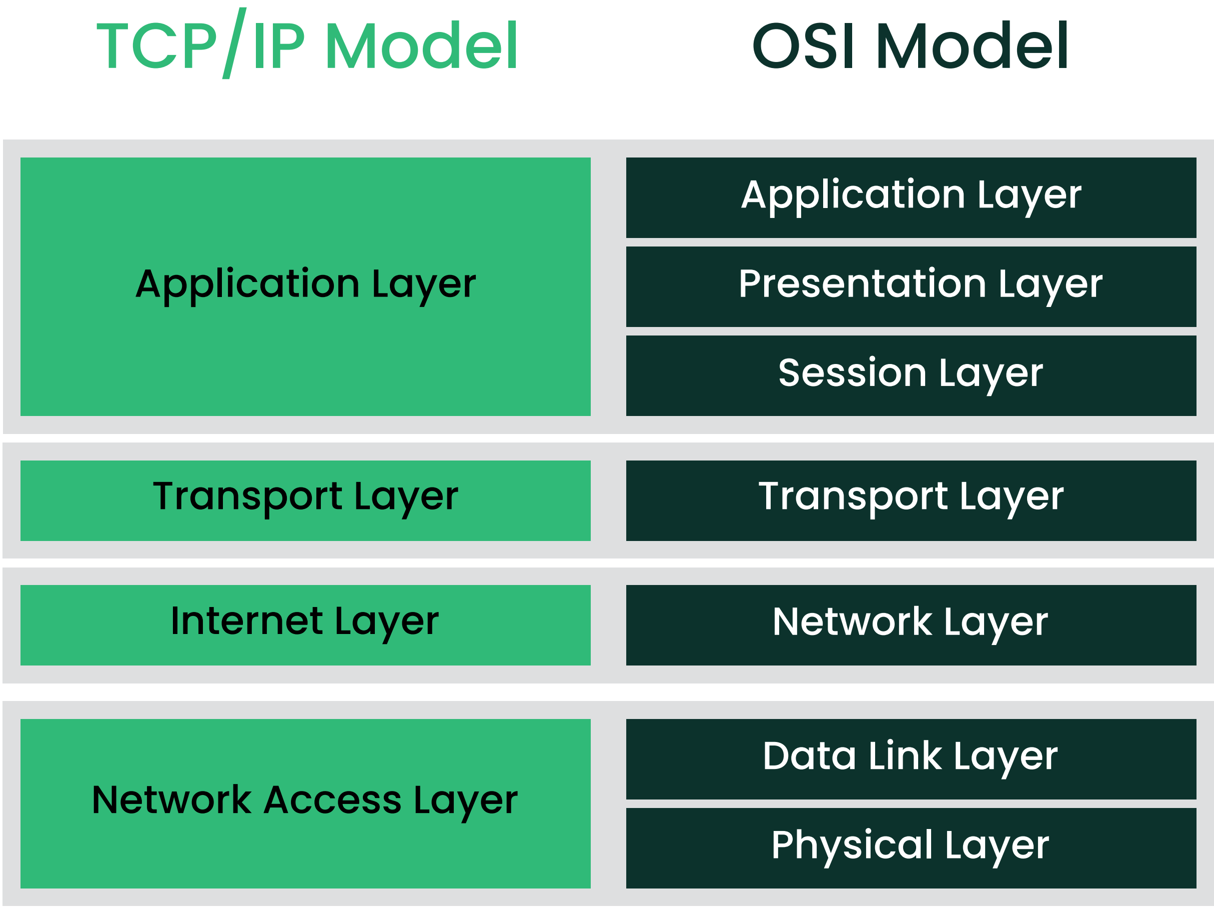


FIGURE 23.1: SIMPLIFIED LAYER MODEL FOR TCP/IP

The diagram provides one or two examples for each layer. The layers are ordered according to *abstraction levels*. The lowest layer is close to the hardware. The uppermost layer, however, is almost a complete abstraction from the hardware. Every layer has its own special function. The special functions of each layer are implicit in their description. The data link and physical layers represent the physical network used, such as Ethernet.

Almost all hardware protocols work on a packet-oriented basis. The data to transmit is collected into *packets* (it cannot be sent all at once). The maximum size of a TCP/IP packet is approximately 64 KB. Packets are normally small, as the network hardware can be a limiting factor. The maximum size of a data packet on Ethernet is about fifteen hundred bytes. The size of a TCP/IP packet is limited to this amount when the data is sent over Ethernet. If more data is transferred, more data packets need to be sent by the operating system.

For the layers to serve their designated functions, additional information regarding each layer must be saved in the data packet. This takes place in the *header* of the packet. Every layer attaches a small block of data, called the protocol header, to the front of each emerging packet. A sample TCP/IP data packet traveling over an Ethernet cable is illustrated in *Figure 23.2, "TCP/IP Ethernet packet"*. The proof sum is located at the end of the packet, not at the beginning. This simplifies things for the network hardware.

FIGURE 23.2: TCP/IP ETHERNET PACKET

When an application sends data over the network, the data passes through each layer, all implemented in the Linux kernel except the physical layer. Each layer handles preparing the data so it can be passed to the next layer. The lowest layer is ultimately responsible for sending the data. The entire procedure is reversed when data is received. Like the layers of an onion, in each layer the protocol headers are removed from the transported data. Finally, the transport layer handles making the data available for use by the applications at the destination. In this manner, one layer only communicates with the layer directly above or below it. For applications, it is irrelevant whether data is transmitted via a wireless or wired connection. Likewise, it is irrelevant for the data line which kind of data is transmitted, if packets are in the correct format.

23.1 IP addresses and routing

The discussion in this section is limited to IPv4 networks. For information about IPv6 protocol, the successor to IPv4, refer to *Section 23.2, "IPv6—the next generation Internet"*.

23.1.1 IP addresses

Every computer on the Internet has a unique 32-bit address. These 32 bits (or 4 bytes) are normally written as illustrated in the second row in *Example 23.1, "Writing IP addresses"*.

EXAMPLE 23.1: WRITING IP ADDRESSES

```
IP Address (binary):  11000000 10101000 00000000 00010100
```

```
IP Address (decimal):      192.     168.       0.      20
```

In decimal form, the four bytes are written in the decimal number system, separated by periods. The IP address is assigned to a host or a network interface. It can be used only once throughout the world. There are exceptions to this rule, but these are not relevant to the following passages.

The points in IP addresses indicate the hierarchical system. Until the 1990s, IP addresses were strictly categorized in classes. However, this system proved too inflexible and was discontinued. Now, *classless routing* (CIDR, classless interdomain routing) is used.

23.1.2 Netmasks and routing

Netmasks are used to define the address range of a subnet. If two hosts are in the same subnet, they can reach each other directly. If they are not in the same subnet, they need the address of a gateway that handles all the traffic for the subnet. To check if two IP addresses are in the same subnet, simply "AND" both addresses with the netmask. If the result is identical, both IP addresses are in the same local network. If there are differences, the remote IP address, and thus the remote interface, can only be reached over a gateway.

To understand how the netmask works, look at *Example 23.2, "Linking IP addresses to the netmask"*. The netmask consists of 32 bits that identify how much of an IP address belongs to the network. All those bits that are `1` mark the corresponding bit in the IP address as belonging to the network. All bits that are `0` mark bits inside the subnet. This means that the more bits are `1`, the smaller the subnet is. Because the netmask always consists of several successive `1` bits, it is also possible to count the number of bits in the netmask. In *Example 23.2, "Linking IP addresses to the netmask"* the first net with 24 bits could also be written as `192.168.0.0/24`.

EXAMPLE 23.2: LINKING IP ADDRESSES TO THE NETMASK

```
IP address (192.168.0.20):  11000000 10101000 00000000 00010100
Netmask   (255.255.255.0):  11111111 11111111 11111111 00000000
---------------------------------------------------------------
Result of the link:         11000000 10101000 00000000 00000000
In the decimal system:           192.     168.       0.       0

IP address (213.95.15.200): 11010101 10111111 00001111 11001000
Netmask    (255.255.255.0): 11111111 11111111 11111111 00000000
---------------------------------------------------------------
Result of the link:         11010101 10111111 00001111 00000000
In the decimal system:           213.      95.      15.       0
```

To give another example: all machines connected with the same Ethernet cable are normally located in the same subnet and are directly accessible. Even when the subnet is physically divided by switches or bridges, these hosts can still be reached directly.

IP addresses outside the local subnet can only be reached if a gateway is configured for the target network. In the most common case, there is only one gateway that handles all traffic that is external. However, it is also possible to configure several gateways for different subnets.

If a gateway has been configured, all external IP packets are sent to the appropriate gateway. This gateway then attempts to forward the packets in the same manner—from host to host—until it reaches the destination host or the packet's TTL (time to live) expires.

SPECIFIC ADDRESSES

Base Network Address

This is the netmask AND any address in the network, as shown in *Example 23.2, "Linking IP addresses to the netmask"* under `Result`. This address cannot be assigned to any hosts.

Broadcast Address

This could be paraphrased as: "Access all hosts in this subnet." To generate this, the netmask is inverted in binary form and linked to the base network address with a logical OR. The above example therefore results in 192.168.0.255. This address cannot be assigned to any hosts.

Local Host

The address `127.0.0.1` is assigned to the "loopback device" on each host. A connection can be set up to your own machine with this address and with all addresses from the complete `127.0.0.0/8` loopback network as defined with IPv4. With IPv6 there is only one loopback address (`::1`).

Because IP addresses must be unique all over the world, you cannot select random addresses. There are three address domains to use to set up a private IP-based network. These cannot get any connection from the rest of the Internet, because they cannot be transmitted over the Internet. These address domains are specified in RFC 1597 and listed in *Table 23.1, "Private IP address domains"*.

TABLE 23.1: PRIVATE IP ADDRESS DOMAINS

Network/Netmask	Domain
`10.0.0.0`/`255.0.0.0`	`10.x.x.x`

Network/Netmask	Domain
172.16.0.0/255.240.0.0	172.16.x.x – 172.31.x.x
192.168.0.0/255.255.0.0	192.168.x.x

23.2 IPv6—the next generation Internet

Because of the emergence of the World Wide Web (WWW), the Internet has experienced explosive growth, with an increasing number of computers communicating via TCP/IP in the past fifteen years. Since Tim Berners-Lee at CERN (https://public.web.cern.ch) invented the WWW in 1990, the number of Internet hosts has grown from a few thousand to about a hundred million.

As mentioned, an IPv4 address consists of only 32 bits. Also, a few IP addresses are lost—they cannot be used because of the way networks are organized. The number of addresses available in your subnet is two to the power of the number of bits, minus two. A subnet has, for example, 2, 6 or 14 addresses available. To connect 128 hosts to the Internet, for example, you need a subnet with 256 IP addresses, from which only 254 are usable, because two IP addresses are needed for the structure of the subnet itself: the broadcast and the base network address.

Under the current IPv4 protocol, DHCP or NAT (network address translation) are the typical mechanisms used to circumvent the potential address shortage. Combined with the convention to keep private and public address spaces separate, these methods can certainly mitigate the shortage. To set up a host in an IPv4 network, you need several address items, such as the host's own IP address, the subnetmask, the gateway address, and maybe a name server address. All these items need to be known and cannot be derived from somewhere else.

With IPv6, both the address shortage and the complicated configuration should be a thing of the past. The following sections tell more about the improvements and benefits brought by IPv6 and about the transition from the old protocol to the new one.

23.2.1 Advantages

The most important and most visible improvement brought by the IPv6 protocol is the enormous expansion of the available address space. An IPv6 address is made up of 128 bit values instead of the traditional 32 bits. This provides for as many as several quadrillion IP addresses.

However, IPv6 addresses are not only different from their predecessors with regard to their length. They also have a different internal structure that may contain more specific information about the systems and the networks to which they belong. More details about this are found in *Section 23.2.2, "Address types and structure"*.

The following is a list of other advantages of the IPv6 protocol:

Autoconfiguration
: IPv6 makes the network "plug and play" capable, which means that a newly configured system integrates into the (local) network without any manual configuration. The new host uses its automatic configuration mechanism to derive its own address from the information made available by the neighboring routers, relying on a protocol called the *neighbor discovery* (ND) protocol. This method does not require any intervention on the administrator's part and there is no need to maintain a central server for address allocation—an additional advantage over IPv4, where automatic address allocation requires a DHCP server.

  Nevertheless if a router is connected to a switch, the router should send periodic advertisements with flags telling the hosts of a network how they should interact with each other. For more information, see RFC 2462 and the `radvd.conf(5)` man page, and RFC 3315.

Mobility
: IPv6 makes it possible to assign several addresses to one network interface at the same time. This allows users to access several networks easily, something that could be compared with the international roaming services offered by mobile phone companies. When you take your mobile phone abroad, the phone automatically logs in to a foreign service when it enters the corresponding area, so you can be reached under the same number everywhere and can place an outgoing call, as you would in your home area.

Secure communication
: With IPv4, network security is an add-on function. IPv6 includes IPsec as one of its core features, allowing systems to communicate over a secure tunnel to avoid eavesdropping by outsiders on the Internet.

Backward compatibility
: Realistically, it would be impossible to switch the entire Internet from IPv4 to IPv6 at one time. Therefore, it is crucial that both protocols can coexist not only on the Internet, but also on one system. This is ensured by compatible addresses (IPv4 addresses can easily be translated into IPv6 addresses) and by using several tunnels. See *Section 23.2.3, "Coexistence*

of IPv4 and IPv6". Also, systems can rely on a *dual stack IP* technique to support both protocols at the same time, meaning that they have two network stacks that are separate, such that there is no interference between the two protocol versions.

Custom tailored services through multicasting

With IPv4, certain services, such as SMB, need to broadcast their packets to all hosts in the local network. IPv6 allows a much more fine-grained approach by enabling servers to address hosts through *multicasting*, that is by addressing several hosts as parts of a group. This is different from addressing all hosts through *broadcasting* or each host individually through *unicasting*. Which hosts are addressed as a group may depend on the concrete application. There are specific predefined groups to address all name servers (the *all name servers multicast group*), for example, or all routers (the *all routers multicast group*).

23.2.2 Address types and structure

As mentioned, the current IP protocol has two major limitations: there is an increasing shortage of IP addresses and configuring the network and maintaining the routing tables is becoming a more complex and burdensome task. IPv6 solves the first problem by expanding the address space to 128 bits. The second one is mitigated by introducing a hierarchical address structure combined with sophisticated techniques to allocate network addresses, and *multihoming* (the ability to assign several addresses to one device, giving access to several networks).

When dealing with IPv6, it is useful to know about three different types of addresses:

Unicast

Addresses of this type are associated with exactly one network interface. Packets with such an address are delivered to only one destination. Accordingly, unicast addresses are used to transfer packets to individual hosts on the local network or the Internet.

Multicast

Addresses of this type relate to a group of network interfaces. Packets with such an address are delivered to all destinations that belong to the group. Multicast addresses are mainly used by certain network services to communicate with certain groups of hosts in a well-directed manner.

Anycast

Addresses of this type are related to a group of interfaces. Packets with such an address are delivered to the member of the group that is closest to the sender, according to the principles of the underlying routing protocol. Anycast addresses are used to make it easier for hosts to find out about servers offering certain services in the given network area. All servers of the same type have the same anycast address. Whenever a host requests a service, it receives a reply from the server with the closest location, as determined by the routing protocol. If this server should fail, the protocol automatically selects the second closest server, then the third one, and so forth.

An IPv6 address is made up of eight four-digit fields, each representing 16 bits, written in hexadecimal notation. They are separated by colons (`:`). Any leading zero bytes within a given field may be dropped, but zeros within the field or at its end may not. Another convention is that more than four consecutive zero bytes may be collapsed into a double colon. However, only one such `::` is allowed per address. This kind of shorthand notation is shown in *Example 23.3, "Sample IPv6 address"*, where all three lines represent the same address.

EXAMPLE 23.3: SAMPLE IPV6 ADDRESS

```
fe80 : 0000 : 0000 : 0000 : 0000 : 10 : 1000 : 1a4
fe80 :    0 :    0 :    0 :    0 : 10 : 1000 : 1a4
fe80 :                           : 10 : 1000 : 1a4
```

Each part of an IPv6 address has a defined function. The first bytes form the prefix and specify the type of address. The center part is the network portion of the address, but it may be unused. The end of the address forms the host part. With IPv6, the netmask is defined by indicating the length of the prefix after a slash at the end of the address. An address, as shown in *Example 23.4, "IPv6 address specifying the prefix length"*, contains the information that the first 64 bits form the network part of the address and the last 64 form its host part. In other words, the `64` means that the netmask is filled with 64 1-bit values from the left. As with IPv4, the IP address is combined with AND with the values from the netmask to determine whether the host is located in the same subnet or in another one.

EXAMPLE 23.4: IPV6 ADDRESS SPECIFYING THE PREFIX LENGTH

```
fe80::10:1000:1a4/64
```

IPv6 knows about several predefined types of prefixes. Certain are shown in *IPv6 prefixes*.

IPV6 PREFIXES

`00`

IPv4 addresses and IPv4 over IPv6 compatibility addresses. These are used to maintain compatibility with IPv4. Their use still requires a router able to translate IPv6 packets into IPv4 packets. Several special addresses, such as the one for the loopback device, have this prefix as well.

`2` or `3` as the first digit

Aggregatable global unicast addresses. As is the case with IPv4, an interface can be assigned to form part of a certain subnet. Currently, there are the following address spaces: `2001::/16` (production quality address space) and `2002::/16` (6to4 address space).

`fe80::/10`

Link-local addresses. Addresses with this prefix should not be routed and should therefore only be reachable from within the same subnet.

`fec0::/10`

Site-local addresses. These may be routed, but only within the network of the organization to which they belong. In effect, they are the IPv6 equivalent of the current private network address space, such as `10.x.x.x`.

`ff`

These are multicast addresses.

A unicast address consists of three basic components:

Public topology

The first part (which also contains one of the prefixes mentioned above) is used to route packets through the public Internet. It includes information about the company or institution that provides the Internet access.

Site topology

The second part contains routing information about the subnet to which to deliver the packet.

Interface ID

The third part identifies the interface to which to deliver the packet. This also allows for the MAC to form part of the address. Given that the MAC is a globally unique, fixed identifier coded into the device by the hardware maker, the configuration procedure is simplified. In fact, the first 64 address bits are consolidated to form the `EUI-64` token, with the last 48 bits taken from the MAC, and the remaining 24 bits containing special information about the token type. This also makes it possible to assign an `EUI-64` token to interfaces that do not have a MAC, such as those based on point-to-point protocol (PPP).

On top of this basic structure, IPv6 distinguishes between five different types of unicast addresses:

`::` (unspecified)
: This address is used by the host as its source address when the interface is initialized for the first time (at which point, the address cannot yet be determined by other means).

`::1` (loopback)
: The address of the loopback device.

IPv4 compatible addresses
: The IPv6 address is formed by the IPv4 address and a prefix consisting of 96 zero bits. This type of compatibility address is used for tunneling (see *Section 23.2.3, "Coexistence of IPv4 and IPv6"*) to allow IPv4 and IPv6 hosts to communicate with others operating in a pure IPv4 environment.

IPv4 addresses mapped to IPv6
: This type of address specifies a pure IPv4 address in IPv6 notation.

Local addresses
: There are two address types for local use:

  link-local
  : This type of address can only be used in the local subnet. Packets with a source or target address of this type should not be routed to the Internet or other subnets. These addresses contain a special prefix (`fe80::/10`) and the interface ID of the network card, with the middle part consisting of zero bytes. Addresses of this type are used during automatic configuration to communicate with other hosts belonging to the same subnet.

  site-local
  : Packets with this type of address may be routed to other subnets, but not to the wider Internet—they must remain inside the organization's own network. Such addresses are used for intranets and are an equivalent of the private address space defined by IPv4. They contain a special prefix (`fec0::/10`), the interface ID, and a 16-bit field specifying the subnet ID. Again, the rest is filled with zero bytes.

As a new feature introduced with IPv6, each network interface normally gets several IP addresses, with the advantage that several networks can be accessed through the same interface. One of these networks can be configured automatically using the MAC and a known prefix with the result that all hosts on the local network can be reached when IPv6 is enabled (using the link-local address). With the MAC forming part of it, any IP address used in the world is unique. The only variable parts of the address are those specifying the *site topology* and the *public topology*, depending on the actual network in which the host is currently operating.

For a host to go back and forth between different networks, it needs at least two addresses. One of them, the *home address*, not only contains the interface ID but also an identifier of the home network to which it normally belongs (and the corresponding prefix). The home address is a static address and, as such, it does not normally change. Still, all packets destined to the mobile host can be delivered to it, regardless of whether it operates in the home network or somewhere outside. This is made possible by new features introduced with IPv6, such as *stateless autoconfiguration* and *neighbor discovery*. In addition to its home address, a mobile host gets one or more additional addresses that belong to the foreign networks where it is roaming. These are called *care-of* addresses. The home network has a facility that forwards any packets destined to the host when it is roaming outside. In an IPv6 environment, this task is performed by the *home agent*, which takes all packets destined to the home address and relays them through a tunnel. Those packets destined to the care-of address are directly transferred to the mobile host without any special detours.

23.2.3 Coexistence of IPv4 and IPv6

The migration of all hosts connected to the Internet from IPv4 to IPv6 is a gradual process. Both protocols can coexist for a certain time to come. The coexistence on one system is guaranteed where there is a *dual stack* implementation of both protocols. That still leaves the question of how an IPv6 enabled host should communicate with an IPv4 host and how IPv6 packets should be transported by the current networks, which are predominantly IPv4-based. The best solutions offer tunneling and compatibility addresses (see *Section 23.2.2, "Address types and structure"*).

IPv6 hosts that are isolated in the (worldwide) IPv4 network can communicate through tunnels: IPv6 packets are encapsulated as IPv4 packets to move them across an IPv4 network. Such a connection between two IPv4 hosts is called a *tunnel*. To achieve this, packets must include the IPv6 destination address (or the corresponding prefix) and the IPv4 address of the remote host at the receiving end of the tunnel. A basic tunnel can be configured manually according to an agreement between the hosts' administrators. This is also called *static tunneling*.

However, the configuration and maintenance of static tunnels is often too labor-intensive to use them for daily communication needs. Therefore, IPv6 provides for three different methods of *dynamic tunneling*:

6over4

IPv6 packets are automatically encapsulated as IPv4 packets and sent over an IPv4 network capable of multicasting. IPv6 is tricked into seeing the whole network (Internet) as a local area network (LAN). This makes it possible to determine the receiving end of the IPv4 tunnel automatically. However, this method does not scale well and is also hampered because IP multicasting is far from widespread on the Internet. Therefore, it only provides a solution for smaller corporate or institutional networks where multicasting can be enabled. The specifications for this method are laid down in RFC 2529.

6to4

With this method, IPv4 addresses are automatically generated from IPv6 addresses, enabling isolated IPv6 hosts to communicate over an IPv4 network. However, several problems have been reported regarding the communication between those isolated IPv6 hosts and the Internet. The method is described in RFC 3056.

IPv6 tunnel broker

This method relies on special servers that provide dedicated tunnels for IPv6 hosts. It is described in RFC 3053.

23.2.4 Configuring IPv6

To configure IPv6, you normally do not need to make any changes on the individual workstations. IPv6 is enabled by default. To disable or enable IPv6 on an installed system, use the YaST *Network Settings* module. On the *Global Options* tab, select or deselect the *Enable IPv6* option as necessary. To enable it temporarily until the next reboot, enter `modprobe -i ipv6` as `root`. It is impossible to unload the IPv6 module after it has been loaded.

Because of the autoconfiguration concept of IPv6, the network card is assigned an address in the *link-local* network. Normally, no routing table management takes place on a workstation. The network routers can be queried by the workstation, using the *router advertisement protocol*, for what prefix and gateways should be implemented. The radvd program can be used to set up an IPv6 router. This program informs the workstations which prefix to use for the IPv6 addresses and which routers. Alternatively, use FRR (see https://frrouting.org/ ↗) for automatic configuration of both addresses and routing.

For information about how to set up multiple types of tunnels using the `/etc/sysconfig/network` files, see the man page of `ifcfg-tunnel` (**`man ifcfg-tunnel`**).

23.2.5 More information

The above overview does not cover the topic of IPv6 comprehensively. For a more in-depth look at the newer protocol, refer to the following online documentation and books:

https://pulse.internetsociety.org

The starting point for everything about IPv6.

http://www.ipv6day.org

All information needed to start your own IPv6 network.

http://www.ipv6-to-standard.org/

The list of IPv6-enabled products.

https://www.bieringer.de/linux/IPv6/

The Linux IPv6-HOWTO and many links related to the topic.

RFC 2460

The fundamental RFC about IPv6, see https://www.rfc-editor.org/rfc/rfc2460.

IPv6 essentials

A book describing all the important aspects of the topic is *IPv6 Essentials* by Silvia Hagen (ISBN 0-596-00125-8).

23.3 Name resolution

DNS assists in assigning an IP address to one or more names and assigning a name to an IP address. In Linux, this conversion is normally carried out by a special type of software known as bind. The machine that takes care of this conversion is called a *name server*. The names make up a hierarchical system in which each name component is separated by a period. The name hierarchy is, however, independent of the IP address hierarchy described above.

Consider a complete name, such as `jupiter.example.com`, written in the format `hostname.domain`. A full name, called a *fully qualified domain name* (FQDN), consists of a host name and a domain name (`example.com`). The latter also includes the *top level domain* or TLD (`com`).

TLD assignment has become confusing for historical reasons. Traditionally, three-letter domain names are used in the USA. In the rest of the world, the two-letter ISO national codes are the standard. Additionally, longer TLDs were introduced in 2000 that represent certain spheres of activity (for example, `.info`, `.name`, `.museum`).

In the early days of the Internet (before 1990), the file `/etc/hosts` was used to store the names of all the machines represented over the Internet. This quickly proved to be impractical in the face of the rapidly growing number of computers connected to the Internet. For this reason, a decentralized database was developed to store the host names in a widely distributed manner. This database, similar to the name server, does not have the data pertaining to all hosts in the Internet available, but can dispatch requests to other name servers.

The top of the hierarchy is occupied by *root name servers*. These root name servers manage the top level domains and are run by the Network Information Center (NIC). Each root name server knows about the name servers responsible for a given top level domain. Information about top level domain NICs is available at https://www.internic.net.

DNS can do more than resolve host names. The name server also knows which host is receiving e-mails for an entire domain—the *mail exchanger (MX)*.

For your machine to resolve an IP address, it must know about at least one name server and its IP address. Easily specify such a name server using YaST.

The protocol `whois` is closely related to DNS. With this program, quickly find out who accounts for a given domain.

Note: MDNS and .local domain names

The `.local` top level domain is treated as link-local domain by the resolver. DNS requests are sent as multicast DNS requests instead of normal DNS requests. If you already use the `.local` domain in your name server configuration, you must switch this option off in `/etc/host.conf`. For more information, see the `host.conf` man page.

To switch off MDNS during installation, use `nomdns=1` as a boot parameter.

For more information on multicast DNS, see http://www.multicastdns.org.

23.4 Configuring a network connection with YaST

There are many supported networking types on Linux. Most of them use different device names and the configuration files are spread over several locations in the file system. For a detailed overview of the aspects of manual network configuration, see *Section 23.6, “Configuring a network connection manually”*.

On SUSE Linux Enterprise Desktop, where NetworkManager is active by default, all network cards are configured. If NetworkManager is not active, only the first interface with link up (with a network cable connected) is automatically configured. Additional hardware can be configured any time on the installed system. The following sections describe the network configuration for all types of network connections supported by SUSE Linux Enterprise Desktop.

23.4.1 Configuring the network card with YaST

To configure your Ethernet or Wi-Fi/Bluetooth card in YaST, select *System* › *Network Settings*. After starting the module, YaST displays the *Network Settings* dialog with four tabs: *Global Options*, *Overview*, *Hostname/DNS* and *Routing*.

The *Global Options* tab allows you to set general networking options such as the network setup method, IPv6, and general DHCP options. For more information, see *Section 23.4.1.1, “Configuring global networking options”*.

The *Overview* tab contains information about installed network interfaces and configurations. Any properly detected network card is listed with its name. You can manually configure new cards, remove or change their configuration in this dialog. To manually configure a card that was not automatically detected, see *Section 23.4.1.3, “Configuring an undetected network card”*. To change the configuration of an already configured card, see *Section 23.4.1.2, “Changing the configuration of a network card”*.

The *Hostname/DNS* tab allows to set the host name of the machine and name the servers to be used. For more information, see *Section 23.4.1.4, “Configuring host name and DNS”*.

The *Routing* tab is used for the configuration of routing. See *Section 23.4.1.5, “Configuring routing”* for more information.

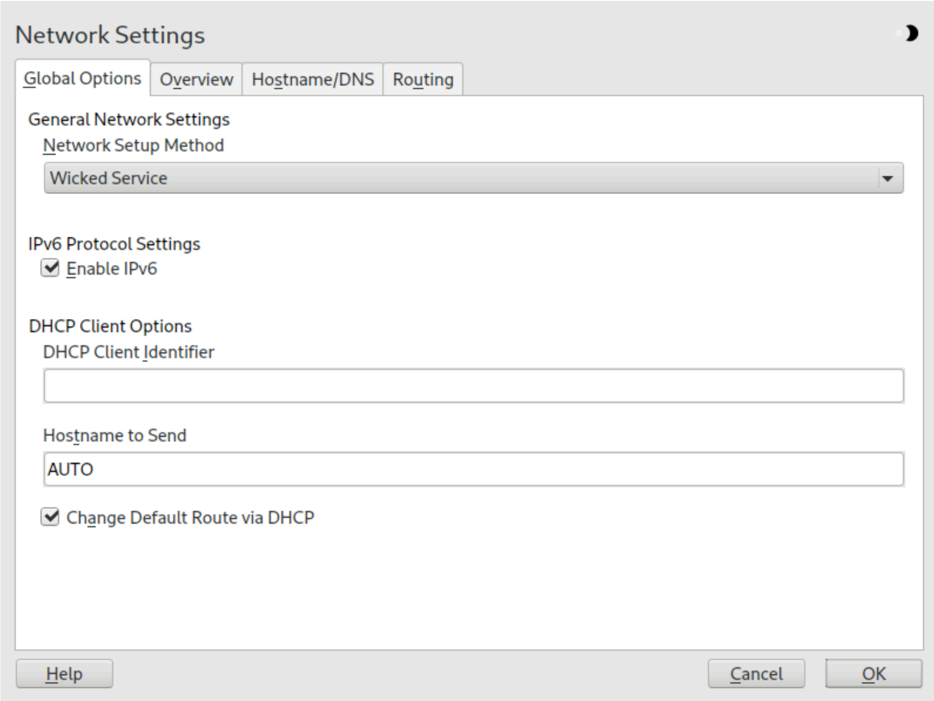


FIGURE 23.3: CONFIGURING NETWORK SETTINGS

23.4.1.1 Configuring global networking options

The *Global Options* tab of the YaST *Network Settings* module allows you to set important global networking options, such as the use of NetworkManager, IPv6 and DHCP client options. These settings are applicable for all network interfaces.

In the *Network Setup Method* choose the way network connections are managed. If you want a NetworkManager desktop applet to manage connections for all interfaces, choose *NetworkManager Service*. NetworkManager is well suited for switching between multiple wired and wireless networks. If you do not run a desktop environment, or if your computer is a Xen server, virtual system, or provides network services such as DHCP or DNS in your network, use the *Wicked Service* method. If NetworkManager is used, **`nm-applet`** should be used to configure network options and the *Overview*, *Hostname/DNS* and *Routing* tabs of the *Network Settings* module are disabled. For more information on NetworkManager, see *Chapter 31, Using NetworkManager*.

In the *IPv6 Protocol Settings* choose whether to use the IPv6 protocol. It is possible to use IPv6 together with IPv4. By default, IPv6 is enabled. However, in networks not using IPv6 protocol, response times can be faster with IPv6 protocol disabled. To disable IPv6, deactivate *Enable IPv6*. If IPv6 is disabled, the kernel no longer loads the IPv6 module automatically. This setting will be applied after reboot.

In the *DHCP Client Options* configure options for the DHCP client. The *DHCP Client Identifier* must be different for each DHCP client on a single network. If left empty, it defaults to the hardware address of the network interface. However, if you are running several virtual machines using the same network interface and, therefore, the same hardware address, specify a unique free-form identifier here.

The *Hostname to Send* specifies a string used for the host name option field when the DHCP client sends messages to DHCP server. Some DHCP servers update name server zones (forward and reverse records) according to this host name (Dynamic DNS). Also, some DHCP servers require the *Hostname to Send* option field to contain a specific string in the DHCP messages from clients. Leave `AUTO` to send the current host name (that is the one defined in `/etc/hostname`). Make the option field empty for not sending any host name.

If you do not want to change the default route according to the information from DHCP, deactivate *Change Default Route via DHCP*.

23.4.1.2 Changing the configuration of a network card

To change the configuration of a network card, select a card from the list of the detected cards in *Network Settings* › *Overview* in YaST and click *Edit*. The *Network Card Setup* dialog appears in which to adjust the card configuration using the *General*, *Address* and *Hardware* tabs.

23.4.1.2.1 Configuring IP addresses

You can set the IP address of the network card or the way its IP address is determined in the *Address* tab of the *Network Card Setup* dialog. Both IPv4 and IPv6 addresses are supported. The network card can have *No IP Address* (which is useful for bonding devices), a *Statically Assigned IP Address* (IPv4 or IPv6) or a *Dynamic Address* assigned via *DHCP* or *Zeroconf* or both.

If using *Dynamic Address*, select whether to use *DHCP Version 4 Only* (for IPv4), *DHCP Version 6 Only* (for IPv6) or *DHCP Both Version 4 and 6*.

If possible, the first network card with link that is available during the installation is automatically configured to use automatic address setup via DHCP. On SUSE Linux Enterprise Desktop, where NetworkManager is active by default, all network cards are configured.

DHCP should also be used if you are using a DSL line but with no static IP assigned by the ISP (Internet Service Provider). If you decide to use DHCP, configure the details in *DHCP Client Options* in the *Global Options* tab of the *Network Settings* dialog of the YaST network card configuration module. If you have a virtual host setup where different hosts communicate through the same interface, an *DHCP Client Identifier* is necessary to distinguish them.

DHCP is a good choice for client configuration but it is not ideal for server configuration. To set a static IP address, proceed as follows:

1. Select a card from the list of detected cards in the *Overview* tab of the YaST network card configuration module and click *Edit*.
2. In the *Address* tab, choose *Statically Assigned IP Address*.
3. Enter the *IP Address*. Both IPv4 and IPv6 addresses can be used. Enter the network mask in *Subnet Mask*. If the IPv6 address is used, use *Subnet Mask* for prefix length in format `/64`. Optionally, you can enter a fully qualified *Hostname* for this address, which will be written to the `/etc/hosts` configuration file.
4. Click *Next*.
5. To activate the configuration, click *OK*.

Note: Interface activation and link detection

During activation of a network interface, **`wicked`** checks for a carrier and only applies the IP configuration when a link has been detected. If you need to apply the configuration regardless of the link status (for example, when you want to test a service listening to a certain address), you can skip link detection by adding the variable `LINK_REQUIRED=no` to the configuration file of the interface in `/etc/sysconfig/network/ifcfg`.

Additionally, you can use the variable `LINK_READY_WAIT=5` to specify the timeout for waiting for a link in seconds.

For more information about the `ifcfg-*` configuration files, refer to *Section 23.6.2.5, "/etc/sysconfig/network/ifcfg-*"* and **`man 5 ifcfg`**.

If you use the static address, the name servers and default gateway are not configured automatically. To configure name servers, proceed as described in *Section 23.4.1.4, "Configuring host name and DNS"*. To configure a gateway, proceed as described in *Section 23.4.1.5, "Configuring routing"*.

23.4.1.2.2 Configuring multiple addresses

A single network device can have multiple IP addresses called aliases or labels.

> **Note: Aliases are a compatibility feature**
>
> Aliases or labels work with IPv4 only. Using `iproute2` network interfaces makes it possible to have one or more addresses.

To set additional addresses for your network card using YaST, proceed as follows:

1. Select a card from the list of detected cards in the *Overview* tab of the YaST *Network Settings* dialog and click *Edit*.
2. In the *Address* › *Additional Addresses* tab, click *Add*.
3. Enter *IPv4 Address Label*, *IP Address*, and *Netmask*. Note that IP aliases must be added with the `/32` netmask. Do not include the interface name in the alias name.
4. To activate the configuration, confirm the settings.

23.4.1.2.3 Changing the device name and udev rules

It is possible to change the device name of the network card when it is used. It is also possible to determine whether the network card should be identified by udev via its hardware (MAC) address or via the bus ID. The latter option is preferable in large servers to simplify hotplugging of cards. To set these options with YaST, proceed as follows:

1. Select a card from the list of detected cards in the *Overview* tab of the YaST *Network Settings* dialog and click *Edit*.
2. Go to the *General* tab. The current device name is shown in *Udev Rules*. Click *Change*.
3. Select whether udev should identify the card by its *MAC Address* or *Bus ID*. The current MAC address and bus ID of the card are shown in the dialog.
4. To change the device name, check the *Change Device Name* option and edit the name.
5. To activate the configuration, confirm the settings.

23.4.1.2.4 Changing network card kernel driver

For some network cards, several kernel drivers may be available. If the card is already configured, YaST allows you to select a kernel driver to be used from a list of available suitable drivers. It is also possible to specify options for the kernel driver. To set these options with YaST, proceed as follows:

1. Select a card from the list of detected cards in the *Overview* tab of the YaST Network Settings module and click *Edit*.

2. Go to the *Hardware* tab.

3. Select the kernel driver to be used in *Module Name*. Enter any options for the selected driver in *Options* in the form **`=`** =`VALUE`. If more options are used, they should be space-separated.

4. To activate the configuration, confirm the settings.

23.4.1.2.5 Activating the network device

If you use the method with **`wicked`**, you can configure your device to either start during boot, on cable connection, on card detection, manually, or never. To change device start-up, proceed as follows:

1. In YaST select a card from the list of detected cards in *System* › *Network Settings* and click *Edit*.

2. In the *General* tab, select the desired entry from *Device Activation*.
   Choose *At Boot Time* to start the device during the system boot. With *On Cable Connection*, the interface is watched for any existing physical connection. With *On Hotplug*, the interface is set when available. It is similar to the *At Boot Time* option, and only differs in that no error occurs if the interface is not present at boot time. Choose *Manually* to control the interface manually with **`ifup`**. Choose *Never* to not start the device. The *On NFSroot* is similar to *At Boot Time*, but the interface does not shut down with the **`systemctl stop network`** command; the `network` service also cares about the `wicked` service if **`wicked`** is active. Use this if you use an NFS or iSCSI root file system.

3. To activate the configuration, confirm the settings.

Tip: NFS as a root file system

On (diskless) systems where the root partition is mounted via network as an NFS share, you need to be careful when configuring the network device with which the NFS share is accessible.

When shutting down or rebooting the system, the default processing order is to turn off network connections, then unmount the root partition. With NFS root, this order causes problems as the root partition cannot be cleanly unmounted as the network connection to the NFS share is already not activated. To prevent the system from deactivating the relevant network device, open the network device configuration tab as described in *Section 23.4.1.2.5, "Activating the network device"* and choose *On NFSroot* in the *Device Activation* pane.

23.4.1.2.6 Setting up maximum transfer unit size

You can set a maximum transmission unit (MTU) for the interface. MTU refers to the largest allowed packet size in bytes. A higher MTU brings higher bandwidth efficiency. However, large packets can block up a slow interface for some time, increasing the lag for further packets.

1. In YaST select a card from the list of detected cards in *System* › *Network Settings* and click *Edit*.
2. In the *General* tab, select the desired entry from the *Set MTU* list.
3. To activate the configuration, confirm the settings.

23.4.1.2.7 PCIe multifunction devices

Multifunction devices that support LAN, iSCSI, and FCoE are supported. The YaST FCoE client (**`yast2 fcoe-client`**) shows the private flags in additional columns to allow the user to select the device meant for FCoE. The YaST network module (**`yast2 lan`**) excludes "storage only devices" for network configuration.

23.4.1.2.8 Infiniband configuration for IP-over-InfiniBand (IPoIB)

1. In YaST select the InfiniBand device in *System* › *Network Settings* and click *Edit*.

2. In the *General* tab, select one of the *IP-over-InfiniBand* (IPoIB) modes: *connected* (default) or *datagram*.

3. To activate the configuration, confirm the settings.

For more information about InfiniBand, see `/usr/src/linux/Documentation/infiniband/ipoib.txt`.

23.4.1.2.9 Configuring the firewall

Without having to perform the detailed firewall setup as described in *Book "Security and Hardening Guide", Chapter 23 "Masquerading and firewalls", Section 23.4 "`firewalld`"*, you can determine the basic firewall configuration for your device as part of the device setup. Proceed as follows:

1. Open the YaST *System* › *Network Settings* module. In the *Overview* tab, select a card from the list of detected cards and click *Edit*.

2. Enter the *General* tab of the *Network Settings* dialog.

3. Determine the *Firewall Zone* to which your interface should be assigned. The following options are available:

   Firewall disabled
   : This option is available only if the firewall is disabled and the firewall does not run. Only use this option if your machine is part of a greater network that is protected by an outer firewall.

   Automatically assign zone
   : This option is available only if the firewall is enabled. The firewall is running and the interface is automatically assigned to a firewall zone. The zone which contains the keyword `any` or the external zone will be used for such an interface.

   Internal zone (unprotected)
   : The firewall is running, but does not enforce any rules to protect this interface. Use this option if your machine is part of a greater network that is protected by an outer firewall. It is also useful for the interfaces connected to the internal network, when the machine has more network interfaces.

Demilitarized zone

A demilitarized zone is an additional line of defense in front of an internal network and the (hostile) Internet. Hosts assigned to this zone can be reached from the internal network and from the Internet, but cannot access the internal network.

External zone

The firewall is running on this interface and fully protects it against other—presumably hostile—network traffic. This is the default option.

4. To activate the configuration, confirm the settings.

23.4.1.3 Configuring an undetected network card

If a network card is not detected correctly, the card is not included in the list of detected cards. If you are sure that your system includes a driver for your card, you can configure it manually. You can also configure special network device types, such as bridge, bond, TUN or TAP. To configure an undetected network card (or a special device) proceed as follows:

1. In the *System* › *Network Settings* › *Overview* dialog in YaST click *Add*.
2. In the *Hardware* dialog, set the *Device Type* of the interface from the available options and *Configuration Name*. If the network card is a USB device, activate the respective check box and exit this dialog with *Next*. Otherwise, you can define the kernel *Module Name* to be used for the card and its *Options*, if necessary.
   In *Ethtool Options*, you can set **`ethtool`** options used by **`ifup`** for the interface. For information about available options, see the **`ethtool`** manual page.
   If the option string starts with a `-` (for example, `-K INTERFACE_NAME rx on`), the second word in the string is replaced with the current interface name. Otherwise (for example, `autoneg off speed 10`) **`ifup`** adds `-s INTERFACE_NAME` to the beginning.
3. Click *Next*.
4. Configure any needed options, such as the IP address, device activation or firewall zone for the interface in the *General*, *Address*, and *Hardware* tabs. For more information about the configuration options, see *Section 23.4.1.2, "Changing the configuration of a network card"*.
5. If you selected *Wireless* as the device type of the interface, configure the wireless connection in the next dialog.

6. To activate the new network configuration, confirm the settings.

23.4.1.4 Configuring host name and DNS

If you did not change the network configuration during installation and the Ethernet card was already available, a host name was automatically generated for your computer and DHCP was activated. The same applies to the name service information your host needs to integrate into a network environment. If DHCP is used for network address setup, the list of domain name servers is automatically filled with the appropriate data. If a static setup is preferred, set these values manually.

To change the name of your computer and adjust the name server search list, proceed as follows:

1. Go to the *Network Settings* › *Hostname/DNS* tab in the *System* module in YaST.
2. Enter the *Hostname*. Note that the host name is global and applies to all network interfaces. If you are using DHCP to get an IP address, the host name of your computer will be automatically set by the DHCP server. You should disable this behavior if you connect to different networks, because they may assign different host names and changing the host name at runtime may confuse the graphical desktop. To disable using DHCP to get an IP address deactivate *Change Hostname via DHCP*.
3. In *Modify DNS Configuration*, select the way the DNS configuration (name servers, search list, the content of the `/run/netconfig/resolv.conf` file) is modified.
   If the *Use Default Policy* option is selected, the configuration is handled by the **`netconfig`** script which merges the data defined statically (with YaST or in the configuration files) with data obtained dynamically (from the DHCP client or NetworkManager). This default policy is usually sufficient.
   If the *Only Manually* option is selected, **`netconfig`** is not allowed to modify the `/run/netconfig/resolv.conf` file. However, this file can be edited manually.
   If the *Custom Policy* option is selected, a *Custom Policy Rule* string defining the merge policy should be specified. The string consists of a comma-separated list of interface names to be considered a valid source of settings. Except for complete interface names, basic wild cards to match multiple interfaces are allowed, as well. For example, `eth* ppp?` will first target all eth and then all ppp0-ppp9 interfaces. There are two special policy values that indicate how to apply the static settings defined in the `/etc/sysconfig/network/config` file:

   `STATIC`
   : The static settings need to be merged together with the dynamic settings.

`STATIC_FALLBACK`

The static settings are used only when no dynamic configuration is available.

For more information, see the man page of **`netconfig`**(8) (**`man 8 netconfig`**).

4. Enter the *Name Servers* and fill in the *Domain Search* list. Name servers must be specified by IP addresses, such as 192.168.1.116, not by host names. Names specified in the *Domain Search* tab are domain names used for resolving host names without a specified domain. If more than one *Domain Search* is used, separate domains with commas or white space.

5. To activate the configuration, confirm the settings.

It is also possible to edit the host name using YaST from the command line. The changes made by YaST take effect immediately (which is not the case when editing the `/etc/hostname` file manually). To change the host name, use the following command:

```
# yast dns edit hostname=HOSTNAME
```

To change the name servers, use the following commands:

```
# yast dns edit nameserver1=192.168.1.116
# yast dns edit nameserver2=192.168.1.117
# yast dns edit nameserver3=192.168.1.118
```

23.4.1.5 Configuring routing

To make your machine communicate with other machines and other networks, routing information must be given to make network traffic take the correct path. If DHCP is used, this information is automatically provided. If a static setup is used, this data must be added manually.

1. In YaST go to *Network Settings* › *Routing*.

2. Enter the IP address of the *Default Gateway* (IPv4 and IPv6 if necessary). The default gateway matches every possible destination, but if a routing table entry exists that matches the required address, this will be used instead of the default route via the Default Gateway.

3. More entries can be entered in the *Routing Table*. Enter the *Destination* network IP address, *Gateway* IP address and the *Netmask*. Select the *Device* through which the traffic to the defined network will be routed (the minus sign stands for any device). To omit any of these values, use the minus sign `-`. To enter a default gateway into the table, use `default` in the *Destination* field.

> **Note: Route prioritization**
>
> If more default routes are used, it is possible to specify the metric option to determine which route has a higher priority. To specify the metric option, enter `- metric NUMBER` in *Options*. The lowest possible metric is 0. The route with the lowest metric has the highest priority and is used as default. If the network device is disconnected, its route will be removed and the next one will be used.

4. If the system is a router, enable *IPv4 Forwarding* and *IPv6 Forwarding* in the *Network Settings* as needed.
5. To activate the configuration, confirm the settings.

23.5 NetworkManager

NetworkManager is the ideal solution for laptops and other portable computers. With NetworkManager, you do not need to worry about configuring network interfaces and switching between networks when you are moving.

> **Important: Support coverage**
>
> NetworkManager is only supported by SUSE for desktop workloads with SLED or the Workstation extension. All server certifications are done with **`wicked`** as the network configuration tool, and using NetworkManager may invalidate them. NetworkManager is not supported by SUSE for server workloads.

23.5.1 NetworkManager and `wicked`

However, NetworkManager is not a suitable solution for all cases, so you can still choose between the **`wicked`** controlled method for managing network connections and NetworkManager. If you want to manage your network connection with NetworkManager, enable NetworkManager in the YaST Network Settings module as described in *Section 31.2, "Enabling or disabling NetworkManager"* and configure your network connections with NetworkManager. For a list of use cases and a detailed description of how to configure and use NetworkManager, refer to *Chapter 31, Using NetworkManager*.

Some differences between wicked and NetworkManager:

`root` privileges

: If you use NetworkManager for network setup, you can easily switch, stop or start your network connection at any time from within your desktop environment using an applet. NetworkManager also makes it possible to change and configure wireless card connections without requiring `root` privileges. For this reason, NetworkManager is the ideal solution for a mobile workstation.

  `wicked` also provides some ways to switch, stop or start the connection with or without user intervention, like user-managed devices. However, this always requires `root` privileges to change or configure a network device. This is often a problem for mobile computing, where it is not possible to preconfigure all the connection possibilities.

Types of network connections

: Both **`wicked`** and NetworkManager can handle network connections with a wireless network (with WEP, WPA-PSK, and WPA-Enterprise access) and wired networks using DHCP and static configuration. They also support connection through dial-up and VPN. With NetworkManager you can also connect a mobile broadband (3G) modem or set up a DSL connection, which is not possible with the traditional configuration.

  NetworkManager tries to keep your computer connected at all times using the best connection available. If the network cable is accidentally disconnected, it tries to reconnect. It can find the network with the best signal strength from the list of your wireless connections and automatically use it to connect. To get the same functionality with **`wicked`**, more configuration effort is required.

23.5.2 NetworkManager functionality and configuration files

The individual network connection settings created with NetworkManager are stored in configuration profiles. The *system* connections configured with either NetworkManager or YaST are saved in `/etc/NetworkManager/system-connections/*` or in `/etc/sysconfig/network/ifcfg-*`. For GNOME, all user-defined connections are stored in GConf.

In case no profile is configured, NetworkManager automatically creates one and names it `Auto $INTERFACE-NAME`. That is made in an attempt to work without any configuration for as many cases as (securely) possible. If the automatically created profiles do not suit your needs, use the network connection configuration dialogs provided by GNOME to modify them as desired. For more information, see *Section 31.3, “Configuring network connections”*.

23.5.3 Controlling and locking down NetworkManager features

On centrally administered machines, certain NetworkManager features can be controlled or disabled with Polkit, for example if a user is allowed to modify administrator defined connections or if a user is allowed to define their own network configurations. To view or change the respective NetworkManager policies, start the graphical *Authorizations* tool for Polkit. In the tree on the left side, find them below the *network-manager-settings* entry. For an introduction to Polkit and details on how to use it, refer to *Book "Security and Hardening Guide", Chapter 18 "The Polkit authentication framework"*.

23.6 Configuring a network connection manually

Manual configuration of the network software should be the last alternative. Using YaST is recommended. However, this background information about the network configuration can also assist your work with YaST.

23.6.1 The `wicked` network configuration

The tool and library called **`wicked`** provides a new framework for network configuration.

One of the challenges with traditional network interface management is that different layers of network management get jumbled together into one single script, or at most two different scripts. These scripts interact with each other in a way that is not well defined. This leads to unpredictable issues, obscure constraints and conventions, etc. Several layers of special hacks for a variety of different scenarios increase the maintenance burden. Address configuration protocols are being used that are implemented via daemons like dhcpcd, which interact rather poorly with the rest of the infrastructure. Funky interface naming schemes that require heavy udev support are introduced to achieve persistent identification of interfaces.

The idea of wicked is to decompose the problem in several ways. None of them is entirely novel, but trying to put ideas from different projects together is hopefully going to create a better solution overall.

One approach is to use a client/server model. This allows wicked to define standardized facilities for things like address configuration that are well integrated with the overall framework. For example, using a specific address configuration, the administrator may request that an interface should be configured via DHCP or IPv4 zeroconf. In this case, the address configuration service simply obtains the lease from its server and passes it on to the wicked server process that installs the requested addresses and routes.

The other approach to decomposing the problem is to enforce the layering aspect. For any type of network interface, it is possible to define a dbus service that configures the network interface's device layer—a VLAN, a bridge, a bonding, or a paravirtualized device. Common functionality, such as address configuration, is implemented by joint services that are layered on top of these device specific services without having to implement them specifically.

The wicked framework implements these two aspects by using a variety of dbus services, which get attached to a network interface depending on its type. Here is a rough overview of the current object hierarchy in wicked.

Each network interface is represented via a child object of `/org/opensuse/Network/Interfaces`. The name of the child object is given by its ifindex. For example, the loopback interface, which usually gets ifindex 1, is `/org/opensuse/Network/Interfaces/1`, the first Ethernet interface registered is `/org/opensuse/Network/Interfaces/2`.

Each network interface has a "class" associated with it, which is used to select the dbus interfaces it supports. By default, each network interface is of class `netif`, and `wickedd` will automatically attach all interfaces compatible with this class. In the current implementation, this includes the following interfaces:

org.opensuse.Network.Interface

Generic network interface functions, such as taking the link up or down, assigning an MTU, etc.

org.opensuse.Network.Addrconf.ipv4.dhcp,
org.opensuse.Network.Addrconf.ipv6.dhcp,
org.opensuse.Network.Addrconf.ipv4.auto

Address configuration services for DHCP, IPv4 zeroconf, etc.

Beyond this, network interfaces may require or offer special configuration mechanisms. For an Ethernet device, for example, you should be able to control the link speed, offloading of checksumming, etc. To achieve this, Ethernet devices have a class of their own, called `netif-ethernet`, which is a subclass of `netif`. As a consequence, the dbus interfaces assigned to an Ethernet interface include all the services listed above, plus the `org.opensuse.Network.Ethernet` service available only to objects belonging to the `netif-ethernet` class.

Similarly, there exist classes for interface types like bridges, VLANs, bonds, or infinibands.

How do you interact with an interface like VLAN (which is really a virtual network interface that sits on top of an Ethernet device) that needs to be created first? For this, wicked defines factory interfaces, such as `org.opensuse.Network.VLAN.Factory`. Such a factory interface offers a single function that lets you create an interface of the requested type. These factory interfaces are attached to the `/org/opensuse/Network/Interfaces` list node.

23.6.1.1 `wicked` architecture and features

The `wicked` service comprises several parts as depicted in *Figure 23.4, “`wicked` architecture”*.

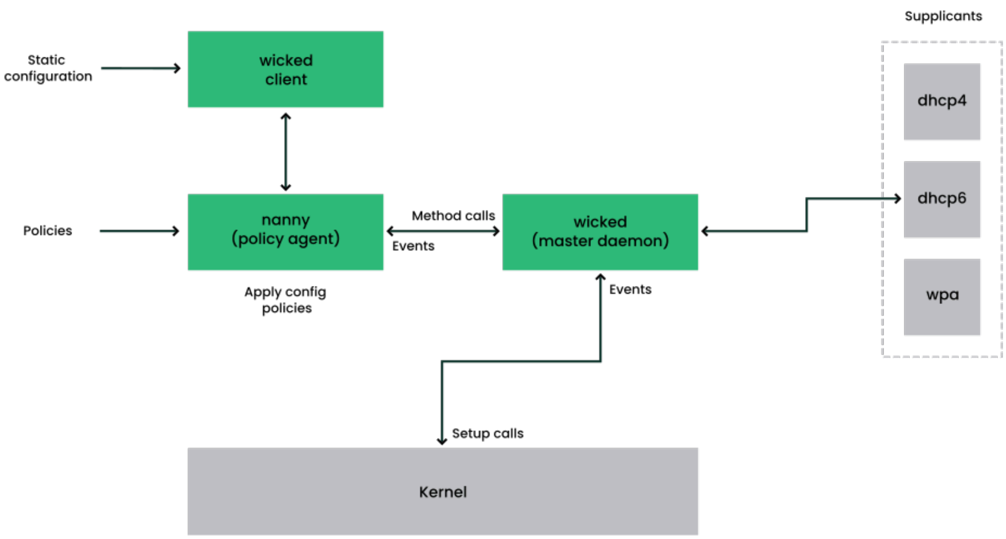


FIGURE 23.4: `wicked` ARCHITECTURE

`wicked` currently supports the following:

- Configuration file back-ends to parse SUSE style `/etc/sysconfig/network` files.
- An internal configuration back-end to represent network interface configuration in XML.
- Bring up and shutdown of “normal” network interfaces such as Ethernet or InfiniBand, VLAN, bridge, bonds, tun, tap, dummy, macvlan, macvtap, hsi, qeth, iucv, and wireless (currently limited to one wpa-psk/eap network) devices.
- A built-in DHCPv4 client and a built-in DHCPv6 client.

- The nanny daemon (enabled by default) helps to automatically bring up configured interfaces when the device is available (interface hotplugging) and set up the IP configuration when a link (carrier) is detected. See *Section 23.6.1.3, "Nanny"* for more information.
- `wicked` was implemented as a group of DBus services that are integrated with systemd. So the usual **`systemctl`** commands will apply to `wicked`.

23.6.1.2 Using wicked

On SUSE Linux Enterprise, `wicked` runs by default. If you want to check what is currently enabled and whether it is running, call:

```
systemctl status network
```

If `wicked` is enabled, you will see something along these lines:

```
wicked.service - wicked managed network interfaces
    Loaded: loaded (/usr/lib/systemd/system/wicked.service; enabled)
    ...
```

In case something different is running (for example, NetworkManager) and you want to switch to `wicked`, first stop what is running and then enable `wicked`:

```
systemctl is-active network && \
systemctl stop      network
systemctl enable --force wicked
```

This enables the wicked services, creates the `network.service` to `wicked.service` alias link, and starts the network at the next boot.

Starting the server process:

```
systemctl start wickedd
```

This starts **`wickedd`** (the main server) and associated supplicants:

```
/usr/lib/wicked/bin/wickedd-auto4 --systemd --foreground
/usr/lib/wicked/bin/wickedd-dhcp4 --systemd --foreground
/usr/lib/wicked/bin/wickedd-dhcp6 --systemd --foreground
/usr/sbin/wickedd --systemd --foreground
/usr/sbin/wickedd-nanny --systemd --foreground
```

Then bringing up the network:

```
systemctl start wicked
```

Alternatively use the `network.service` alias:

```
systemctl start network
```

These commands are using the default or system configuration sources as defined in `/etc/wicked/client.xml`.

To enable debugging, set `WICKED_DEBUG` in `/etc/sysconfig/network/config`, for example:

```
WICKED_DEBUG="all"
```

Or, to omit some:

```
WICKED_DEBUG="all,-dbus,-objectmodel,-xpath,-xml"
```

Use the client utility to display interface information for all interfaces or the interface specified with *`IFNAME`*:

```
wicked show all
wicked show IFNAME
```

In XML output:

```
wicked show-xml all
wicked show-xml IFNAME
```

Bringing up one interface:

```
wicked ifup eth0
wicked ifup wlan0
...
```

Because there is no configuration source specified, the wicked client checks its default sources of configuration defined in `/etc/wicked/client.xml`:

1. `firmware:` iSCSI Boot Firmware Table (iBFT)
2. `compat:` `ifcfg` files—implemented for compatibility

Whatever `wicked` gets from those sources for a given interface is applied. The intended order of importance is `firmware`, then `compat`—this may be changed in the future.

For more information, see the **`wicked`** man page.

23.6.1.3 Nanny

Nanny is an event and policy driven daemon that is responsible for asynchronous or unsolicited scenarios such as hotplugging devices. Thus the nanny daemon helps with starting or restarting delayed or temporarily gone devices. Nanny monitors device and link changes, and integrates new devices defined by the current policy set. Nanny continues to set up even if **`ifup`** already exited because of specified timeout constraints.

By default, the nanny daemon is active on the system. It is enabled in the `/etc/wicked/common.xml` configuration file:

```
<config>
  ...
  <use-nanny>true</use-nanny>
</config>
```

This setting causes ifup and ifreload to apply a policy with the effective configuration to the nanny daemon; then, nanny configures `wickedd` and thus ensures hotplug support. It waits in the background for events or changes (such as new devices or carrier on).

23.6.1.4 Bringing up multiple interfaces

For bonds and bridges, it may make sense to define the entire device topology in one file (ifcfg-bondX), and bring it up in one go. wicked then can bring up the whole configuration if you specify the top level interface names (of the bridge or bond):

```
wicked ifup br0
```

This command automatically sets up the bridge and its dependencies in the appropriate order without the need to list the dependencies (ports, etc.) separately.

To bring up multiple interfaces in one command:

```
wicked ifup bond0 br0 br1 br2
```

Or also all interfaces:

```
wicked ifup all
```

23.6.1.5 Using tunnels with Wicked

When you need to use tunnels with Wicked, the `TUNNEL_DEVICE` is used for this. It permits to specify an optional device name to bind the tunnel to the device. The tunneled packets will only be routed via this device.

For more information, refer to **`man 5 ifcfg-tunnel`**.

23.6.1.6 Handling incremental changes

With **`wicked`**, there is no need to actually take down an interface to reconfigure it (unless it is required by the kernel). For example, to add another IP address or route to a statically configured network interface, add the IP address to the interface definition, and do another “ifup” operation. The server will try hard to update only those settings that have changed. This applies to link-level options such as the device MTU or the MAC address, and network-level settings, such as addresses, routes, or even the address configuration mode (for example, when moving from a static configuration to DHCP).

Things get tricky of course with virtual interfaces combining several real devices such as bridges or bonds. For bonded devices, it is not possible to change certain parameters while the device is up. Doing that will result in an error.

However, what should still work, is the act of adding or removing the child devices of a bond or bridge, or choosing a bond's primary interface.

23.6.1.7 Wicked extensions: address configuration

`wicked` is designed to be extensible with shell scripts. These extensions can be defined in the `config.xml` file.

Currently, several classes of extensions are supported:

- link configuration: these are scripts responsible for setting up a device's link layer according to the configuration provided by the client, and for tearing it down again.
- address configuration: these are scripts responsible for managing a device's address configuration. Usually address configuration and DHCP are managed by **`wicked`** itself, but can be implemented by means of extensions.
- firewall extension: these scripts can apply firewall rules.

Typically, extensions have a start and a stop command, an optional “pid file”, and a set of environment variables that get passed to the script.

To illustrate how this is supposed to work, look at a firewall extension defined in `etc/server.xml`:

```
<dbus-service interface="org.opensuse.Network.Firewall">
 <action name="firewallUp"   command="/etc/wicked/extensions/firewall up"/>
 <action name="firewallDown" command="/etc/wicked/extensions/firewall down"/>

 <!-- default environment for all calls to this extension script -->
 <putenv name="WICKED_OBJECT_PATH" value="$object-path"/>
 <putenv name="WICKED_INTERFACE_NAME" value="$property:name"/>
 <putenv name="WICKED_INTERFACE_INDEX" value="$property:index"/>
</dbus-service>
```

The extension is attached to the `<dbus-service>` tag and defines commands to execute for the actions of this interface. Further, the declaration can define and initialize environment variables passed to the actions.

23.6.1.8 Wicked extensions: configuration files

You can extend the handling of configuration files with scripts as well. For example, DNS updates from leases are ultimately handled by the `extensions/resolver` script, with behavior configured in `server.xml`:

```
<system-updater name="resolver">
 <action name="backup" command="/etc/wicked/extensions/resolver backup"/>
 <action name="restore" command="/etc/wicked/extensions/resolver restore"/>
 <action name="install" command="/etc/wicked/extensions/resolver install"/>
 <action name="remove" command="/etc/wicked/extensions/resolver remove"/>
</system-updater>
```

When an update arrives in `wickedd`, the system updater routines parse the lease and call the appropriate commands (`backup`, `install`, etc.) in the resolver script. This in turn configures the DNS settings using **`/sbin/netconfig`**, or by manually writing `/run/netconfig/resolv.conf` as a fallback.

23.6.2 Configuration files

This section provides an overview of the network configuration files and explains their purpose and the format used.

23.6.2.1 /etc/wicked/common.xml

The /etc/wicked/common.xml file contains common definitions that should be used by all applications. It is sourced/included by the other configuration files in this directory. Although you can use this file to enable debugging across all wicked components, we recommend to use the file /etc/wicked/local.xml for this purpose. After applying maintenance updates you might lose your changes as the /etc/wicked/common.xml might be overwritten. The /etc/wicked/common.xml file includes the /etc/wicked/local.xml in the default installation, thus you typically do not need to modify the /etc/wicked/common.xml.

In case you want to disable nanny by setting the <use-nanny> to false, restart the wickedd.service and then run the following command to apply all configurations and policies:

```
> sudo wicked ifup all
```

> **Note: Configuration files**
>
> The wickedd, wicked, or nanny programs try to read /etc/wicked/common.xml if their own configuration files do not exist.

23.6.2.2 /etc/wicked/server.xml

The file /etc/wicked/server.xml is read by the wickedd server process at start-up. The file stores extensions to the /etc/wicked/common.xml. On top of that this file configures handling of a resolver and receiving information from addrconf supplicants, for example DHCP.

We recommend to add changes required to this file into a separate file /etc/wicked/server-local.xml, that gets included by /etc/wicked/server.xml. By using a separate file you avoid overwriting of your changes during maintenance updates.

23.6.2.3 /etc/wicked/client.xml

The /etc/wicked/client.xml is used by the **wicked** command. The file specifies the location of a script used when discovering devices managed by ibft and configures locations of network interface configurations.

We recommend to add changes required to this file into a separate file /etc/wicked/client-local.xml, that gets included by /etc/wicked/server.xml. By using a separate file you avoid overwriting of your changes during maintenance updates.

23.6.2.4 /etc/wicked/nanny.xml

The /etc/wicked/nanny.xml configures types of link layers. We recommend to add specific configuration into a separate file: /etc/wicked/nanny-local.xml to avoid losing the changes during maintenance updates.

23.6.2.5 /etc/sysconfig/network/ifcfg-*

These files contain the traditional configurations for network interfaces.

> **Note: wicked and the ifcfg-* files**
>
> **wicked** reads these files if you specify the compat: prefix. According to the SUSE Linux Enterprise Desktop default configuration in /etc/wicked/client.xml, **wicked** tries these files before the XML configuration files in /etc/wicked/ifconfig.
>
> The --ifconfig switch is provided mostly for testing only. If specified, default configuration sources defined in /etc/wicked/ifconfig are not applied.

The ifcfg-* files include information such as the start mode and the IP address. Possible parameters are described in the manual page of ifup. Additionally, most variables from the dhcp and wireless files can be used in the ifcfg-* files if a general setting should be used for only one interface. However, most of the /etc/sysconfig/network/config variables are global and cannot be overridden in ifcfg files. For example, NETCONFIG_* variables are global.

For configuring macvlan and macvtab interfaces, see the ifcfg-macvlan and ifcfg-macvtap man pages. For example, for a macvlan interface provide a ifcfg-macvlan0 with settings as follows:

```
STARTMODE='auto'
MACVLAN_DEVICE='eth0'
#MACVLAN_MODE='vepa'
#LLADDR=02:03:04:05:06:aa
```

For ifcfg.template, see *Section 23.6.2.6, "/etc/sysconfig/network/config, /etc/sysconfig/network/dhcp, and /etc/sysconfig/network/wireless"*.

23.6.2.6 /etc/sysconfig/network/config, /etc/sysconfig/network/dhcp, and /etc/sysconfig/network/wireless

The file `config` contains general settings for the behavior of **`ifup`**, **`ifdown`** and **`ifstatus`**. `dhcp` contains settings for DHCP and `wireless` for wireless LAN cards. The variables in all three configuration files are commented. Some variables from `/etc/sysconfig/network/config` can also be used in `ifcfg-*` files, where they are given a higher priority. The `/etc/sysconfig/network/ifcfg.template` file lists variables that can be specified in a per interface scope. However, most of the `/etc/sysconfig/network/config` variables are global and cannot be overridden in ifcfg-files. For example, `NETWORKMANAGER` or `NETCONFIG_*` variables are global.

Note: Using DHCPv6

In SUSE Linux Enterprise 11, DHCPv6 used to work even on networks where IPv6 Router Advertisements (RAs) were not configured properly. Starting with SUSE Linux Enterprise 12, DHCPv6 requires that at least one of the routers on the network sends out RAs that indicate that this network is managed by DHCPv6.

For networks where the router cannot be configured correctly, the `ifcfg` option allows the user to override this behavior by specifying `DHCLIENT6_MODE='managed'` in the `ifcfg` file. You can also activate this workaround with a boot parameter in the installation system:

```
ifcfg=eth0=dhcp6,DHCLIENT6_MODE=managed
```

23.6.2.7 /etc/sysconfig/network/routes and /etc/sysconfig/network/ifroute-*

The static routing of TCP/IP packets is determined by the `/etc/sysconfig/network/routes` and `/etc/sysconfig/network/ifroute-*` files. All the static routes required by the various system tasks can be specified in `/etc/sysconfig/network/routes`: routes to a host, routes to a host via a gateway and routes to a network. For each interface that needs individual routing, define an additional configuration file: `/etc/sysconfig/network/ifroute-*`. Replace the wild card (`*`) with the name of the interface. The entries in the routing configuration files look like this:

```
# Destination     Gateway           Netmask           Interface  Options
```

The route's destination is in the first column. This column may contain the IP address of a network or host or, in the case of *reachable* name servers, the fully qualified network or host name. The network should be written in CIDR notation (address with the associated routing prefix-length) such as 10.10.0.0/16 for IPv4 or fc00::/7 for IPv6 routes. The keyword `default` indicates that the route is the default gateway in the same address family as the gateway. For devices without a gateway use explicit 0.0.0.0/0 or ::/0 destinations.

The second column contains the default gateway or a gateway through which a host or network can be accessed.

The third column is deprecated; it used to contain the IPv4 netmask of the destination. For IPv6 routes, the default route, or when using a prefix-length (CIDR notation) in the first column, enter a dash (`-`) here.

The fourth column contains the name of the interface. If you leave it empty using a dash (`-`), it can cause unintended behavior in `/etc/sysconfig/network/routes`. For more information, see the `routes` man page.

An (optional) fifth column can be used to specify special options. For details, see the `routes` man page.

EXAMPLE 23.5: COMMON NETWORK INTERFACES AND SOME STATIC ROUTES

```
# --- IPv4 routes in CIDR prefix notation:
# Destination     [Gateway]         -                  Interface
127.0.0.0/8       -                 -                  lo
204.127.235.0/24  -                 -                  eth0
default           204.127.235.41    -                  eth0
207.68.156.51/32  207.68.145.45     -                  eth1
192.168.0.0/16    207.68.156.51     -                  eth1

# --- IPv4 routes in deprecated netmask notation"
# Destination     [Dummy/Gateway]   Netmask            Interface
#
127.0.0.0         0.0.0.0           255.255.255.0      lo
204.127.235.0     0.0.0.0           255.255.255.0      eth0
default           204.127.235.41    0.0.0.0            eth0
207.68.156.51     207.68.145.45     255.255.255.255    eth1
192.168.0.0       207.68.156.51     255.255.0.0        eth1

# --- IPv6 routes are always using CIDR notation:
# Destination     [Gateway]                -           Interface
2001:DB8:100::/64 -                        -           eth0
2001:DB8:100::/32 fe80::216:3eff:fe6d:c042 -           eth0
```

23.6.2.8 /var/run/netconfig/resolv.conf

The domain to which the host belongs is specified in /var/run/netconfig/resolv.conf (keyword search). Up to six domains with a total of 256 characters can be specified with the search option. When resolving a name that is not fully qualified, an attempt is made to generate one by attaching the individual search entries. Up to three name servers can be specified with the nameserver option, each on a line of its own. Comments are preceded by hash mark or semicolon signs (# or ;). As an example, see *Example 23.6, "/var/run/netconfig/resolv.conf"*.

However, /etc/resolv.conf should not be edited by hand. It is generated by the **netconfig** script and is a symbolic link to /run/netconfig/resolv.conf. To define static DNS configuration without using YaST, edit the appropriate variables manually in the /etc/sysconfig/network/config file:

NETCONFIG_DNS_STATIC_SEARCHLIST
: list of DNS domain names used for host name lookup

NETCONFIG_DNS_STATIC_SERVERS
: list of name server IP addresses to use for host name lookup

NETCONFIG_DNS_FORWARDER
: the name of the DNS forwarder that needs to be configured, for example bind or resolver

NETCONFIG_DNS_RESOLVER_OPTIONS
: arbitrary options that will be written to /var/run/netconfig/resolv.conf, for example:

```
debug attempts:1 timeout:10
```

For more information, see the resolv.conf man page.

NETCONFIG_DNS_RESOLVER_SORTLIST
: list of up to 10 items, for example:

```
130.155.160.0/255.255.240.0 130.155.0.0
```

For more information, see the resolv.conf man page.

To disable DNS configuration using netconfig, set NETCONFIG_DNS_POLICY=''. For more information about **netconfig**, see the netconfig(8) man page (**man 8 netconfig**).

EXAMPLE 23.6: /var/run/netconfig/resolv.conf

```
# Our domain
search example.com
#
```

```
# We use dns.example.com (192.168.1.116) as nameserver
nameserver 192.168.1.116
```

23.6.2.9 `/sbin/netconfig`

`netconfig` is a modular tool to manage additional network configuration settings. It merges statically defined settings with settings provided by autoconfiguration mechanisms as DHCP or PPP according to a predefined policy. The required changes are applied to the system by calling the netconfig modules that are responsible for modifying a configuration file and restarting a service or a similar action.

`netconfig` recognizes three main actions. The **`netconfig modify`** and **`netconfig remove`** commands are used by daemons such as DHCP or PPP to provide or remove settings to netconfig. Only the **`netconfig update`** command is available for the user:

`modify`

The **`netconfig modify`** command modifies the current interface and service specific dynamic settings and updates the network configuration. Netconfig reads settings from standard input or from a file specified with the `--lease-file` *`FILENAME`* option and internally stores them until a system reboot (or the next modify or remove action). Already existing settings for the same interface and service combination are overwritten. The interface is specified by the `-i` *`INTERFACE_NAME`* parameter. The service is specified by the `-s` *`SERVICE_NAME`* parameter.

`remove`

The **`netconfig remove`** command removes the dynamic settings provided by an editing action for the specified interface and service combination and updates the network configuration. The interface is specified by the `-i` *`INTERFACE_NAME`* parameter. The service is specified by the `-s` *`SERVICE_NAME`* parameter.

`update`

The **`netconfig update`** command updates the network configuration using current settings. This is useful when the policy or the static configuration has changed. Use the `-m` *`MODULE_TYPE`* parameter to update a specified service only (`dns`, `nis`, or `ntp`).

The netconfig policy and the static configuration settings are defined either manually or using YaST in the `/etc/sysconfig/network/config` file. The dynamic configuration settings provided by autoconfiguration tools such as DHCP or PPP are delivered directly by these tools with the **`netconfig modify`** and **`netconfig remove`** actions. When NetworkManager is enabled, netconfig (in policy mode `auto`) uses only NetworkManager settings, ignoring settings from any other interfaces configured using the traditional ifup method. If NetworkManager does not provide any setting, static settings are used as a fallback. A mixed usage of NetworkManager and the **`wicked`** method is not supported.

For more information about **`netconfig`**, see **`man 8 netconfig`**.

23.6.2.10 `/etc/hosts`

In this file, shown in *Example 23.7, "`/etc/hosts`"*, IP addresses are assigned to host names. If no name server is implemented, all hosts to which an IP connection will be set up must be listed here. For each host, enter a line consisting of the IP address, the fully qualified host name, and the host name into the file. The IP address must be at the beginning of the line and the entries separated by blanks and tabs. Comments are always preceded by the `#` sign.

EXAMPLE 23.7: `/etc/hosts`

```
127.0.0.1 localhost
192.168.2.100 jupiter.example.com jupiter
192.168.2.101 venus.example.com venus
```

23.6.2.11 `/etc/networks`

Here, network names are converted to network addresses. The format is similar to that of the `hosts` file, except the network names precede the addresses. See *Example 23.8, "`/etc/networks`"*.

EXAMPLE 23.8: `/etc/networks`

```
loopback     127.0.0.0
localnet     192.168.0.0
```

23.6.2.12 `/etc/host.conf`

Name resolution—the translation of host and network names via the *resolver* library—is controlled by this file. This file is only used for programs linked to libc4 or libc5. For current glibc programs, refer to the settings in `/etc/nsswitch.conf`. Each parameter must always be entered on a separate line. Comments are preceded by a `#` sign. *Table 23.2, "Parameters for /etc/host.conf"* shows the parameters available. A sample `/etc/host.conf` is shown in *Example 23.9, "`/etc/host.conf`"*.

TABLE 23.2: PARAMETERS FOR /ETC/HOST.CONF

order *hosts, bind*	Specifies in which order the services are accessed for the name resolution. Available arguments are (separated by blank spaces or commas):
	hosts: searches the `/etc/hosts` file
	bind: accesses a name server
	nis: uses NIS
multi *on/off*	Defines if a host entered in `/etc/hosts` can have multiple IP addresses.
nospoof *on* spoofalert *on/off*	These parameters influence the name server *spoofing* but do not exert any influence on the network configuration.
trim *domainname*	The specified domain name is separated from the host name after host name resolution (as long as the host name includes the domain name). This option is useful only if names from the local domain are in the `/etc/hosts` file, but should still be recognized with the attached domain names.

EXAMPLE 23.9: `/etc/host.conf`

```
# We have named running
order hosts bind
# Allow multiple address
multi on
```

23.6.2.13 `/etc/nsswitch.conf`

The introduction of the GNU C Library 2.0 was accompanied by the introduction of the *Name Service Switch* (NSS). Refer to the `nsswitch.conf(5)` man page and *The GNU C Library Reference Manual* for details.

The order for queries is defined in the file /etc/nsswitch.conf. A sample nsswitch.conf is shown in *Example 23.10, "/etc/nsswitch.conf"*. Comments are preceded by # signs. In this example, the entry under the hosts database means that a request is sent to /etc/hosts (files) via DNS.

EXAMPLE 23.10: /etc/nsswitch.conf

```
passwd:     compat
group:      compat

hosts:      files dns
networks:   files dns

services:   db files
protocols:  db files
rpc:        files
ethers:     files
netmasks:   files
netgroup:   files nis
publickey:  files

bootparams: files
automount:  files nis
aliases:    files nis
shadow:     compat
```

The "databases" available over NSS are listed in *Table 23.3, "Databases available via /etc/nsswitch.conf"*. The configuration options for NSS databases are listed in *Table 23.4, "Configuration options for NSS "databases""*.

TABLE 23.3: DATABASES AVAILABLE VIA /ETC/NSSWITCH.CONF

aliases	Mail aliases implemented by sendmail; see **man** 5 aliases.
ethers	Ethernet addresses.
netmasks	List of networks and their subnet masks. Only needed, if you use subnetting.
group	User groups used by getgrent. See also the man page for **group**.

`hosts`	Host names and IP addresses, used by `gethostbyname` and similar functions.
`netgroup`	Valid host and user lists in the network for controlling access permissions; see the `netgroup(5)` man page.
`networks`	Network names and addresses, used by `getnetent`.
`publickey`	Public and secret keys for Secure_RPC used by NFS and NIS+.
`passwd`	User passwords, used by `getpwent`; see the `passwd(5)` man page.
`protocols`	Network protocols, used by `getprotoent`; see the `protocols(5)` man page.
`rpc`	Remote procedure call names and addresses, used by `getrpcbyname` and similar functions.
`services`	Network services, used by `getservent`.
`shadow`	Shadow passwords of users, used by `getspnam`; see the `shadow(5)` man page.

TABLE 23.4: CONFIGURATION OPTIONS FOR NSS “DATABASES”

`files`	directly access files, for example, `/etc/aliases`
`db`	access via a database
`nis`, `nisplus`	NIS, see also *Book “Security and Hardening Guide”, Chapter 3 “Using NIS”*
`dns`	can only be used as an extension for `hosts` and `networks`

`compat`	can only be used as an extension for `passwd`, `shadow` and `group`

23.6.2.14 `/etc/nscd.conf`

This file is used to configure nscd (name service cache daemon). See the `nscd(8)` and `nscd.conf(5)` man pages. By default, the system entries of `passwd`, `groups` and `hosts`are cached by nscd. This is important for the performance of directory services, like NIS and LDAP, because otherwise the network connection needs to be used for every access to names, groups or hosts.

If the caching for `passwd` is activated, it usually takes about fifteen seconds until a newly added local user is recognized. Reduce this waiting time by restarting nscd with:

```
> sudo systemctl restart nscd
```

23.6.2.15 `/etc/hostname`

`/etc/hostname` contains the fully qualified host name (FQHN). The fully qualified host name is the host name with the domain name attached. This file must contain only one line (in which the host name is set). It is read while the machine is booting.

23.6.3 Testing the configuration

Before you write your configuration to the configuration files, you can test it. To set up a test configuration, use the **`ip`** command. To test the connection, use the **`ping`** command.

The command **`ip`** changes the network configuration directly without saving it in the configuration file. Unless you enter your configuration in the correct configuration files, the changed network configuration is lost on reboot.

> **Note: `ifconfig` and `route` are obsolete**
>
> The **`ifconfig`** and **`route`** tools are obsolete. Use **`ip`** instead. **`ifconfig`**, for example, limits interface names to 9 characters.

23.6.3.1 Configuring a network interface with `ip`

`ip` is a tool to show and configure network devices, routing, policy routing, and tunnels.

`ip` is a very complex tool. Its common syntax is `ip` *`OPTIONS OBJECT COMMAND`*. You can work with the following objects:

link
: This object represents a network device.

address
: This object represents the IP address of device.

neighbor
: This object represents an ARP or NDISC cache entry.

route
: This object represents the routing table entry.

rule
: This object represents a rule in the routing policy database.

maddress
: This object represents a multicast address.

mroute
: This object represents a multicast routing cache entry.

tunnel
: This object represents a tunnel over IP.

If no command is given, the default command is used (usually `list`).

Change the state of a device with the command:

```
> sudo ip link set DEV_NAME
```

For example, to deactivate device eth0, enter

```
> sudo ip link set eth0 down
```

To activate it again, use

```
> sudo ip link set eth0 up
```

Tip: Disconnecting NIC device

If you deactivate a device with

```
> sudo ip link set DEV_NAME down
```

it disables the network interface on a software level.

If you want to simulate losing the link as if the Ethernet cable is unplugged or the connected switch is turned off, run

```
> sudo ip link set DEV_NAME carrier off
```

For example, while **ip link set** *DEV_NAME* **down** drops all routes using *DEV_NAME*, **ip link set DEV carrier off** does not. Be aware that **carrier off** requires support from the network device driver.

To connect the device back to the physical network, run

```
> sudo ip link set DEV_NAME carrier on
```

After activating a device, you can configure it. To set the IP address, use

```
> sudo ip addr add IP_ADDRESS + dev DEV_NAME
```

For example, to set the address of the interface eth0 to 192.168.12.154/30 with standard broadcast (option `brd`), enter

```
> sudo ip addr add 192.168.12.154/30 brd + dev eth0
```

To have a working connection, you must also configure the default gateway. To set a gateway for your system, enter

```
> sudo ip route add default via gateway_ip_address
```

To display all devices, use

```
> sudo ip link ls
```

To display the running interfaces only, use

```
> sudo ip link ls up
```

To print interface statistics for a device, enter

```
> sudo ip -s link ls DEV_NAME
```

To view additional useful information, specifically about virtual network devices, enter

```
> sudo ip -d link ls DEV_NAME
```

Moreover, to view network layer (IPv4, IPv6) addresses of your devices, enter

```
> sudo ip addr
```

In the output, you can find information about MAC addresses of your devices. To show all routes, use

```
> sudo ip route show
```

For more information about using **ip**, enter **ip** `help` or see the **man 8 ip** manual page. The `help` option is also available for all **ip** subcommands, such as:

```
> sudo ip addr help
```

Find the **ip** manual in `/usr/share/doc/packages/iproute2/ip-cref.pdf`.

23.6.3.2 Testing a connection with ping

The **ping** command is the standard tool for testing whether a TCP/IP connection works. It uses the ICMP protocol to send a small data packet, ECHO_REQUEST datagram, to the destination host, requesting an immediate reply. If this works, **ping** displays a message to that effect. This indicates that the network link is functioning.

ping does more than only test the function of the connection between two computers: it also provides some basic information about the quality of the connection. In *Example 23.11, "Output of the command ping"*, you can see an example of the **ping** output. The second-to-last line contains information about the number of transmitted packets, packet loss, and total time of **ping** running.

As the destination, you can use a host name or IP address, for example, **ping** `example.com` or **ping** `192.168.3.100`. The program sends packets until you press `Ctrl`–`C`.

If you only need to check the functionality of the connection, you can limit the number of the packets with the `-c` option. For example to limit ping to three packets, enter **ping** `-c 3 example.com`.

EXAMPLE 23.11: OUTPUT OF THE COMMAND PING

```
ping -c 3 example.com
PING example.com (192.168.3.100) 56(84) bytes of data.
64 bytes from example.com (192.168.3.100): icmp_seq=1 ttl=49 time=188 ms
```

```
64 bytes from example.com (192.168.3.100): icmp_seq=2 ttl=49 time=184 ms
64 bytes from example.com (192.168.3.100): icmp_seq=3 ttl=49 time=183 ms
--- example.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 183.417/185.447/188.259/2.052 ms
```

The default interval between two packets is one second. To change the interval, ping provides the option `-i`. For example, to increase the ping interval to ten seconds, enter **`ping`** `-i 10 example.com`.

In a system with multiple network devices, it is sometimes useful to send the ping through a specific interface address. To do so, use the `-I` option with the name of the selected device, for example, **`ping`** `-I wlan1 example.com`.

For more options and information about using ping, enter **`ping`** `-h` or see the `ping (8)` man page.

Tip: Pinging IPv6 addresses

For IPv6 addresses use the **`ping6`** command. Note, to ping link-local addresses, you must specify the interface with `-I`. The following command works, if the address is reachable via `eth1`:

```
ping6 -I eth1 fe80::117:21ff:feda:a425
```

23.6.4 Unit files and start-up scripts

Apart from the configuration files described above, there are also systemd unit files and various scripts that load the network services while the machine is booting. These are started when the system is switched to the `multi-user.target` target. Some of these unit files and scripts are described in *Some unit files and start-up scripts for network programs*. For more information about `systemd`, see *Chapter 19, The `systemd` daemon* and for more information about the `systemd` targets, see the man page of `systemd.special` (**`man systemd.special`**).

SOME UNIT FILES AND START-UP SCRIPTS FOR NETWORK PROGRAMS

`network.target`

: `network.target` is the systemd target for networking, but its mean depends on the settings provided by the system administrator.

  For more information, see https://www.freedesktop.org/wiki/Software/systemd/NetworkTarget/.

`multi-user.target`
: `multi-user.target` is the systemd target for a multiuser system with all required network services.

`rpcbind`
: Starts the rpcbind utility that converts RPC program numbers to universal addresses. It is needed for RPC services, such as an NFS server.

`ypserv`
: Starts the NIS server.

`ypbind`
: Starts the NIS client.

`/etc/init.d/nfsserver`
: Starts the NFS server.

`/etc/init.d/postfix`
: Controls the postfix process.

23.7 Setting up bonding devices

For some systems, there is a desire to implement network connections that comply to more than the standard data security or availability requirements of a typical Ethernet device. In these cases, several Ethernet devices can be aggregated to a single bonding device.

The configuration of the bonding device is done by means of bonding module options. The behavior is mainly affected by the mode of the bonding device. By default, this is `active-backup` which means that a different bond port will become active if the active port fails. The following bonding modes are available:

0 **(balance-rr)**
: Packets are transmitted in round-robin fashion from the first to the last available interface. Provides fault tolerance and load balancing.

1 **(active-backup)**
: Only one network interface is active. If it fails, a different interface becomes active. This setting is the default for SUSE Linux Enterprise Desktop. Provides fault tolerance.

2 (balance-xor)

Traffic is split between all available interfaces based on the number of devices included in the bonding. It requires support from the switch. Provides fault tolerance and load balancing.

3 (broadcast)

All traffic is broadcast on all interfaces. Requires support from the switch. Provides fault tolerance.

4 (802.3ad)

Aggregates interfaces into groups that share the same speed and duplex settings. Requires **`ethtool`** support in the interface drivers, and a switch that supports and is configured for IEEE 802.3ad Dynamic link aggregation. Provides fault tolerance and load balancing.

5 (balance-tlb)

Adaptive transmit load balancing. Requires **`ethtool`** support in the interface drivers but not switch support. Provides fault tolerance and load balancing.

6 (balance-alb)

Adaptive load balancing. Requires **`ethtool`** support in the interface drivers but not switch support. Provides fault tolerance and load balancing.

For a more detailed description of the modes, see https://www.kernel.org/doc/Documentation/networking/bonding.txt.

Tip: Bonding and Xen

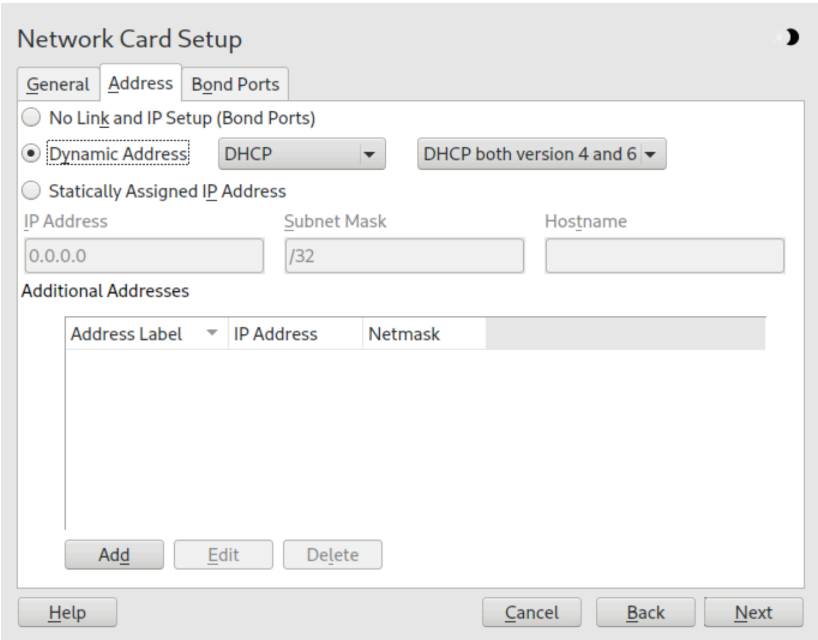
Using bonding devices is only of interest for machines where you have multiple real network cards available. In most configurations, this means that you should use the bonding configuration only in Dom0. Only if you have multiple network cards assigned to a VM Guest system it may also be useful to set up the bond in a VM Guest.

Note: IBM POWER: Bonding modes 5 and 6 (balance-tlb / balance-alb) unsupported by ibmveth

There is a conflict with the tlb/alb bonding configuration and Power firmware. In short, the bonding driver in tlb/alb mode sends Ethernet Loopback packets with both the source and destination MAC addresses listed as the Virtual Ethernet MAC address. These packets are not supported by Power firmware. Therefore bonding modes 5 and 6 are unsupported by ibmveth.

To configure a bonding device, use the following procedure:

1. Run *YaST* › *System* › *Network Settings*.
2. Use *Add* and change the *Device Type* to *Bond*. Proceed with *Next*.



3. Select how to assign the IP address to the bonding device. Three methods are at your disposal:

   - No IP Address
   - Dynamic Address (with DHCP or Zeroconf)
   - Statically assigned IP Address

   Use the method that is appropriate for your environment.

4. In the *Bond Ports* tab, select the Ethernet devices that should be included into the bond by activating the related check box.

5. Edit the *Bond Driver Options* and choose a bonding mode.

6. Make sure that the parameter `miimon=100` is added to the *Bond Driver Options*. Without this parameter, the data integrity is not checked regularly.

7. Click *Next* and leave YaST with *OK* to create the device.

23.7.1 Hotplugging of bond ports

In specific network environments (such as High Availability), there are cases when you need to replace a bond port interface with another one. The reason may be a constantly failing network device. The solution is to set up hotplugging of bond ports.

The bond is configured as usual (according to **`man 5 ifcfg-bonding`**), for example:

```
ifcfg-bond0
          STARTMODE='auto' # or 'onboot'
          BOOTPROTO='static'
          IPADDR='192.168.0.1/24'
          BONDING_MASTER='yes'
          BONDING_SLAVE_0='eth0'
          BONDING_SLAVE_1='eth1'
          BONDING_MODULE_OPTS='mode=active-backup miimon=100'
```

The bond ports are specified with `STARTMODE=hotplug` and `BOOTPROTO=none`:

```
ifcfg-eth0
          STARTMODE='hotplug'
          BOOTPROTO='none'

ifcfg-eth1
          STARTMODE='hotplug'
          BOOTPROTO='none'
```

`BOOTPROTO=none` uses the **`ethtool`** options (when provided), but does not set the link up on **`ifup eth0`**. The reason is that the bond port interface is controlled by the bond device.

`STARTMODE=hotplug` causes the bond port interface to join the bond automatically when it is available.

The udev rules in /etc/udev/rules.d/70-persistent-net.rules need to be changed to match the device by bus ID (udev KERNELS keyword equal to "SysFS BusID" as visible in **hwinfo --netcard**) instead of by MAC address. This allows replacement of defective hardware (a network card in the same slot but with a different MAC) and prevents confusion when the bond changes the MAC address of all its bond ports.

For example:

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
KERNELS=="0000:00:19.0", ATTR{dev_id}=="0x0", ATTR{type}=="1",
KERNEL=="eth*", NAME="eth0"
```

At boot time, the systemd network.service does not wait for the hotplug bond ports, but for the bond to become ready, which requires at least one available bond port. When one of the bond port interfaces gets removed (unbind from NIC driver, **rmmod** of the NIC driver or true PCI hotplug remove) from the system, the kernel removes it from the bond automatically. When a new card is added to the system (replacement of the hardware in the slot), udev renames it using the bus-based persistent name rule to the name of the bond port, and calls **ifup** for it. The **ifup** call automatically joins it into the bond.

23.7.2 Predictable naming scheme

There is a limitation with the persistent network name generator and bonding. When an interface is enslaved to a bond, the MAC address is temporarily overridden and set to match the MAC address of the bond interface. This creates inconsistencies in the 70-persistent-net.rules file when the rules for enslaved interfaces are based on MAC addresses.

If an uevent occurs for an enslaved NIC with an updated MAC address,udev may try to rename it using the name of the first enslaved interface, resulting in a temporary name such as rename4. Modifying the persistent network name generator to avoid MAC-based rules is not feasible due to the high risk of regressions and the necessity of MAC filtering in certain cases. The persistent network name generator is obsolete and replaced by the predictable naming scheme, which offers more configurability and avoids reliance on MAC addresses. We recommend enabling the predictable naming scheme by appending net.ifnames=1' to the Boot Options prompt during a new system installation.

PROCEDURE 23.1: ENABLING THE NAMING SCHEME ON AN ALREADY INSTALLED SYSTEM

1. If the biosdevname package is installed, uninstall it.

```
> sudo zypper rm biosdevname
```

2. Back up existing naming rules from `/etc/udev/rules.d` if they exist. For example:

```
> sudo cp /etc/udev/rules.d/70-persistent-net.rules /backup
```

> **Note**
>
> Remove the file `/etc/udev/rules.d/70-persistent-net.rules`, if it exists, to enable predictable network interface names.

3. Regenerate the initrd.

```
> sudo dracut -f
```

4. Start the YaST Boot Loader module and append `net.ifnames=1` to the kernel command line. Confirm with *OK* and reboot the system.

5. If the system is using Wicked as a network manager, align the configuration of network interfaces. Use the **`yast lan`** module or rename `/etc/sysconfig/network/ifcfg-*` files accordingly. Then restart Wicked.

```
> sudo systemctl restart wicked.service
```

23.8 Setting up team devices for Network Teaming

The term “link aggregation” is the general term which describes combining (or aggregating) a network connection to provide a logical layer. Sometimes you find the terms “channel teaming”, “Ethernet bonding”, “port trunking”, etc. which are synonyms and refer to the same concept.

This concept is widely known as “bonding” and was originally integrated into the Linux kernel (see *Section 23.7, “Setting up bonding devices”* for the original implementation). The term *Network Teaming* is used to refer to the new implementation of this concept.

The main difference between bonding and Network Teaming is that teaming supplies a set of small kernel modules responsible for providing an interface for teamd instances. Everything else is handled in user space. This is different from the original bonding implementation which contains all of its functionality exclusively in the kernel. For a comparison refer to *Table 23.5, “Feature comparison between bonding and team”*.

TABLE 23.5: FEATURE COMPARISON BETWEEN BONDING AND TEAM

Feature	Bonding	Team
broadcast, round-robin TX policy	yes	yes
active-backup TX policy	yes	yes
LACP (802.3ad) support	yes	yes
hash-based TX policy	yes	yes
user can set hash function	no	yes
TX load-balancing support (TLB)	yes	yes
TX load-balancing support for LACP	no	yes
Ethtool link monitoring	yes	yes
ARP link monitoring	yes	yes
NS/NA (IPV6) link monitoring	no	yes
RCU locking on TX/RX paths	no	yes
port prio and stickiness	no	yes
separate per-port link monitoring setup	no	yes
multiple link monitoring setup	limited	yes
VLAN support	yes	yes
multiple device stacking	yes	yes
Source: https://libteam.org/files/teamdev.pp.pdf		

Both implementations, bonding and Network Teaming, can be used in parallel. Network Teaming is an alternative to the existing bonding implementation. It does not replace bonding.

Network Teaming can be used for different use cases. The two most important use cases are explained later and involve:

- Load balancing between different network devices.
- Failover from one network device to another in case one of the devices should fail.

Currently, there is no YaST module to support creating a teaming device. You need to configure Network Teaming manually. The general procedure is shown below which can be applied for all your Network Teaming configurations:

PROCEDURE 23.2: GENERAL PROCEDURE

1. Install the package `libteam-tools`:

   ```
   > sudo zypper in libteam-tools
   ```

2. Create a configuration file under `/etc/sysconfig/network/`. Usually it will be `ifcfg-team0`. If you need more than one Network Teaming device, give them ascending numbers. This configuration file contains several variables which are explained in the man pages (see **`man ifcfg`** and **`man ifcfg-team`**). An example configuration can be found in your system in the file `/etc/sysconfig/network/ifcfg.template`.

3. Remove the configuration files of the interfaces which will be used for the teaming device (usually `ifcfg-eth0` and `ifcfg-eth1`).
   It is recommended to make a backup and remove both files. Wicked will re-create the configuration files with the necessary parameters for teaming.

4. Optionally, check if everything is included in Wicked's configuration file:

   ```
   > sudo wicked show-config
   ```

5. Start the Network Teaming device `team0`:

   ```
   > sudo wicked ifup team0
   ```

   In case you need additional debug information, use the option `--debug all` after the **`all`** subcommand.

6. Check the status of the Network Teaming device. This can be done by the following commands:

   - Get the state of the teamd instance from Wicked:

     ```
     > sudo wicked ifstatus --verbose team0
     ```

   - Get the state of the entire instance:

     ```
     > sudo teamdctl team0 state
     ```

   - Get the systemd state of the teamd instance:

     ```
     > sudo systemctl status teamd@team0
     ```

   Each of them shows a slightly different view depending on your needs.

7. In case you need to change something in the `ifcfg-team0` file afterward, reload its configuration with:

   ```
   > sudo wicked ifreload team0
   ```

Do *not* use **`systemctl`** for starting or stopping the teaming device! Instead, use the **`wicked`** command as shown above.

To completely remove the team device, use this procedure:

PROCEDURE 23.3: REMOVING A TEAM DEVICE

1. Stop the Network Teaming device `team0`:

   ```
   > sudo wicked ifdown team0
   ```

2. Rename the file `/etc/sysconfig/network/ifcfg-team0` to `/etc/sysconfig/network/.ifcfg-team0`. Inserting a dot in front of the file name makes it "invisible" for wicked. If you really do not need the configuration anymore, you can also remove the file.

3. Reload the configuration:

   ```
   > sudo wicked ifreload all
   ```

23.8.1 Use case: load balancing with Network Teaming

Load balancing is used to improve bandwidth. Use the following configuration file to create a Network Teaming device with load balancing capabilities. Proceed with *Procedure 23.2, “General procedure”* to set up the device. Check the output with **`teamdctl`**.

EXAMPLE 23.12: CONFIGURATION FOR LOAD BALANCING WITH NETWORK TEAMING

```
STARTMODE=auto ❶
BOOTPROTO=static ❷
IPADDRESS="192.168.1.1/24" ❷
IPADDR6="fd00:deca:fbad:50::1/64" ❷

TEAM_RUNNER="loadbalance" ❸
TEAM_LB_TX_HASH="ipv4,ipv6,eth,vlan"
TEAM_LB_TX_BALANCER_NAME="basic"
TEAM_LB_TX_BALANCER_INTERVAL="100"

TEAM_PORT_DEVICE_0="eth0" ❹
TEAM_PORT_DEVICE_1="eth1" ❹

TEAM_LW_NAME="ethtool" ❺
TEAM_LW_ETHTOOL_DELAY_UP="10" ❻
TEAM_LW_ETHTOOL_DELAY_DOWN="10" ❻
```

❶ Controls the start of the teaming device. The value of `auto` means, the interface will be set up when the network service is available and will be started automatically on every reboot. In case you need to control the device yourself (and prevent it from starting automatically), set `STARTMODE` to `manual`.

❷ Sets a static IP address (here `192.168.1.1` for IPv4 and `fd00:deca:fbad:50::1` for IPv6). If the Network Teaming device should use a dynamic IP address, set `BOOTPROTO="dhcp"` and remove (or comment) the line with `IPADDRESS` and `IPADDR6`.

❸ Sets `TEAM_RUNNER` to `loadbalance` to activate the load balancing mode.

❹ Specifies one or more devices which should be aggregated to create the Network Teaming device.

❺ Defines a link watcher to monitor the state of subordinate devices. The default value `ethtool` checks only if the device is up and accessible. This makes this check fast enough. However, it does not check if the device can really send or receive packets.
If you need a higher confidence in the connection, use the `arp_ping` option. This sends pings to an arbitrary host (configured in the `TEAM_LW_ARP_PING_TARGET_HOST` variable). The Network Teaming device is considered to be up only if the replies are received.

6. Defines the delay in milliseconds between the link coming up (or down) and the runner being notified.

23.8.2 Use case: failover with Network Teaming

Failover is used to ensure high availability of a critical Network Teaming device by involving a parallel backup network device. The backup network device is running all the time and takes over if and when the main device fails.

Use the following configuration file to create a Network Teaming device with failover capabilities. Proceed with *Procedure 23.2, "General procedure"* to set up the device. Check the output with **teamdctl**.

EXAMPLE 23.13: CONFIGURATION FOR DHCP NETWORK TEAMING DEVICE

```
STARTMODE=auto  1
BOOTPROTO=static  2
IPADDR="192.168.1.2/24"  2
IPADDR6="fd00:deca:fbad:50::2/64"  2

TEAM_RUNNER=activebackup  3
TEAM_PORT_DEVICE_0="eth0"  4
TEAM_PORT_DEVICE_1="eth1"  4

TEAM_LW_NAME=ethtool  5
TEAM_LW_ETHTOOL_DELAY_UP="10"  6
TEAM_LW_ETHTOOL_DELAY_DOWN="10"  6
```

1. Controls the start of the teaming device. The value of `auto` means the interface will be set up when the network service is available and will be started automatically on every reboot. In case you need to control the device yourself (and prevent it from starting automatically), set `STARTMODE` to `manual`.
2. Sets a static IP address (here `192.168.1.2` for IPv4 and `fd00:deca:fbad:50::2` for IPv6). If the Network Teaming device should use a dynamic IP address, set `BOOTPROTO="dhcp"` and remove (or comment) the line with `IPADDRESS` and `IPADDR6`.
3. Sets `TEAM_RUNNER` to `activebackup` to activate the failover mode.
4. Specifies one or more devices which should be aggregated to create the Network Teaming device.

5. Defines a link watcher to monitor the state of subordinate devices. The default value `ethtool` checks only if the device is up and accessible. This makes this check fast enough. However, it does not check if the device can really send or receive packets.
   If you need a higher confidence in the connection, use the `arp_ping` option. This sends pings to an arbitrary host (configured in the `TEAM_LW_ARP_PING_TARGET_HOST` variable). Only if the replies are received, the Network Teaming device is considered to be up.
6. Defines the delay in milliseconds between the link coming up (or down) and the runner being notified.

23.8.3 Use case: VLAN over team device

VLAN is an abbreviation of *Virtual Local Area Network*. It allows the running of multiple *logical* (virtual) Ethernets over one single physical Ethernet. It logically splits the network into different broadcast domains so that packets are only switched between ports that are designated for the same VLAN.

The following use case creates two static VLANs on top of a team device:

- `vlan0`, bound to the IP address `192.168.10.1`
- `vlan1`, bound to the IP address `192.168.20.1`

Proceed as follows:

1. Enable the VLAN tags on your switch. To use load balancing for your team device, your switch needs to be capable of *Link Aggregation Control Protocol* (LACP) (802.3ad). Consult your hardware manual about the details.
2. Decide if you want to use load balancing or failover for your team device. Set up your team device as described in *Section 23.8.1, “Use case: load balancing with Network Teaming”* or *Section 23.8.2, “Use case: failover with Network Teaming”*.
3. In `/etc/sysconfig/network` create a file `ifcfg-vlan0` with the following content:

```
STARTMODE="auto"
BOOTPROTO="static" 1
IPADDR='192.168.10.1/24' 2
ETHERDEVICE="team0" 3
VLAN_ID="0" 4
VLAN='yes'
```

1. Defines a fixed IP address, specified in `IPADDR`.
2. Defines the IP address, here with its netmask.
3. Contains the real interface to use for the VLAN interface, here our team device (`team0`).
4. Specifies a unique ID for the VLAN. Preferably, the file name and the `VLAN_ID` corresponds to the name `ifcfg-vlan`*`VLAN_ID`*. In our case `VLAN_ID` is `0` which leads to the file name `ifcfg-vlan0`.

4. Copy the file `/etc/sysconfig/network/ifcfg-vlan0` to `/etc/sysconfig/network/ifcfg-vlan1` and change the following values:
   - `IPADDR` from `192.168.10.1/24` to `192.168.20.1/24`.
   - `VLAN_ID` from `0` to `1`.
5. Start the two VLANs:

```
# wicked ifup vlan0 vlan1
```

6. Check the output of **`ifconfig`**:

```
# ifconfig -a
[...]
vlan0     Link encap:Ethernet  HWaddr 08:00:27:DC:43:98
          inet addr:192.168.10.1 Bcast:192.168.10.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fedc:4398/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:816 (816.0 b)

vlan1     Link encap:Ethernet  HWaddr 08:00:27:DC:43:98
          inet addr:192.168.20.1 Bcast:192.168.20.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fedc:4398/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:816 (816.0 b)
```

24 Printer operation

Revision History

2024-06-21

SUSE® Linux Enterprise Desktop supports printing with many types of printers, including remote network printers. Printers can be configured manually or with YaST. For configuration instructions, refer to *Chapter 34, Setting up a printer*. Both graphical and command line utilities are available for starting and managing print jobs. If your printer does not work as expected, refer to *Section 24.8, "Troubleshooting"*.

CUPS (Common Unix Printing System) is the standard print system in SUSE Linux Enterprise Desktop.

Printers can be distinguished by interface, such as USB or network, and printer language. When buying a printer, make sure that the printer has an interface that is supported (USB, Ethernet or Wi-Fi) and a suitable printer language. Printers can be categorized on the basis of the following three classes of printer languages:

PostScript printers

PostScript is the printer language in which most print jobs in Linux and Unix are generated and processed by the internal print system. If PostScript documents can be processed directly by the printer and do not need to be converted in additional stages in the print system, the number of potential error sources is reduced.

Currently PostScript is being replaced by PDF as the standard print job format. PostScript +PDF printers that can directly print PDF (in addition to PostScript) already exist. For traditional PostScript printers PDF needs to be converted to PostScript in the printing workflow.

Standard printers (languages like PCL and ESC/p)

For known printer languages, the print system can convert PostScript jobs to the respective printer language with Ghostscript. This processing stage is called interpreting. The best-known languages are PCL (which is used by HP printers and their clones) and ESC/P (which is used by Epson printers). These printer languages are normally supported by Linux and produce an adequate print result. Linux may not be able to address certain special printer functions. Except for HP and Epson, there are currently no printer manufacturers who develop Linux drivers and make them available to Linux distributors under an open source license.

Proprietary printers (also called GDI printers)

These printers do not support any of the common printer languages. They use their own undocumented printer languages, which are subject to change when a new edition of a model is released. Normally, only Windows drivers are available for these printers. See *Section 24.8.1, "Printers without standard printer language support"* for more information.

Before you buy a new printer, refer to the following sources to check how well the printer you intend to buy is supported:

https://www.openprinting.org/printers

The OpenPrinting home page with the printer database. The database shows the latest Linux support status. However, a Linux distribution can only integrate the drivers available at production time. Accordingly, a printer currently rated as "perfectly supported" may not have had this status when the latest SUSE Linux Enterprise Desktop version was released. Thus, the databases may not necessarily indicate the correct status, but only provide an approximation.

https://www.ghostscript.com

The Ghostscript Web page.

`/usr/share/doc/packages/ghostscript/catalog.devices`

List of built-in Ghostscript drivers.

24.1 The CUPS workflow

The user creates a print job. The print job consists of the data to print plus information for the spooler. This includes the name of the printer or the name of the print queue, and optionally, information for the filter, such as printer-specific options.

At least one dedicated print queue exists for every printer. The spooler holds the print job in the queue until the desired printer is ready to receive data. When the printer is ready, the spooler sends the data through the filter and back-end to the printer.

The filter converts the data generated by the application that is printing (normally PostScript or PDF, but also ASCII, JPEG, etc.) into printer-specific data (PostScript, PCL, ESC/P, etc.). The features of the printer are described in the PPD files. A PPD file contains printer-specific options with the parameters needed to enable them on the printer. The filter system makes sure that options selected by the user are enabled.

If you use a PostScript printer, the filter system converts the data into printer-specific PostScript. This does not require a printer driver. If you use a non-PostScript printer, the filter system converts the data into printer-specific data. This requires a printer driver suitable for your printer. The back-end receives the printer-specific data from the filter then passes it to the printer.

24.2 Methods and protocols for connecting printers

There are multiple possibilities for connecting a printer to the system. The configuration of CUPS does not distinguish between a local printer and a printer connected to the system over the network. For more information about the printer connection, read the article *CUPS in a Nutshell* at https://en.opensuse.org/SDB:CUPS_in_a_Nutshell.

Warning: Changing cable connections in a running system

When connecting the printer to the machine, do not forget that only USB devices can be plugged in or unplugged during operation. To avoid damaging your system or printer, shut down the system before changing any connections that are not USB.

24.3 Installing the software

PPD (PostScript printer description) is the computer language that describes the properties, like resolution, and options, such as the availability of a duplex unit. These descriptions are required for using printer options in CUPS. Without a PPD file, the print data would be forwarded to the printer in a "raw" state, which is not desired.

To configure a PostScript printer, the best approach is to get a suitable PPD file. Many PPD files are available in the packages `manufacturer-PPDs` and `OpenPrintingPPDs-postscript`. See *Section 24.7.3, "PPD files in multiple packages"* and *Section 24.8.2, "No suitable PPD file available for a PostScript printer"*.

New PPD files can be stored in the directory `/usr/share/cups/model/` or added to the print system with YaST as described in *Section 34.1.1, "Adding drivers with YaST"*. Subsequently, the PPD file can be selected during the printer setup.

Be careful if a printer manufacturer wants you to install entire software packages. This kind of installation may result in the loss of the support provided by SUSE Linux Enterprise Desktop. Also, print commands may work differently and the system may no longer be able to address devices of other manufacturers. For this reason, the installation of manufacturer software is not recommended.

24.4 Network printers

A network printer can support multiple protocols. Although most of the supported protocols are standardized, certain manufacturers modify the standard. Manufacturers then provide drivers for only a few operating systems and Linux drivers are rarely provided. The current situation is such that you cannot act on the assumption that every protocol works smoothly in Linux. Therefore, you may need to experiment with several options to achieve a functional configuration.

CUPS supports the `socket`, `LPD`, `IPP` and `smb` protocols.

socket

Socket refers to a connection in which the plain print data is sent directly to a TCP socket. Socket port numbers that are commonly used are `9100` or `35`. The device URI (uniform resource identifier) syntax is: socket://*IP.OF.THE.PRINTER*:*PORT*, for example: `socket://192.168.2.202:9100/`.

LPD (line printer daemon)

The LPD protocol is described in RFC 1179. Under this protocol, specific job-related data, such as the ID of the print queue, is sent before the actual print data is sent. Therefore, a print queue must be specified when configuring the LPD protocol. The implementations of diverse printer manufacturers are flexible enough to accept any name as the print queue. If necessary, the printer manual should indicate what name to use. LPT, LPT1, LP1 or similar names are often used. The port number for an LPD service is `515`. An example device URI is `lpd://192.168.2.202/LPT1`.

IPP (Internet printing protocol)

IPP is based on the HTTP protocol. With IPP, more job-related data is transmitted than with the other protocols. CUPS uses IPP for internal data transmission. The name of the print queue is necessary to configure IPP correctly. The port number for IPP is `631`. Example device URIs are `ipp://192.168.2.202/ps` and `ipp://192.168.2.202/printers/ps`.

SMB (Windows share)

CUPS also supports printing on printers connected to Windows shares. The protocol used for this purpose is SMB. SMB uses the port numbers `137`, `138` and `139`. Example device URIs are `smb://user:password@workgroup/smb.example.com/printer`, `smb://user:password@smb.example.com/printer`, and `smb://smb.example.com/printer`.

The protocol supported by the printer must be determined before configuration. If the manufacturer does not provide the needed information, the command **`nmap`** (which comes with the `nmap` package) can be used to ascertain the protocol. **`nmap`** checks a host for open ports. For example:

```
> nmap -p 35,137-139,515,631,9100-10000 IP.OF.THE.PRINTER
```

24.5 Configuring CUPS with command line tools

CUPS can be configured with command line tools like **`lpinfo`**, **`lpadmin`** and **`lpoptions`**. You need a device URI consisting of a back-end, such as USB, and parameters. To determine valid device URIs on your system use the command **`lpinfo -v | grep ":/"`**:

```
> sudo lpinfo -v | grep ":/"
direct usb://ACME/FunPrinter%20XL
network socket://192.168.2.253
```

With **`lpadmin`** the CUPS server administrator can add, remove or manage print queues. To add a print queue, use the following syntax:

```
> sudo lpadmin -p QUEUE -v DEVICE-URI -P PPD-FILE -E
```

Then the device (`-v`) is available as *`QUEUE`* (`-p`), using the specified PPD file (`-P`). This means that you must know the PPD file and the device URI to configure the printer manually.

Do not use `-E` as the first option. For all CUPS commands, `-E` as the first argument sets use of an encrypted connection. To enable the printer, `-E` must be used as shown in the following example:

```
> sudo lpadmin -p ps -v usb://ACME/FunPrinter%20XL -P \
/usr/share/cups/model/Postscript.ppd.gz -E
```

The following example configures a network printer:

```
> sudo lpadmin -p ps -v socket://192.168.2.202:9100/ -P \
```

```
/usr/share/cups/model/Postscript-level1.ppd.gz -E
```

For more options of **lpadmin**, see the man page of `lpadmin(8)`.

During printer setup, certain options are set as default. These options can be modified for every print job (depending on the print tool used). Changing these default options with YaST is also possible. Using command line tools, set default options as follows:

1. First, list all options:

   ```
   > sudo lpoptions -p QUEUE -l
   ```

   Example:

   ```
   Resolution/Output Resolution: 150dpi *300dpi 600dpi
   ```

   The activated default option is identified by a preceding asterisk (`*`).

2. Change the option with **lpadmin**:

   ```
   > sudo lpadmin -p QUEUE -o Resolution=600dpi
   ```

3. Check the new setting:

   ```
   > sudo lpoptions -p QUEUE -l

   Resolution/Output Resolution: 150dpi 300dpi *600dpi
   ```

When a normal user runs **lpoptions**, the settings are written to `~/.cups/lpoptions`. However, `root` settings are written to `/etc/cups/lpoptions`.

24.6 Printing from the command line

To print from the command line, enter **lp -d** *QUEUENAME* *FILENAME*, substituting the corresponding names for *QUEUENAME* and *FILENAME*.

Several applications rely on the **lp** command for printing. In this case, enter the correct command in the application's print dialog, normally without specifying *FILENAME*, for example, **lp -d** *QUEUENAME*.

24.7 Special features in SUSE Linux Enterprise Desktop

Several CUPS features have been adapted for SUSE Linux Enterprise Desktop. The most important changes are covered here.

24.7.1 CUPS and firewall

After completing a default installation of SUSE Linux Enterprise Desktop, `firewalld` is active and the network interfaces are configured to be in the `public` zone, which blocks incoming traffic.

When `firewalld` is active, you may need to configure it to allow clients to browse network printers by allowing `mdns` and `ipp` through the internal network zone. The public zone should never expose printer queues.

(More information about the `firewalld` configuration is available in *Book "Security and Hardening Guide", Chapter 23 "Masquerading and firewalls", Section 23.4 "*`firewalld`*"* and at https://en.opensuse.org/SDB:CUPS_and_SANE_Firewall_settings.)

24.7.1.1 CUPS client

Normally, a CUPS client runs on a regular workstation located in a trusted network environment behind a firewall. In this case, it is recommended to configure the network interface to be in the `Internal Zone`, so the workstation is reachable from within the network.

24.7.1.2 CUPS server

If the CUPS server is part of a trusted network environment protected by a firewall, the network interface should be configured to be in the `Internal Zone` of the firewall. It is not recommended to set up a CUPS server in an untrusted network environment unless you ensure that it is protected by special firewall rules and secure settings in the CUPS configuration.

24.7.2 Browsing for network printers

CUPS servers regularly announce the availability and status information of shared printers over the network. Clients can access this information to display a list of available printers in printing dialogs, for example. This is called “browsing”.

CUPS servers announce their print queues over the network either via the traditional CUPS browsing protocol, or via Bonjour/DNS-SD. To enable browsing network print queues, the service `cups-browsed` needs to run on all clients that print via CUPS servers. `cups-browsed` is not started by default. To start it for the active session, use **`sudo systemctl start cups-browsed`**. To ensure it is automatically started after booting, enable it with **`sudo systemctl enable cups-browsed`** on all clients.

In case browsing does not work after having started `cups-browsed`, the CUPS servers announce the network print queues via Bonjour/DNS-SD. In this case, you need to additionally install the package `avahi` and start the associated service with **`sudo systemctl start avahi-daemon`** on all clients.

See *Section 24.7.1, “CUPS and firewall”* for information on allowing printer browsing through `firewalld`.

24.7.3 PPD files in multiple packages

The YaST printer configuration sets up the queues for CUPS using the PPD files installed in `/usr/share/cups/model`. To find the suitable PPD files for the printer model, YaST compares the vendor and model determined during hardware detection with the vendors and models in all PPD files. For this purpose, the YaST printer configuration generates a database from the vendor and model information extracted from the PPD files.

The configuration using only PPD files and no other information sources has the advantage that the PPD files in `/usr/share/cups/model` can be modified freely. For example, if you have PostScript printers the PPD files can be copied directly to `/usr/share/cups/model` (if they do not already exist in the `manufacturer-PPDs` or `OpenPrintingPPDs-postscript` packages) to achieve an optimum configuration for your printers.

Additional PPD files are provided by the following packages:

- `gutenprint`: the Gutenprint driver and its matching PPDs
- `splix`: the SpliX driver and its matching PPDs

- `OpenPrintingPPDs-ghostscript`: PPDs for Ghostscript built-in drivers
- `OpenPrintingPPDs-hpijs`: PPDs for the HPIJS driver for non-HP printers

24.8 Troubleshooting

The following sections cover the most frequently encountered printer hardware and software problems and ways to solve or circumvent these problems. Among the topics covered are GDI printers, PPD files and port configuration. Common network printer problems, defective printouts, and queue handling are also addressed.

24.8.1 Printers without standard printer language support

These printers do not support any common printer language and can only be addressed with special proprietary control sequences. Therefore they can only work with the operating system versions for which the manufacturer delivers a driver. GDI is a programming interface developed by Microsoft* for graphics devices. Usually the manufacturer delivers drivers only for Windows, and since the Windows driver uses the GDI interface these printers are also called *GDI printers*. The actual problem is not the programming interface, but that these printers can only be addressed with the proprietary printer language of the respective printer model.

Certain GDI printers can be switched to operate either in GDI mode or in one of the standard printer languages. Check the manual of the printer to see whether this is possible. Specific models require special Windows software to do the switch. For example, the Windows printer driver may always switch the printer back into GDI mode when printing from Windows. For other GDI printers there are extension modules for a standard printer language available.

Several manufacturers provide proprietary drivers for their printers. The disadvantage of proprietary printer drivers is that there is no guarantee that these work with the installed print system or that they are suitable for the multiple hardware platforms. In contrast, printers that support a standard printer language do not depend on a special print system version or a special hardware platform.

Instead of spending time trying to make a proprietary Linux driver work, it may be more cost-effective to purchase a printer which supports a standard printer language (preferably PostScript). This would solve the driver problem once and for all, eliminating the need to install and configure special driver software and obtain driver updates that may be required because of new developments in the print system.

24.8.2 No suitable PPD file available for a PostScript printer

If the `manufacturer-PPDs` or `OpenPrintingPPDs-postscript` packages do not contain a suitable PPD file for a PostScript printer, it should be possible to use the PPD file from the driver CD of the printer manufacturer or download a suitable PPD file from the Web page of the printer manufacturer.

If the PPD file is provided as a zip archive (.zip) or a self-extracting zip archive (`.exe`), unpack it with **`unzip`**. First, review the license terms of the PPD file. Then use the **`cupstestppd`** utility to check if the PPD file complies with “Adobe PostScript Printer Description File Format Specification, version 4.3.” If the utility returns “FAIL,” the errors in the PPD files are serious and cause major problems. The problem spots reported by **`cupstestppd`** should be eliminated. If necessary, ask the printer manufacturer for a suitable PPD file.

24.8.3 Network printer connections

Identifying network problems

Connect the printer directly to the computer. For test purposes, configure the printer as a local printer. If this works, the problems are related to the network.

Checking the TCP/IP network

The TCP/IP network and name resolution must be functional.

Checking a remote `lpd`

Use the following command to test if a TCP connection can be established to **`lpd`** (port `515`) on *`HOST`*:

```
> netcat -z HOST 515 && echo ok || echo failed
```

If the connection to **`lpd`** cannot be established, **`lpd`** may not be active or there may be basic network problems.

Provided that the respective **`lpd`** is active and the host accepts queries, run the following command as `root` to query a status report for *`QUEUE`* on remote *`HOST`*:

```
# echo -e "\004queue" \
| netcat -w 2 -p 722 HOST 515
```

If **`lpd`** does not respond, it may not be active or there may be basic network problems. If **`lpd`** responds, the response should show why printing is not possible on the `queue` on `host`. If you receive a response like that shown in *Example 24.1, “Error message from **`lpd`**”*, the problem is caused by the remote **`lpd`**.

EXAMPLE 24.1: ERROR MESSAGE FROM **lpd**

```
lpd: your host does not have line printer access
lpd: queue does not exist
printer: spooling disabled
printer: printing disabled
```

Checking a remote **cupsd**

A CUPS network server can broadcast its queues by default every 30 seconds on UDP port 631. Accordingly, the following command can be used to test whether there is a broadcasting CUPS network server in the network. Make sure to stop your local CUPS daemon before executing the command.

```
> netcat -u -l -p 631 & PID=$! ; sleep 40 ; kill $PID
```

If a broadcasting CUPS network server exists, the output appears as shown in *Example 24.2, “Broadcast from the CUPS network server”*.

EXAMPLE 24.2: BROADCAST FROM THE CUPS NETWORK SERVER

```
ipp://192.168.2.202:631/printers/queue
```

The following command can be used to test if a TCP connection can be established to **cupsd** (port 631) on *HOST*:

```
> netcat -z HOST 631 && echo ok || echo failed
```

If the connection to **cupsd** cannot be established, **cupsd** may not be active or there may be basic network problems. **lpstat -h** *HOST* -l -t returns a status report for all queues on *HOST*, provided the respective **cupsd** is active and the host accepts queries.
The next command can be used to test if the *QUEUE* on *HOST* accepts a print job consisting of a single carriage-return character. Nothing should be printed. A blank page may be ejected.

```
> echo -en "\r" \
| lp -d queue -h HOST
```

Troubleshooting a Network Printer or Print Server Machine

Spoolers running in a print server machine sometimes cause problems when they need to deal with multiple print jobs. Since this is caused by the spooler in the print server machine, there no way to resolve this issue. As a work-around, circumvent the spooler in the print server machine by addressing the printer connected to the print server machine directly with the TCP socket. See *Section 24.4, “Network printers”*.

In this way, the print server machine is reduced to a converter between the multiple forms of data transfer (TCP/IP network and local printer connection). To use this method, you need to know the TCP port on the print server machine. If the printer is connected to the print server machine and turned on, this TCP port can normally be determined with the **`nmap`** utility from the `nmap` package certain time after the print server machine is powered up. For example, **`nmap`** *`IP-address`* may deliver the following output for a print server machine:

```
Port       State      Service
23/tcp     open       telnet
80/tcp     open       http
515/tcp    open       printer
631/tcp    open       cups
9100/tcp   open       jetdirect
```

This output indicates that the printer connected to the print server machine can be addressed via TCP socket on port `9100`. By default, **`nmap`** only checks several commonly known ports listed in `/usr/share/nmap/nmap-services`. To check all possible ports, use the command **`nmap -p`** *`FROM_PORT-TO_PORT`* *`IP_ADDRESS`*. For further information, refer to the man page of **`nmap`**.

Enter a command like

```
> echo -en "\rHello\r\f" | netcat -w 1 IP-address port
cat file | netcat -w 1 IP-address port
```

to send character strings or files directly to the respective port to test if the printer can be addressed on this port.

24.8.4 Defective printouts without error message

For the print system, the print job is completed when the CUPS back-end completes the data transfer to the recipient (printer). If further processing on the recipient fails (for example, if the printer is not able to print the printer-specific data) the print system does not notice this. If the printer cannot print the printer-specific data, select a PPD file that is more suitable for the printer.

24.8.5 Disabled queues

If the data transfer to the recipient fails entirely after several attempts, the CUPS back-end, such as `USB` or `socket`, reports an error to the print system (to **`cupsd`**). The back-end determines how many unsuccessful attempts are appropriate until the data transfer is reported as impossible. As further attempts would be in vain, **`cupsd`** disables printing for the respective queue. After eliminating the cause of the problem, the system administrator must re-enable printing with the command **`cupsenable`**.

24.8.6 CUPS browsing: deleting print jobs

If a CUPS network server broadcasts its queues to the client hosts via browsing and a suitable local **`cupsd`** is active on the client hosts, the client **`cupsd`** accepts print jobs from applications and forwards them to the **`cupsd`** on the server. When **`cupsd`** on the server accepts a print job, it is assigned a new job number. Therefore, the job number on the client host is different from the job number on the server. As a print job is usually forwarded immediately, it cannot be deleted with the job number on the client host This is because the client **`cupsd`** regards the print job as completed when it has been forwarded to the server **`cupsd`**.

To delete the print job on the server, use a command such as **`lpstat -h cups.example.com -o`** to determine the job number on the server. This assumes that the server has not already completed the print job by sending it to the printer. Use the obtained job number to delete the print job on the server as follows:

```
> cancel -h cups.example.com QUEUE-JOBNUMBER
```

24.8.7 Defective print jobs and data transfer errors

If you switch the printer off or shut down the computer during the printing process, print jobs remain in the queue. Printing resumes when the computer (or the printer) is switched back on. Defective print jobs must be removed from the queue with **`cancel`**.

If a print job is corrupted or an error occurs in the communication between the host and the printer, the printer cannot process the data correctly and prints numerous sheets of paper with unintelligible characters. To fix the problem, follow these steps:

1. To stop printing, remove all paper from ink jet printers or open the paper trays of laser printers. High-quality printers have a button for canceling the current printout.

2. The print job may still be in the queue, because jobs are only removed after they are sent to the printer. Use **lpstat -o** or **lpstat -h cups.example.com -o** to check which queue is currently printing. Delete the print job with **cancel** *QUEUE-JOBNUMBER* or **cancel -h cups.example.com** *QUEUE-JOBNUMBER*.

3. Certain data may still be transferred to the printer even though the print job has been deleted from the queue. Check if a CUPS back-end process is still running for the respective queue and stop it.

4. Reset the printer by switching it off for a while. Then insert the paper and turn on the printer.

24.8.8 Debugging CUPS

Use the following generic procedure to locate problems in CUPS:

1. Set **LogLevel debug** in `/etc/cups/cupsd.conf`.

2. Stop **cupsd**.

3. Remove `/var/log/cups/error_log*` to avoid having to search through large log files.

4. Start **cupsd**.

5. Repeat the action that led to the problem.

6. Check the messages in `/var/log/cups/error_log*` to identify the cause of the problem.

24.8.9 More information

In-depth information about printing on SUSE Linux Enterprise Desktop is presented in the openSUSE Support Database at https://en.opensuse.org/Portal:Printing. Solutions to many specific problems are presented in the SUSE Knowledgebase (https://www.suse.com/support/). Locate the relevant articles with a text search for `CUPS`.

25 Graphical user interface

Revision History

2024-05-17

SUSE Linux Enterprise Desktop includes the X.org server, Wayland and the GNOME desktop. This chapter describes the configuration of the graphical user interface for all users.

25.1 X Window System

The X.org server is the de facto standard for implementing the X11 protocol. X is network-based, enabling applications started on one host to be displayed on another host connected over any kind of network (LAN or Internet).

The X Window System needs no configuration in most cases. The hardware is dynamically detected during X start-up. The use of `xorg.conf` is therefore deprecated. If you still need to specify custom options to change the way X behaves, you can still do so by modifying configuration files under `/etc/X11/xorg.conf.d/`.

In SUSE Linux Enterprise Desktop 15 SP7 Wayland is included as an alternative to the X.org server. It can be selected during the installation.

Install the package `xorg-docs` to get more in-depth information about X11. **`man 5 xorg.conf`** tells you more about the format of the manual configuration (if needed). More information on the X11 development can be found on the project's home page at https://www.x.org.

Drivers are found in `xf86-video-*` packages, for example `xf86-video-ati`. Many of the drivers delivered with these packages are described in detail in the related manual page. For example, if you use the `ati` driver, find more information about this driver in **`man 4 ati`**.

Information about third-party drivers is available in `/usr/share/doc/packages/<package_name>`. For example, the documentation of `x11-video-nvidiaG03` is available in `/usr/share/doc/packages/x11-video-nvidiaG04` after the package was installed.

Install the package `xrdp` on a server and use RDP client software to access the server via the remote desktop protocol.

25.2 Installing and configuring fonts

Fonts in Linux can be categorized into two parts:

Outline or vector fonts

Contains a mathematical description as drawing instructions about the shape of a glyph. As such, each glyph can be scaled to arbitrary sizes without loss of quality. Before such a font (or glyph) can be used, the mathematical descriptions need to be transformed into a raster (grid). This process is called *font rasterization*. *Font hinting* (embedded inside the font) improves and optimizes the rendering result for a particular size. Rasterization and hinting is done with the FreeType library.

Common formats under Linux are PostScript Type 1 and Type 2, TrueType, and OpenType.

Bitmap or raster fonts

Consists of an array of pixels designed for a specific font size. Bitmap fonts are extremely fast and simple to render. However, compared to vector fonts, bitmap fonts cannot be scaled without losing quality. As such, these fonts are usually distributed in different sizes. These days, bitmap fonts are still used in the Linux console and sometimes in terminals.

Under Linux, Portable Compiled Format (PCF) or Glyph Bitmap Distribution Format (BDF) are the most common formats.

The appearance of these fonts can be influenced by two main aspects:

- choosing a suitable font family,
- rendering the font with an algorithm that achieves results comfortable for the receiver's eyes.

The last point is only relevant to vector fonts. Although the above two points are highly subjective, some defaults need to be created.

Linux font rendering systems consist of several libraries with different relations. The basic font rendering library is FreeType (https://www.freetype.org/), which converts font glyphs of supported formats into optimized bitmap glyphs. The rendering process is controlled by an algorithm and its parameters (which may be subject to patent issues).

Every program or library which uses FreeType should consult the Fontconfig (https://www.fontconfig.org/) library. This library gathers font configuration from users and from the system. When a user amends their Fontconfig setting, this change will result in Fontconfig-aware applications.

More sophisticated OpenType shaping needed for scripts such as Arabic, Han or Phags-Pa and other higher level text processing is done using Harfbuzz (https://harfbuzz.github.io/) or Pango (https://www.pango.org/).

25.2.1 Showing installed fonts

To get an overview about which fonts are installed on your system, ask the commands **rpm** or **fc-list**. Both will give you a good answer, but may return a different list depending on system and user configuration:

rpm

Invoke **rpm** to see which software packages containing fonts are installed on your system:

```
> rpm -qa '*fonts*'
```

Every font package should satisfy this expression. However, the command may return some false positives like `fonts-config` (which is neither a font nor does it contain fonts).

fc-list

Invoke **fc-list** to get an overview about what font families can be accessed, whether they are installed on the system or in your home:

```
> fc-list ':' family
```

Note: Command fc-list

The command **fc-list** is a wrapper to the Fontconfig library. It is possible to query a lot of interesting information from Fontconfig—or, to be more precise, from its cache. See **man 1 fc-list** for more details.

25.2.2 Viewing fonts

If you want to know what an installed font family looks like, either use the command **ftview** (package `ft2demos`) or visit https://fontinfo.opensuse.org/. For example, to display the FreeMono font in 14 point, use **ftview** like this:

```
> ftview 14 /usr/share/fonts/truetype/FreeMono.ttf
```

If you need further information, go to https://fontinfo.opensuse.org/ to find out which styles (regular, bold, italic, etc.) and languages are supported.

25.2.3 Querying fonts

To query which font is used when a pattern is given, use the **fc-match** command.

For example, if your pattern contains an already installed font, **fc-match** returns the file name, font family, and the style:

```
> fc-match 'Liberation Serif'
LiberationSerif-Regular.ttf: "Liberation Serif" "Regular"
```

If the desired font does not exist on your system, Fontconfig's matching rules take place and try to find the most similar fonts available. This means, your request is substituted:

```
> fc-match 'Foo Family'
DejaVuSans.ttf: "DejaVu Sans" "Book"
```

Fontconfig supports *aliases*: a name is substituted with another family name. A typical example are the generic names such as “sans-serif”, “serif”, and “monospace”. These alias names can be substituted by real family names or even a preference list of family names:

```
> for font in serif sans mono; do fc-match "$font" ; done
DejaVuSerif.ttf: "DejaVu Serif" "Book"
DejaVuSans.ttf: "DejaVu Sans" "Book"
DejaVuSansMono.ttf: "DejaVu Sans Mono" "Book"
```

The result may vary on your system, depending on which fonts are currently installed.

Note: Similarity rules according to fontconfig

Fontconfig *always* returns a real family (if at least one is installed) according to the given request, as similar as possible. “Similarity” depends on Fontconfig's internal metrics and on the user's or administrator's Fontconfig settings.

25.2.4 Installing fonts

To install a new font there are these major methods:

1. Manually install the font files such as `*.ttf` or `*.otf` to a known font directory. If it needs to be system-wide, use the standard directory `/usr/share/fonts`. For installation in your home directory, use `~/.config/fonts`.
   If you want to deviate from the standard directories, Fontconfig allows you to choose another one. Let Fontconfig know by using the `<dir>` element, see *Section 25.2.5.2, "Diving into fontconfig XML"* for details.

2. Install fonts using **`zypper`**. Lots of fonts are already available as a package, be it on your SUSE distribution or in the M17N:fonts (https://download.opensuse.org/repositories/M17N:/fonts/) repository. Add the repository to your list using the following command. For example, to add a repository for SUSE Linux Enterprise Desktop 15 SP7:

   ```
   > sudo zypper ar
       https://download.opensuse.org/repositories/M17N:/fonts/SLE_15/
   ```

   To search for your *`FONT_FAMILY_NAME`* use this command:

   ```
   > zypper se 'FONT_FAMILY_NAME*fonts'
   ```

25.2.5 Configuring the appearance of fonts

Depending on the rendering medium, and font size, the result may be unsatisfactory. For example, an average monitor these days has a resolution of 100dpi which makes pixels too big and glyphs look clunky.

There are several algorithms available to deal with low resolutions, such as anti-aliasing (grayscale smoothing), hinting (fitting to the grid), or subpixel rendering (tripling resolution in one direction). These algorithms can also differ from one font format to another.

Via Fontconfig, it is possible to select a rendering algorithms for every font individually or for a set of fonts.

25.2.5.1 Configuring fonts via `sysconfig`

SUSE Linux Enterprise Desktop comes with a `sysconfig` layer above Fontconfig. This is a good starting point for experimenting with font configuration. To change the default settings, edit the configuration file `/etc/sysconfig/fonts-config`. (or use the YaST sysconfig module). After you have edited the file, run **`fonts-config`**:

```
> sudo /usr/sbin/fonts-config
```

Restart the application to make the effect visible. Keep in mind the following issues:

- A few applications do need not to be restarted. For example, Firefox re-reads Fontconfig configuration from time to time. Newly created or reloaded tabs get new font configurations later.
- The **`fonts-config`** script is called automatically after every package installation or removal (if not, it is a bug of the font software package).
- Every sysconfig variable can be temporarily overridden by the **`fonts-config`** command line option. See **`fonts-config --help`** for details.

There are several sysconfig variables which can be altered. See **`man 1 fonts-config`** or the help page of the YaST sysconfig module. The following variables are examples:

Usage of rendering algorithms

Consider `FORCE_HINTSTYLE`, `FORCE_AUTOHINT`, `FORCE_BW`, `FORCE_BW_MONOSPACE`, `USE_EMBEDDED_BITMAPS` and `EMBEDDED_BITMAP_LANGAGES`

Preference lists of generic aliases

Use `PREFER_SANS_FAMILIES`, `PREFER_SERIF_FAMILIES`, `PREFER_MONO_FAMILIES` and `SEARCH_METRIC_COMPATIBLE`

The following list provides some configuration examples, sorted from the “most readable” fonts (more contrast) to “most beautiful” (more smoothed).

Bitmap fonts

Prefer bitmap fonts via the `PREFER_*_FAMILIES` variables. Follow the example in the help section for these variables. Be aware that these fonts are rendered black and white, not smoothed and that bitmap fonts are available in several sizes only. Consider using

```
SEARCH_METRIC_COMPATIBLE="no"
```

to disable metric compatibility-driven family name substitutions.

Scalable fonts rendered black and white

Scalable fonts rendered without antialiasing can result in a similar outcome to bitmap fonts, while maintaining font scalability. Use well hinted fonts like the Liberation families. Unfortunately, there is a lack of well hinted fonts though. Set the following variable to force this method:

```
FORCE_BW="yes"
```

Monospaced fonts rendered black and white

Render monospaced fonts without antialiasing only, otherwise use default settings:

```
FORCE_BW_MONOSPACE="yes"
```

Default settings

All fonts are rendered with antialiasing. Well hinted fonts will be rendered with the *byte code interpreter* (BCI) and the rest with autohinter (`hintstyle=hintslight`). Leave all relevant sysconfig variables to the default setting.

CFF fonts

Use fonts in CFF format. They can be considered also more readable than the default TrueType fonts given the current improvements in FreeType2. Try them out by following the example of `PREFER_*_FAMILIES`. Possibly make them more dark and bold with:

```
SEARCH_METRIC_COMPATIBLE="no"
```

as they are rendered by `hintstyle=hintslight` by default. Also consider using:

```
SEARCH_METRIC_COMPATIBLE="no"
```

Autohinter exclusively

Even for a well hinted font, use FreeType2's autohinter. That can lead to thicker, sometimes fuzzier letter shapes with lower contrast. Set the following variable to activate this:

```
FORCE_AUTOHINTER="yes"
```

Use `FORCE_HINTSTYLE` to control the level of hinting.

25.2.5.2 Diving into fontconfig XML

Fontconfig's configuration format is the *eXtensible Markup Language* (XML). These few examples are not a complete reference, but a brief overview. Details and other inspiration can be found in **`man 5 fonts-conf`** or in `/etc/fonts/conf.d/`.

The central Fontconfig configuration file is `/etc/fonts/fonts.conf`, which—along other work—includes the whole `/etc/fonts/conf.d/` directory. To customize Fontconfig, there are two places where you can insert your changes:

FONTCONFIG CONFIGURATION FILES

1. **System-wide changes.** Edit the file `/etc/fonts/local.conf` (by default, it contains an empty `fontconfig` element).

2. **User-specific changes.** Edit the file `~/.config/fontconfig/fonts.conf`. Place Fontconfig configuration files in the `~/.config/fontconfig/conf.d/` directory.

User-specific changes overwrite any system-wide settings.

Note: Deprecated user configuration file

The file `~/.fonts.conf` is marked as deprecated and should not be used anymore. Use `~/.config/fontconfig/fonts.conf` instead.

Every configuration file needs to have a `fontconfig` element. As such, the minimal file looks like this:

```
<?xml version="1.0"?>
   <!DOCTYPE fontconfig SYSTEM "fonts.dtd">
   <fontconfig>
   <!-- Insert your changes here -->
   </fontconfig>
```

If the default directories are not enough, insert the `dir` element with the respective directory:

```
<dir>/usr/share/fonts2</dir>
```

Fontconfig searches *recursively* for fonts.

Font-rendering algorithms can be chosen with following Fontconfig snippet (see *Example 25.1, “Specifying rendering algorithms”*):

EXAMPLE 25.1: SPECIFYING RENDERING ALGORITHMS

```
<match target="font">
 <test name="family">
  <string>FAMILY_NAME</string>
 </test>
```

```
 <edit name="antialias" mode="assign">
  <bool>true</bool>
 </edit>
 <edit name="hinting" mode="assign">
  <bool>true</bool>
 </edit>
 <edit name="autohint" mode="assign">
  <bool>false</bool>
 </edit>
 <edit name="hintstyle" mode="assign">
  <const>hintfull</const>
 </edit>
</match>
```

Various properties of fonts can be tested. For example, the `<test>` element can test for the font family (as shown in the example), size interval, spacing, font format, and others. When abandoning `<test>` completely, all `<edit>` elements will be applied to every font (global change).

EXAMPLE 25.2: ALIASES AND FAMILY NAME SUBSTITUTIONS

Rule 1

```
<alias>
 <family>Alegreya SC</family>
 <default>
  <family>serif</family>
 </default>
</alias>
```

Rule 2

```
<alias>
 <family>serif</family>
 <prefer>
  <family>Droid Serif</family>
 </prefer>
</alias>
```

Rule 3

```
<alias>
 <family>serif</family>
 <accept>
  <family>STIXGeneral</family>
 </accept>
</alias>
```

The rules from *Example 25.2, "Aliases and family name substitutions"* create a ***prioritized family list*** (PFL). Depending on the element, different actions are performed:

`<default>` from *Rule 1*

This rule adds a `serif` family name *at the end* of the PFL.

`<prefer>` from *Rule 2*

This rule adds "Droid Serif" *just before* the first occurrence of `serif` in the PFL, whenever `Alegreya SC` is in PFL.

`<accept>` from *Rule 3*

This rule adds a "STIXGeneral" family name *just after* the first occurrence of the `serif` family name in the PFL.

Putting this together, when snippets occur in the order *Rule 1* - *Rule 2* - *Rule 3* and the user requests "Alegreya SC", then the PFL is created as depicted in *Table 25.1, "Generating PFL from fontconfig rules"*.

TABLE 25.1: GENERATING PFL FROM FONTCONFIG RULES

Order	Current PFL
Request	`Alegreya SC`
Rule 1	`Alegreya SC`, `serif`
Rule 2	`Alegreya SC`, `Droid Serif`, `serif`
Rule 3	`Alegreya SC`, `Droid Serif`, `serif`, `STIXGeneral`

In Fontconfig's metrics, the family name has the highest priority over other patterns, like style, size, etc. Fontconfig checks which family is currently installed on the system. If "Alegreya SC" is installed, then Fontconfig returns it. If not, it asks for "Droid Serif", etc.

Be careful. When the order of Fontconfig snippets is changed, Fontconfig can return different results, as depicted in *Table 25.2, "Results from generating PFL from fontconfig rules with changed order"*.

TABLE 25.2: RESULTS FROM GENERATING PFL FROM FONTCONFIG RULES WITH CHANGED ORDER

Order	Current PFL	Note
Request	`Alegreya SC`	Same request performed.

Order	Current PFL	Note
Rule 2	`Alegreya SC`	`serif` not in PFL, nothing is substituted
Rule 3	`Alegreya SC`	`serif` not in PFL, nothing is substituted
Rule 1	`Alegreya SC`, `serif`	`Alegreya SC` present in PFL, substitution is performed

Note: Implication

Think of the `<default>` alias as a classification or inclusion of this group (if not installed). As the example shows, `<default>` should always precede the `<prefer>` and `<accept>` aliases of that group.

`<default>` classification is not limited to the generic aliases serif, sans-serif and monospace. See `/usr/share/fontconfig/conf.avail/30-metric-aliases.conf` for a complex example.

The following Fontconfig snippet in *Example 25.3, "Aliases and family name substitutions"* creates a `serif` group. Every family in this group could substitute others when a former font is not installed.

EXAMPLE 25.3: ALIASES AND FAMILY NAME SUBSTITUTIONS

```
<alias>
 <family>Alegreya SC</family>
 <default>
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>Droid Serif</family>
 <default>
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>STIXGeneral</family>
 <default>
```

```
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>serif</family>
 <accept>
  <family>Droid Serif</family>
  <family>STIXGeneral</family>
  <family>Alegreya SC</family>
 </accept>
</alias>
```

Priority is given by the order in the `<accept>` alias. Similarly, stronger `<prefer>` aliases can be used.

Example 25.2, "Aliases and family name substitutions" is expanded by *Example 25.4, "Aliases and family names substitutions"*.

EXAMPLE 25.4: ALIASES AND FAMILY NAMES SUBSTITUTIONS

Rule 4

```
<alias>
 <family>serif</family>
 <accept>
  <family>Liberation Serif</family>
 </accept>
</alias>
```

Rule 5

```
<alias>
 <family>serif</family>
 <prefer>
  <family>DejaVu Serif</family>
 </prefer>
</alias>
```

The expanded configuration from *Example 25.4, "Aliases and family names substitutions"* would lead to the following PFL evolution:

TABLE 25.3: RESULTS FROM GENERATING PFL FROM FONTCONFIG RULES

Order	Current PFL
Request	`Alegreya SC`

Order	Current PFL
Rule 1	Alegreya SC, serif
Rule 2	Alegreya SC, Droid Serif, serif
Rule 3	Alegreya SC, Droid Serif, serif, STIXGeneral
Rule 4	Alegreya SC, Droid Serif, serif, Liberation Serif, STIXGeneral
Rule 5	Alegreya SC, Droid Serif, DejaVu Serif, serif, Liberation Serif, STIXGeneral

Note: Implications.

- In case multiple <accept> declarations for the same generic name exist, the declaration that is parsed last “wins”. If possible, do not use <accept> **after** user (/etc/fonts/conf.d/*-user.conf) when creating a system-wide configuration.
- In case multiple <prefer declarations for the same generic name exist, the declaration that is parsed last “wins”. If possible, do not use <prefer> **before** user in the system-wide configuration.
- Every <prefer> declaration overwrites <accept> declarations for the same generic name. If the administrator wants to allow the user to use <accept> and not only <prefer>, the administrator should not use <prefer> in the system-wide configuration. On the other hand, as users mostly use <prefer>, this should not have any detrimental effect. We also see the use of <prefer> in system-wide configurations.

25.3 GNOME configuration for administrators

25.3.1 The dconf system

Configuration of the GNOME desktop is managed with `dconf`. It is a hierarchically structured database or registry that allows users to modify their personal settings, and system administrators to set default or mandatory values for all users. `dconf` replaces the `gconf` system of GNOME 2.

Use **`dconf-editor`** to view the `dconf` options with a graphical user interface. Use **`dconf`** to access and modify configuration options with the command line.

The GNOME `Tweaks` tool provides an easy-to-use user interface for additional configuration options beyond the normal GNOME configuration. The tool can be started from the GNOME application menu or from the command line with **`gnome-tweak-tool`**.

25.3.2 System-wide configuration

Global `dconf` configuration parameters can be set in the `/etc/dconf/db/` directory. This includes the configuration for GDM or locking certain configuration options for users.

Use the following procedure as an example to create a system-wide configuration:

1. Create a new directory that ends with a `.d` in `/etc/dconf/db/`. This directory can contain an arbitrary amount of text files with configuration options. For this example, create the file `/etc/dconf/db/network.d/00-proxy` with the following content:

```
# This is a comment
[system/proxy/http]
host='10.0.0.1'
enabled=true
```

2. Parse the new configuration directives into the dconf database format:

```
> sudo dconf update
```

3. Add the new `network` configuration database to the default user profile, by creating the file `/etc/dconf/profile/user`. Then add the following content:

```
system-db:network
```

The file `/etc/dconf/profile/user` is a GNOME default. Other profiles can be defined in the environment variable `DCONF_PROFILE`.

4. Optional: to lock the proxy configuration for users, create the file `/etc/dconf/db/network/locks/proxy`. Then add a line to this file with the keys that may not be changed:

```
/system/proxy/http/host
/system/proxy/http/enabled
```

You can use the graphical **`dconf-editor`** to create a profile with one user and then use **`dconf dump /`** to list all configuration options. The configuration options can then be stored in a global profile.

A detailed description of the global configuration is available at https://wiki.gnome.org/Projects/dconf/SystemAdministrators.

25.3.3 Managing tiling windows on Wayland

If you are running the GNOME desktop on Wayland and want to have your windows arranged next to each other, we recommend using the *gTile* GNOME extension. gTile lets you re-arrange windows based on a configurable grid scheme. For example, you can launch six terminals and have them arranged in 3 rows and 2 columns so that they use the whole desktop area.

25.3.3.1 Installing gTile

Important

To use the gTile GNOME extension, you need to be running the GNOME desktop and have the `chrome-gnome-shell` (for SLED 15 SP4 and older) or `gnome-browser-connector` (for SLED 15 SP5 and newer) package installed.

1. Point your browser to the gTile extension web page https://extensions.gnome.org/extension/28/gtile/. If you see a warning about a missing browser extension, click the link and follow your Web browser's instruction on how to install and enable the GNOME Shell extension.

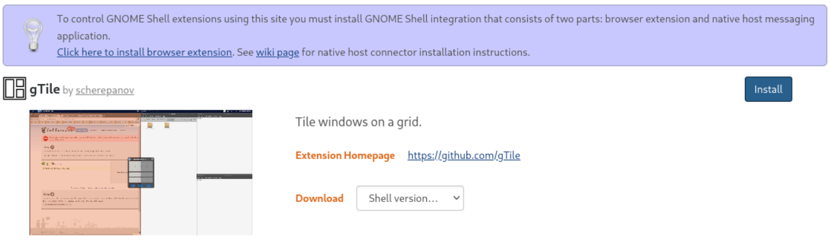


FIGURE 25.1: WARNING ABOUT A MISSING GNOME SHELL EXTENSION

2. After the GNOME Shell extension is installed in your Web browser, you can see an *OFF* button on the top right of the gTile Web page. To activate the gTile extension, switch it to *ON* by clicking it.

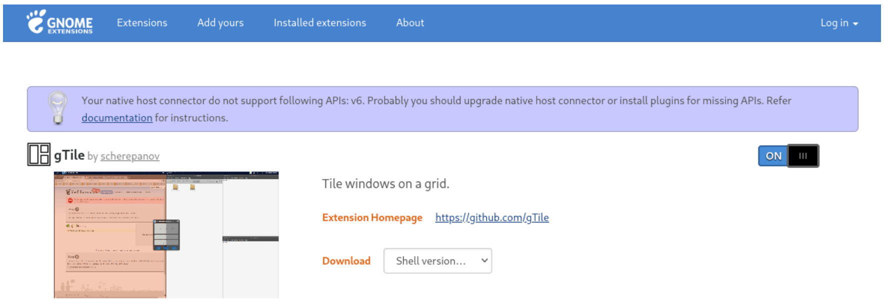


FIGURE 25.2: GNOME GTILE EXTENSION ACTIVATED

3. A gTile tray icon appears in the GNOME tray in the right side of the GNOME panel. You can use it to activate gTile window layouts.

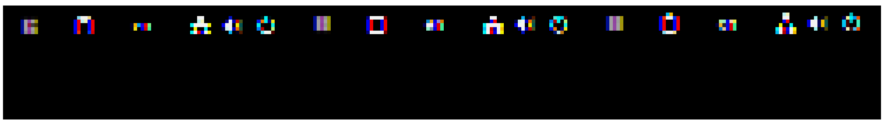


FIGURE 25.3: GTILE TRAY ICON

25.3.3.2 Working with the gTile extension

PROCEDURE 25.1: ARRANGING THE WINDOWS

1. To open gTile for the currently focused window, press `Meta`–`Enter`.

2. To set the number of columns in which the windows will be arranged, press a number key.

3. To automatically tile the windows according to the *Autotile Main window sizes* setting, press the `M` key. By default, the first time you press the `M` key, it will make the current window use half the screen, and all other windows to be tiled in a column on the other half of the screen. You can press `M` repeatedly, to cycle between variants. This is helpful to keep a big main window, and several smaller secondary windows.

4. To move the current window to within the predefined tiling of the screen, use arrow keys. To shrink or grow the number of tiles that the window occupies, use `Shift` and an arrow key.

5. To snap a window to the edges of the neighboring windows and the desktop, press `Ctrl`–`Alt`–`S` .

6. To close the gTile window, press `Esc` .

PROCEDURE 25.2: ADVANCED CONFIGURATION

1. To open the gTile settings, run the following command: **`gnome-extensions-app`**.

2. Select gTile and click *Settings*.

3. Define a preset and a key combination to easily switch between frequently used window layouts.
   For more details on how to configure the gTile extension, refer to https://github.com/gTile/gTile.

25.3.4 More information

For more information, see https://help.gnome.org/admin/.

25.4 Switching between Intel and NVIDIA Optimus GPUs with SUSE Prime

SUSE Prime is a tool for switching between onboard Intel graphical processing units (GPUs) and NVIDIA GPUs equipped with NVIDIA's switchable graphics Optimus technology. Optimus provides a mechanism for easily switching between an onboard Intel GPU and a discrete NVIDIA GPU. This is designed for running a laptop in a power-saving mode or at maximum performance: use the Intel GPU to save power, and the NVIDIA GPU for 3D applications.

SUSE Prime is included in the SUSE Linux Enterprise Workstation Extension for SUSE Linux Enterprise 15 SP7.

SUSE Prime works only on systems running X11, not Wayland. If your system runs Wayland, you must disable it and fall back to X11 to use SUSE Prime (see *Section 25.4.1, "Prerequisites"*).

25.4.1 Prerequisites

You must have a configured and working NVIDIA Optimus GPU using the NVIDIA drivers included in SUSE Linux Enterprise 15 SP7 (see *Section 25.4.3, "Installing NVIDIA drivers"*), and an onboard Intel GPU. Bumblebee, the older switching tool for NVIDIA Optimus, must not be installed.

There must not be a `/etc/X11/xorg.conf` file, and no configuration files with active `ServerLayout`, `Device`, or `Screen` sections in the `/etc/X11/xorg.conf.d` directory.

SUSE Prime works only with X11. Use the **`loginctl`** command to see if your system is using X11 or Wayland:

```
> loginctl
   SESSION        UID USER             SEAT            TTY
         2       1000 tux             seat0
> loginctl show-session 2|grep Type
Type=x11
```

If your system uses Wayland, disable it by editing `/etc/gdm/custom.conf` and un-commenting `WaylandEnable=false`. Then reboot.

25.4.2 Installing and using SUSE Prime

Your NVIDIA graphics card should already be installed and working. If it is not, see *Section 25.4.3, "Installing NVIDIA drivers"*.

Install the `suse-prime` package:

```
> sudo zypper install suse-prime
```

To switch your GPU run one of the following commands, then log out and log back in:

```
> sudo prime-select intel
> sudo prime-select intel2
> sudo prime-select nvidia
```

Use the **intel** driver when it is the modesetting driver. **intel2** is for systems that use the `xf86-video-intel` driver. You can get this information by installing and running `inxi`:

```
> inxi -G
Graphics: Device-1: Intel Xeon E3-1200 v3/4th Gen Core Processor Integrated Graphics
 Controller
          Display Server: x11(X.org 1.20.1 ) drivers: modesetting (unloaded: fbdev, vesa)
          Resolution: 1920x1080@60.00hz
          OpenGL: renderer: Mesa DRI Intel Haswell Desktop version: 4.5 Mesa 18.2.8
```

Which GPU is currently active?

```
> sudo /usr/sbin/prime-select get-current
Driver configured: intel
```

25.4.3 Installing NVIDIA drivers

If you need to identify your NVIDIA card so you know which driver to use, run the following command:

```
>  /sbin/lspci | grep -i nvidia
```

Follow these steps to install the drivers with Zypper.

List the available driver packages:

```
> sudo zypper se nvidia
```

Then install the drivers for your NVIDIA graphics card:

```
> sudo zypper se packagename
```

26 Accessing file systems with FUSE

Revision History

2024-06-21

FUSE is the acronym for *Filesystem in Userspace*. This means you can configure and mount a file system as an unprivileged user. Normally, you need to be `root` for this task. FUSE alone is a kernel module. Combined with plug-ins, it allows you to extend FUSE to access almost all file systems like remote SSH connections, ISO images, and more.

26.1 Configuring FUSE

Before you can use FUSE, you need to install the package `fuse`. Depending which file system you want to use, you need additional plug-ins available as separate packages.

Generally you do not need to configure FUSE. However, it is a good idea to create a directory where all your mount points are combined. For example, you can create a directory `~/mounts` and insert your subdirectories for your different file systems there.

26.2 Mounting an NTFS partition

NTFS, the *New Technology File System*, is the default file system of Windows. Since under normal circumstances the unprivileged user cannot mount NTFS block devices using the external FUSE library, the process of mounting a Windows partition described below requires root privileges. Mounting NTFS partitions is supported on SUSE Linux Enterprise Server and SUSE Linux Enterprise Desktop with SUSE Linux Enterprise Workstation Extension only.

1. Become `root` and install the package `ntfs-3g`. It is available in SUSE Linux Enterprise Workstation Extension.
2. Create a directory that is to be used as a mount point, for example, `~/mounts/windows`.
3. Find out which Windows partition you need. Use YaST and start the partitioner module to see which partition belongs to Windows, but do not modify anything. Alternatively, become `root` and execute **`/sbin/fdisk`** `-l`. Look for partitions with a partition type of `HPFS/NTFS`.

4. Mount the partition in read-write mode. Replace the placeholder *DEVICE* with your respective Windows partition:

```
> ntfs-3g /dev/DEVICE MOUNT POINT
```

To use your Windows partition in read-only mode, append `-o ro`:

```
> ntfs-3g /dev/DEVICE MOUNT POINT -o ro
```

The command **`ntfs-3g`** uses the current user (UID) and group (GID) to mount the given device. To set the write permissions to a different user, use the command **`id`** `USER` to get the output of the UID and GID values. Set it with:

```
# id tux
uid=1000(tux) gid=100(users) groups=100(users),16(dialout),33(video)
ntfs-3g /dev/DEVICE MOUNT POINT -o uid=1000,gid=100
```

Find additional options in the man page.

To unmount the resource, run **`fusermount -u`** *`MOUNT POINT`*.

26.3 More information

For more information, see the home page of FUSE at https://github.com/libfuse/libfuse.

27 Installing multiple kernel versions

Revision History

2024-05-13

SUSE Linux Enterprise Desktop supports the parallel installation of multiple kernel versions. When installing a second kernel, a boot entry and an initrd are automatically created, so no further manual configuration is needed. When rebooting the machine, the newly added kernel is available as an additional boot parameter.

Using this functionality, you can safely test kernel updates while being able to always fall back to the proven former kernel. To do this, do not use the update tools (such as the YaST Online Update or the updater applet), but instead follow the process described in this chapter.

Warning: Support entitlement

Be aware that you lose your entire support entitlement for the machine when installing a self-compiled or a third-party kernel. Only kernels shipped with SUSE Linux Enterprise Desktop and kernels delivered via the official update channels for SUSE Linux Enterprise Desktop are supported.

Tip: Check your boot loader configuration kernel

It is recommended to check your boot loader configuration after having installed another kernel to set the default boot entry of your choice. See *Section 18.3, "Configuring the boot loader with YaST"* for more information.

27.1 Enabling and configuring multiversion support

Installing multiple versions of a software package (multiversion support) is enabled by default with SUSE Linux Enterprise Server 12 and newer versions. To verify this setting, proceed as follows:

1. Open `/etc/zypp/zypp.conf` with the editor of your choice as `root`.

2. Search for the string `multiversion`. If multiversion is enabled for all kernel packages capable of this feature, the following line appears uncommented:

```
multiversion = provides:multiversion(kernel)
```

3. To restrict multi-version support to certain kernel flavors, add the package names as a comma-separated list to the `multiversion` option in `/etc/zypp/zypp.conf`, for example,

```
multiversion = kernel-default,kernel-default-base,kernel-source
```

4. Save your changes.

Warning: Kernel Module Packages (KMP)

Make sure that required vendor-provided kernel modules (Kernel Module Packages) are also installed for the new updated kernel. The kernel update process will not warn about eventually missing kernel modules because package requirements are still fulfilled by the old kernel that is kept on the system.

27.1.1 Automatically deleting unused kernels

When frequently testing new kernels with multiversion support enabled, the boot menu quickly becomes confusing. Since a `/boot` partition normally has limited space, you may run into trouble with `/boot` overflowing. While you can delete unused kernel versions manually with YaST or Zypper (as described below), you can also configure `libzypp` to automatically delete kernels no longer used. By default no kernels are deleted.

1. Open `/etc/zypp/zypp.conf` with the editor of your choice as `root`.

2. Search for the string `multiversion.kernels` and activate this option by uncommenting the line. This option takes a comma-separated list of the following values:

   `5.3.18-53.3`: keep the kernel with the specified version number

   `latest`: keep the kernel with the highest version number

   `latest-N`: keep the kernel with the Nth highest version number

   `running`: keep the running kernel

`oldest`: keep the kernel with the lowest version number (the one that was originally shipped with SUSE Linux Enterprise Desktop)

`oldest+N`. keep the kernel with the Nth lowest version number

Here are several examples

`multiversion.kernels = latest,running`
: Keep the latest kernel and the one currently running. This is similar to not enabling the multiversion feature, except that the old kernel is removed *after the next reboot* and not immediately after the installation.

`multiversion.kernels = latest,latest-1,running`
: Keep the last two kernels and the one currently running.

`multiversion.kernels = latest,running,5.3.18-53.3`
: Keep the latest kernel, the one currently running, and `5.3.18-53.3`.

Tip: Keep the running kernel

Unless you are using a special setup, always keep the kernel marked `running`.

If you do not keep the running kernel, it will be deleted when updating the kernel. In turn, this means that all the running kernel's modules are also deleted and cannot be loaded anymore.

If you decide not to keep the running kernel, always reboot immediately after a kernel upgrade to avoid issues with modules.

27.1.2 Use case: deleting an old kernel after reboot only

You want to make sure that an old kernel will only be deleted after the system has rebooted successfully with the new kernel.

Change the following line in `/etc/zypp/zypp.conf`:

```
multiversion.kernels = latest,running
```

The previous parameters tell the system to keep the latest kernel and the running one only if they differ.

27.1.3 Use case: keeping older kernels as fallback

You want to keep one or more kernel versions to have one or more “spare” kernels.

This can be useful if you need kernels for testing. If something goes wrong (for example, your machine does not boot), you still can use one or more kernel versions which are known to be good.

Change the following line in `/etc/zypp/zypp.conf`:

```
multiversion.kernels = latest,latest-1,latest-2,running
```

When you reboot your system after the installation of a new kernel, the system will keep three kernels: the current kernel (configured as `latest,running`) and its two immediate predecessors (configured as `latest-1` and `latest-2`).

27.1.4 Use case: keeping a specific kernel version

You make regular system updates and install new kernel versions. However, you are also compiling your own kernel version and want to make sure that the system will keep them.

Change the following line in `/etc/zypp/zypp.conf`:

```
multiversion.kernels = latest,5.3.18-53.3,running
```

When you reboot your system after the installation of a new kernel, the system will keep two kernels: the new and running kernel (configured as `latest,running`) and your self-compiled kernel (configured as `5.3.18-53.3`).

27.2 Installing/removing multiple kernel versions with YaST

You can install or remove multiple kernels with YaST:

1. Start YaST and open the software manager via *Software* › *Software Management*.
2. List all packages capable of providing multiple versions by choosing *View* › *Package Classification* › *Multiversion Packages*.

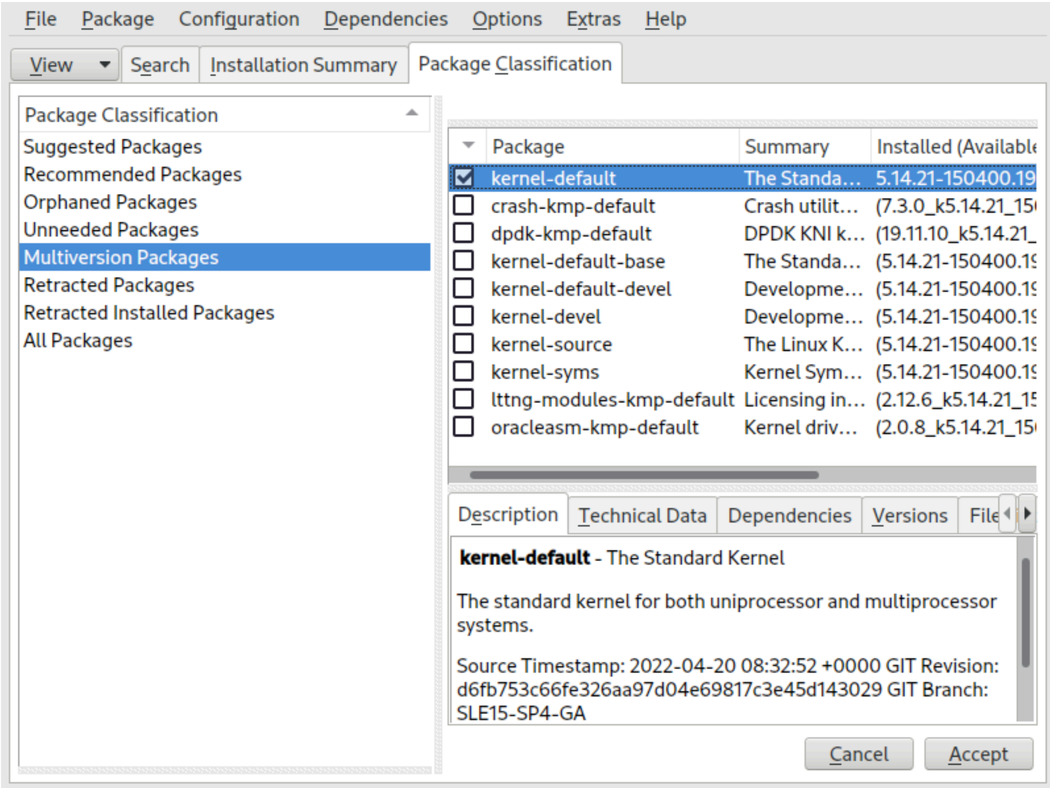


FIGURE 27.1: THE YAST SOFTWARE MANAGER: MULTIVERSION VIEW

3. Select a package and open its *Version* tab in the bottom pane on the left.

4. To install a package, click the check box next to it. A green check mark indicates it is selected for installation.

   To remove an already installed package (marked with a white check mark), click the check box next to it until a red X indicates it is selected for removal.

5. Click *Accept* to start the installation.

27.3 Installing/removing multiple kernel versions with Zypper

You can install or remove multiple kernels with **zypper**:

1. Use the command **zypper se -s 'kernel*'** to display a list of all kernel packages available:

```
S  | Name                      | Type      | Version                  | Arch      | Repository
```

```
---+---------------------+---------+------------------+--------
+----------------------------------------------------
i+ | kernel-default                        | package | 6.4.0-150600.9.2             | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-default-base                   | package | 6.4.0-150600.9.2.150600.10.40 | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-default-devel                  | package | 6.4.0-150600.9.2             | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-devel                          | package | 6.4.0-150600.9.2             | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-all                   | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-amdgpu                | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ath10k                | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ath11k                | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ath12k                | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-atheros               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-bluetooth             | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-bnx2                  | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-brcm                  | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-chelsio               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-dpaa2                 | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-i915                  | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-intel                 | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-iwlwifi               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-liquidio              | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-marvell               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-media                 | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-mediatek              | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-mellanox              | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-mwifiex               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-network               | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-nfp                   | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-nvidia                | package | 20240201-150600.1.1          | noarch | SLE-
Module-Basesystem15-SP6-Pool
```

```
   | kernel-firmware-nvidia-gsp-G06         | package | 525.116.04-150500.1.1         | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-firmware-nvidia-gspx-G06        | package | 550.54.14-150600.1.1          | x86_64 | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-platform               | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-prestera               | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-qcom                   | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-qlogic                 | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-radeon                 | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-realtek                | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-serial                 | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-sound                  | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ti                     | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-ueagle                 | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
i  | kernel-firmware-usb-network            | package | 20240201-150600.1.1           | noarch | SLE-
Module-Basesystem15-SP6-Pool
   | kernel-macros                          | package | 6.4.0-150600.9.2              | noarch | SLE-
Module-Basesystem15-SP6-Pool
```

2. Specify the exact version when installing:

```
> sudo zypper in kernel-default-6.4.0-150600.9.2
```

3. When uninstalling a kernel, use the commands `zypper se -si 'kernel*'` to list all kernels installed and `zypper rm` *`PACKAGENAME-VERSION`* to remove the package.

28 Managing kernel modules

Revision History

2024-03-14

Although Linux is a monolithic kernel, it can be extended using kernel modules. These are special objects that can be inserted into the kernel and removed on demand. In practical terms, kernel modules make it possible to add and remove drivers and interfaces that are not included in the kernel itself. Linux provides several commands for managing kernel modules.

28.1 Listing loaded modules with lsmod and modinfo

Use the **`lsmod`** command to view what kernel modules are currently loaded. The output of the command may look as follows:

```
> lsmod
Module                  Size  Used by
snd_usb_audio         188416  2
snd_usbmidi_lib        36864  1 snd_usb_audio
hid_plantronics        16384  0
snd_rawmidi            36864  1 snd_usbmidi_lib
snd_seq_device         16384  1 snd_rawmidi
fuse                  106496  3
nfsv3                  45056  1
nfs_acl                16384  1 nfsv3
```

The output is divided into three columns. The `Module` column lists the names of the loaded modules, while the `Size` column displays the size of each module. The `Used by` column shows the number of referring modules and their names. This list may be incomplete.

To view detailed information about a specific kernel module, use the **`modinfo`** *`MODULE_NAME`* command, where *`MODULE_NAME`* is the name of the desired kernel module. The **`modinfo`** binary resides in the `/sbin` directory that is not in the user's PATH environment variable. This means that you must specify the full path to the binary when running **`modinfo`** command as a regular user:

```
> /sbin/modinfo kvm
filename:       /lib/modules/6.4.0-150700.38-default/kernel/arch/x86/kvm/kvm.ko.zst
license:        GPL
author:         Qumranet
suserelease:    SLE15-SP7
```

```
srcversion:     9DACE73AC65F98D556DAD60
depends:        irqbypass
supported:      yes
retpoline:      Y
intree:         Y
name:           kvm
vermagic:       6.4.0-150700.38-default SMP mod_unload modversions
```

28.2 Adding and removing kernel modules

While it is possible to use `insmod` and `rmmod` to add and remove kernel modules, it is recommended to use the `modprobe` tool instead. `modprobe` offers several important advantages, including automatic dependency resolution and blacklisting.

When used without any parameters, the `modprobe` command installs a specified kernel module. `modprobe` must be run with root privileges:

```
> sudo modprobe acpi
```

To remove a kernel module, use the `-r` parameter:

```
> sudo modprobe -r acpi
```

28.2.1 Loading kernel modules automatically on boot

Instead of loading kernel modules manually, you can load them automatically during the boot process using the `systemd-modules-load.service` service. To enable a kernel module, add a `.conf` file to the `/etc/modules-load.d/` directory. It is good practice to give the configuration file the same name as the module, for example:

```
/etc/modules-load.d/rt2800usb.conf
```

The configuration file must contain the name of the desired kernel module (for example, `rt2800usb`).

The described technique allows you to load kernel modules without any parameters. If you need to load a kernel module with specific options, add a configuration file to the `/etc/modprobe.d/` directory instead. The file must have the `.conf` extension. The name of the file should adhere to the following naming convention: `priority-modulename.conf`, for example: `50-`

thinkfan.conf. The configuration file must contain the name of the kernel module and the desired parameters. You can use the example command below to create a configuration file containing the name of the kernel module and its parameters:

```
> echo "options thinkpad_acpi fan_control=1" | sudo tee /etc/modprobe.d/thinkfan.conf
```

Note: Loading kernel modules

Most kernel modules are loaded by the system automatically when a device is detected or user space requests specific functionality. Thus, adding modules manually to `/etc/modules-load.d/` is rarely required.

28.2.2 Blacklisting kernel modules with modprobe

Blacklisting a kernel module prevents it from loading during the boot process. This can be useful when you want to disable a module that you suspect is causing problems on your system. You can still load blacklisted kernel modules manually using the `insmod` or `modprobe` tools.

To blacklist a module, create a file `/etc/modprobe.d/60-blacklist-MODULE_NAME.conf` with the following content:

```
blacklist MODULE_NAME
```

Run the **dracut** command as root to generate a new `initrd` image, then reboot your machine (replace *NAME* with the name of the current initrd and *KERNELVERSION* with the currently running kernel):

```
> su
echo "blacklist nouveau" >> /etc/modprobe.d/60-blacklist-nouveau.conf
/usr/bin/dracut --logfile /var/log/YaST2/mkinitrd.log --force /boot/$initrd-NAME
 $KERNELVERSION
reboot
```

To disable a kernel module temporarily only, blacklist it on-the-fly during the boot. To do this, press the E key when you see the boot screen. This drops you into a minimal editor that allows you to modify boot parameters. Locate the line that looks as follows:

```
linux /boot/vmlinuz...splash= silent quiet showopts
```

Add the **modprobe.blacklist=*MODULE_NAME*** command to the end of the line. For example:

```
linux /boot/vmlinuz...splash= silent quiet showopts modprobe.blacklist=nouveau
```

Press `F10` or `Ctrl`–`X` to boot with the specified configuration.

To blacklist a kernel module permanently via GRUB, open the `/etc/default/grub` file for editing, and add the **`modprobe.blacklist=`***`MODULE_NAME`* option to the **`GRUB_CMDLINE_LINUX`** command. Then run the **`sudo grub2-mkconfig -o /boot/grub2/grub.cfg`** command to enable the changes.

29 Dynamic kernel device management with udev

Revision History

2024-05-13

The kernel can add or remove almost any device in a running system. Changes in the device state (whether a device is plugged in or removed) need to be propagated to user space. Devices need to be configured when they are plugged in and recognized. Users of a certain device need to be informed about any changes in this device's recognized state. `udev` provides the needed infrastructure to dynamically maintain the device node files and symbolic links in the `/dev` directory. `udev` rules provide a way to plug external tools into the kernel device event processing. This allows you to customize `udev` device handling by adding certain scripts to execute as part of kernel device handling, or request and import additional data to evaluate during device handling.

29.1 The /dev directory

The device nodes in the `/dev` directory provide access to the corresponding kernel devices. With `udev`, the `/dev` directory reflects the current state of the kernel. Every kernel device has one corresponding device file. If a device is disconnected from the system, the device node is removed.

The content of the `/dev` directory is kept on a temporary file system and all files are rendered at every system start-up. Manually created or modified files do not, by design, survive a reboot. Static files and directories that should always be in the `/dev` directory regardless of the state of the corresponding kernel device can be created with systemd-tmpfiles. The configuration files are found in `/usr/lib/tmpfiles.d/` and `/etc/tmpfiles.d/`; for more information, see the `systemd-tmpfiles(8)` man page.

29.2 Kernel uevents and udev

The required device information is exported by the `sysfs` file system. For every device the kernel has detected and initialized, a directory with the device name is created. It contains attribute files with device-specific properties.

Every time a device is added or removed, the kernel sends a uevent to notify `udev` of the change. The `udev` daemon reads and parses all rules from the `/usr/lib/udev/rules.d/*.rules` and `/etc/udev/rules.d/*.rules` files at start-up and keeps them in memory. If rules files are changed, added or removed, the daemon can reload their in-memory representation with the command **udevadm control --reload**. For more details on udev rules and their syntax, refer to *Section 29.6, "Influencing kernel device event handling with udev rules"*.

Every received event is matched against the set of provides rules. The rules can add or change event environment keys, request a specific name for the device node to create, add symbolic links pointing to the node or add programs to run after the device node is created. The driver core uevents are received from a kernel netlink socket.

29.3 Drivers, kernel modules and devices

The kernel bus drivers probe for devices. For every detected device, the kernel creates an internal device structure while the driver core sends a uevent to the udev daemon. Bus devices identify themselves by a specially formatted ID, which tells what kind of device it is. These IDs consist of vendor and product ID and other subsystem-specific values. Every bus has its own scheme for these IDs, called MODALIAS. The kernel takes the device information, composes a MODALIAS ID string from it and sends that string along with the event. For a USB mouse, it looks like this:

```
MODALIAS=usb:v046DpC03Ed2000dc00dsc00dp00ic03isc01ip02
```

Every device driver carries a list of known aliases for devices it can handle. The list is contained in the kernel module file itself. The program depmod reads the ID lists and creates the file modules.alias in the kernel's /lib/modules directory for all currently available modules. With this infrastructure, module loading is as easy as calling **modprobe** for every event that carries a MODALIAS key. If **modprobe $MODALIAS** is called, it matches the device alias composed for the device with the aliases provided by the modules. If a matching entry is found, that module is loaded. All this is automatically triggered by udev.

29.4 Booting and initial device setup

All device events happening during the boot process before the udev daemon is running are lost, because the infrastructure to handle these events resides on the root file system and is not available at that time. To cover that loss, the kernel provides a uevent file located in the device directory of every device in the sysfs file system. By writing add to that file, the kernel resends the same event as the one lost during boot. A simple loop over all uevent files in /sys triggers all events again to create the device nodes and perform device setup.

As an example, a USB mouse present during boot may not be initialized by the early boot logic, because the driver is not available at that time. The event for the device discovery was lost and failed to find a kernel module for the device. Instead of manually searching for connected devices, udev requests all device events from the kernel after the root file system is available, so the event for the USB mouse device runs again. Now it finds the kernel module on the mounted root file system and the USB mouse can be initialized.

From user space, there is no visible difference between a device coldplug sequence and a device discovery during runtime. In both cases, the same rules are used to match and the same configured programs are run.

29.5 Monitoring the running udev daemon

The program **udevadm monitor** can be used to visualize the driver core events and the timing of the udev event processes.

```
UEVENT[1185238505.276660] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1 (usb)
UDEV  [1185238505.279198] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1 (usb)
UEVENT[1185238505.279527] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0 (usb)
UDEV  [1185238505.285573] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0 (usb)
UEVENT[1185238505.298878] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10 (input)
UDEV  [1185238505.305026] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10 (input)
UEVENT[1185238505.305442] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/mouse2 (input)
UEVENT[1185238505.306440] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/event4 (input)
UDEV  [1185238505.325384] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/event4 (input)
UDEV  [1185238505.342257] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/mouse2 (input)
```

The UEVENT lines show the events the kernel has sent over netlink. The UDEV lines show the finished udev event handlers. The timing is printed in microseconds. The time between UEVENT and UDEV is the time udev took to process this event or the udev daemon has delayed its execution to synchronize this event with related and already running events. For example, events for hard disk partitions always wait for the main disk device event to finish, because the partition events may rely on the data that the main disk event has queried from the hardware.

`udevadm monitor --env` shows the complete event environment:

```
ACTION=add
DEVPATH=/devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/input10
SUBSYSTEM=input
SEQNUM=1181
NAME="Logitech USB-PS/2 Optical Mouse"
PHYS="usb-0000:00:1d.2-1/input0"
UNIQ=""
EV=7
KEY=70000 0 0 0 0
REL=103
MODALIAS=input:b0003v046DpC03Ee0110-e0,1,2,k110,111,112,r0,1,8,amlsfw
```

`udev` also sends messages to syslog. The default syslog priority that controls which messages are sent to syslog is specified in the `udev` configuration file `/etc/udev/udev.conf`. The log priority of the running daemon can be changed with **`udevadm control --log_priority=`***`LEVEL/NUMBER`*.

29.6 Influencing kernel device event handling with udev rules

A `udev` rule can match any property the kernel adds to the event itself or any information that the kernel exports to `sysfs`. The rule can also request additional information from external programs. Events are matched against all rules provided in the directories `/usr/lib/udev/rules.d/` (for default rules) and `/etc/udev/rules.d` (system-specific configuration).

Every line in the rules file contains at least one key value pair. There are two kinds of keys, match and assignment keys. If all match keys match their values, the rule is applied and the assignment keys are assigned the specified value. A matching rule may specify the name of the device node, add symbolic links pointing to the node or run a specified program as part of the event handling. If no matching rule is found, the default device node name is used to create the device node. Detailed information about the rule syntax and the provided keys to match or import data are described in the `udev` man page. The following example rules provide a basic introduction to `udev` rule syntax. The example rules are all taken from the `udev` default rule set `/usr/lib/udev/rules.d/50-udev-default.rules`.

EXAMPLE 29.1: EXAMPLE udev RULES

```
# console
```

```
KERNEL=="console", MODE="0600", OPTIONS="last_rule"

# serial devices
KERNEL=="ttyUSB*", ATTRS{product}=="[Pp]alm*Handheld*", SYMLINK+="pilot"

# printer
SUBSYSTEM=="usb", KERNEL=="lp*", NAME="usb/%k", SYMLINK+="usb%k", GROUP="lp"

# kernel firmware loader
SUBSYSTEM=="firmware", ACTION=="add", RUN+="firmware.sh"
```

The `console` rule consists of three keys: one match key (`KERNEL`) and two assign keys (`MODE`, `OPTIONS`). The `KERNEL` match rule searches the device list for any items of the type `console`. Only exact matches are valid and trigger this rule to be executed. The `MODE` key assigns special permissions to the device node, in this case, read and write permissions to the owner of this device only. The `OPTIONS` key makes this rule the last rule to be applied to any device of this type. Any later rule matching this particular device type does not have any effect.

The `serial devices` rule is not available in `50-udev-default.rules` anymore, but it is still worth considering. It consists of two match keys (`KERNEL` and `ATTRS`) and one assign key (`SYMLINK`). The `KERNEL` key searches for all devices of the `ttyUSB` type. Using the `*` wild card, this key matches several of these devices. The second match key, `ATTRS`, checks whether the `product` attribute file in `sysfs` for any `ttyUSB` device contains a certain string. The assign key (`SYMLINK`) triggers the addition of a symbolic link to this device under `/dev/pilot`. The operator used in this key (`+=`) tells `udev` to additionally perform this action, even if previous or later rules add other symbolic links. As this rule contains two match keys, it is only applied if both conditions are met.

The `printer` rule deals with USB printers and contains two match keys which must both apply to get the entire rule applied (`SUBSYSTEM` and `KERNEL`). Three assign keys deal with the naming for this device type (`NAME`), the creation of symbolic device links (`SYMLINK`) and the group membership for this device type (`GROUP`). Using the `*` wild card in the `KERNEL` key makes it match several `lp` printer devices. Substitutions are used in both, the `NAME` and the `SYMLINK` keys to extend these strings by the internal device name. For example, the symbolic link to the first `lp` USB printer would read `/dev/usblp0`.

The `kernel firmware loader` rule makes `udev` load additional firmware by an external helper script during runtime. The `SUBSYSTEM` match key searches for the `firmware` subsystem. The `ACTION` key checks whether any device belonging to the `firmware` subsystem has been added. The `RUN+=` key triggers the execution of the `firmware.sh` script to locate the firmware that is to be loaded.

General characteristics are common to all rules:

- Each rule consists of one or more key value pairs separated by a comma.
- A key's operation is determined by the operator. `udev` rules support several operators.
- Each given value must be enclosed by quotation marks.
- Each line of the rules file represents one rule. If a rule is longer than one line, use `\` to join the different lines as you would do in shell syntax.
- `udev` rules support a shell-style pattern that matches the `*`, `?`, and `[]` patterns.
- `udev` rules support substitutions.

29.6.1 Using operators in udev rules

Creating keys you can choose from several operators, depending on the type of key you want to create. Match keys will normally be used to find a value that either matches or explicitly mismatches the search value. Match keys contain either of the following operators:

`==`

Compare for equality. If the key contains a search pattern, all results matching this pattern are valid.

`!=`

Compare for non-equality. If the key contains a search pattern, all results matching this pattern are valid.

Any of the following operators can be used with assign keys:

`=`

Assign a value to a key. If the key previously consisted of a list of values, the key resets and only the single value is assigned.

`+=`

Add a value to a key that contains a list of entries.

`:=`

Assign a final value. Disallow any later change by later rules.

29.6.2 Using substitutions in udev rules

`udev` rules support the use of placeholders and substitutions. Use them in a similar fashion as you would do in any other scripts. The following substitutions can be used with `udev` rules:

`%r`, `$root`
: The device directory, `/dev` by default.

`%p`, `$devpath`
: The value of `DEVPATH`.

`%k`, `$kernel`
: The value of `KERNEL` or the internal device name.

`%n`, `$number`
: The device number.

`%N`, `$tempnode`
: The temporary name of the device file.

`%M`, `$major`
: The major number of the device.

`%m`, `$minor`
: The minor number of the device.

`%s{ATTRIBUTE}`, `$attr{ATTRIBUTE}`
: The value of a `sysfs` attribute (specified by *ATTRIBUTE*).

`%E{VARIABLE}`, `$env{VARIABLE}`
: The value of an environment variable (specified by *VARIABLE*).

`%c`, `$result`
: The output of `PROGRAM`.

`%%`
: The `%` character.

`$$`
: The `$` character.

29.6.3 Using udev match keys

Match keys describe conditions that must be met before a `udev` rule can be applied. The following match keys are available:

`ACTION`
: The name of the event action, for example, `add` or `remove` when adding or removing a device.

`DEVPATH`
: The device path of the event device, for example, `DEVPATH=/bus/pci/drivers/ipw3945` to search for all events related to the ipw3945 driver.

`KERNEL`
: The internal (kernel) name of the event device.

`SUBSYSTEM`
: The subsystem of the event device, for example, `SUBSYSTEM=usb` for all events related to USB devices.

`ATTR{FILENAME}`
: `sysfs` attributes of the event device. To match a string contained in the `vendor` attribute file name, you could use `ATTR{vendor}=="On[sS]tream"`, for example.

`KERNELS`
: Let `udev` search the device path upward for a matching device name.

`SUBSYSTEMS`
: Let `udev` search the device path upward for a matching device subsystem name.

`DRIVERS`
: Let `udev` search the device path upward for a matching device driver name.

`ATTRS{FILENAME}`
: Let `udev` search the device path upward for a device with matching `sysfs` attribute values.

`ENV{KEY}`
: The value of an environment variable, for example, `ENV{ID_BUS}="ieee1394` to search for all events related to the FireWire bus ID.

`PROGRAM`
: Let `udev` execute an external program. To be successful, the program must return with exit code zero. The program's output, printed to STDOUT, is available to the `RESULT` key.

`RESULT`

Match the output string of the last `PROGRAM` call. Either include this key in the same rule as the `PROGRAM` key or in a later one.

29.6.4 Using udev assign keys

In contrast to the match keys described above, assign keys do not describe conditions that must be met. They assign values, names and actions to the device nodes maintained by `udev`.

`NAME`

The name of the device node to be created. After a rule has set a node name, all other rules with a `NAME` key for this node are ignored.

`SYMLINK`

The name of a symbolic link related to the node to be created. Multiple matching rules can add symbolic links to be created with the device node. You can also specify multiple symbolic links for one node in one rule using the space character to separate the symbolic link names.

`OWNER, GROUP, MODE`

The permissions for the new device node. Values specified here overwrite anything that has been compiled in.

`ATTR{KEY}`

Specify a value to be written to a `sysfs` attribute of the event device. If the `==` operator is used, this key is also used to match against the value of a `sysfs` attribute.

`ENV{KEY}`

Tell `udev` to export a variable to the environment. If the `==` operator is used, this key is also used to match against an environment variable.

`RUN`

Tell `udev` to add a program to the list of programs to be executed for this device. Keep in mind to restrict this to short tasks to avoid blocking further events for this device.

`LABEL`

Add a label where a `GOTO` can jump to.

`GOTO`

Tell `udev` to skip several rules and continue with the one that carries the label referenced by the `GOTO` key.

`IMPORT{TYPE}`

Load variables into the event environment such as the output of an external program. `udev` imports variables of several types. If no type is specified, `udev` tries to determine the type itself based on the executable bit of the file permissions.

- `program` tells `udev` to execute an external program and import its output.
- `file` tells `udev` to import a text file.
- `parent` tells `udev` to import the stored keys from the parent device.

`WAIT_FOR_SYSFS`

Tells `udev` to wait for the specified `sysfs` file to be created for a certain device. For example, `WAIT_FOR_SYSFS="ioerr_cnt"` informs `udev` to wait until the `ioerr_cnt` file has been created.

`OPTIONS`

The `OPTION` key may have several values:

- `last_rule` tells `udev` to ignore all later rules.
- `ignore_device` tells `udev` to ignore this event.
- `ignore_remove` tells `udev` to ignore all later remove events for the device.
- `all_partitions` tells `udev` to create device nodes for all available partitions on a block device.

29.7 Persistent device naming

The dynamic device directory and the `udev` rules infrastructure make it possible to provide stable names for all disk devices—regardless of their order of recognition or the connection used for the device. Every appropriate block device the kernel creates is examined by tools with special knowledge about certain buses, drive types or file systems. Along with the dynamic kernel-provided device node name, `udev` maintains classes of persistent symbolic links pointing to the device:

```
/dev/disk
```

```
|-- by-id
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B -> ../../sda
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part1 -> ../../sda1
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part6 -> ../../sda6
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part7 -> ../../sda7
|   |-- usb-Generic_STORAGE_DEVICE_02773 -> ../../sdd
|   `-- usb-Generic_STORAGE_DEVICE_02773-part1 -> ../../sdd1
|-- by-label
|   |-- Photos -> ../../sdd1
|   |-- SUSE10 -> ../../sda7
|   `-- devel -> ../../sda6
|-- by-path
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0 -> ../../sda
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part1 -> ../../sda1
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part6 -> ../../sda6
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part7 -> ../../sda7
|   |-- pci-0000:00:1f.2-scsi-1:0:0:0 -> ../../sr0
|   |-- usb-02773:0:0:2 -> ../../sdd
|   |-- usb-02773:0:0:2-part1 -> ../../sdd1
`-- by-uuid
    |-- 159a47a4-e6e6-40be-a757-a629991479ae -> ../../sda7
    |-- 3e999973-00c9-4917-9442-b7633bd95b9e -> ../../sda6
    `-- 4210-8F8C -> ../../sdd1
```

29.8 Files used by udev

`/sys/*`
: Virtual file system provided by the Linux kernel, exporting all currently known devices. This information is used by `udev` to create device nodes in `/dev`

`/dev/*`
: Dynamically created device nodes and static content created with systemd-tmpfiles; for more information, see the `systemd-tmpfiles(8)` man page.

The following files and directories contain the crucial elements of the `udev` infrastructure:

`/etc/udev/udev.conf`
: Main `udev` configuration file.

`/etc/udev/rules.d/*`
: System-specific `udev` event matching rules. You can add custom rules here to modify or override the default rules from `/usr/lib/udev/rules.d/*`.

Files are parsed in alphanumeric order. Rules from files with a higher priority modify or override rules with lower priority. The lower the number, the higher the priority.

`/usr/lib/udev/rules.d/*`

Default `udev` event matching rules. The files in this directory are owned by packages and will be overwritten by updates. Do not add, remove or edit files here, use `/etc/udev/rules.d` instead.

`/usr/lib/udev/*`

Helper programs called from `udev` rules.

`/usr/lib/tmpfiles.d/` **and** `/etc/tmpfiles.d/`

Responsible for static `/dev` content.

29.9 More information

For more information about the `udev` infrastructure, refer to the following man pages:

`udev`

General information about `udev`, keys, rules and other important configuration issues.

`udevadm`

`udevadm` can be used to control the runtime behavior of `udev`, request kernel events, manage the event queue and provide simple debugging mechanisms.

`udevd`

Information about the `udev` event managing daemon.

30 Special system features

Revision History

2024-05-13

This chapter starts with information about specific software packages, the virtual consoles and the keyboard layout. We talk about software components like `bash`, `cron` and `logrotate`, because they were changed or enhanced during the last release cycles. Even if they are small or considered of minor importance, users should change their default behavior, because these components are often closely coupled with the system. The chapter concludes with a section about language and country-specific settings (I18N and L10N).

30.1 Information about special software packages

The following chapter provides basic information about the following tools: `bash`, `cron`, `logrotate`, `locate`, `ulimit` and `free`.

30.1.1 The bash package and `/etc/profile`

Bash is the default system shell. When used as a login shell, it reads several initialization files. Bash processes them in the order they appear in this list:

1. `/etc/profile`
2. `~/.profile`
3. `/etc/bash.bashrc`
4. `~/.bashrc`

Make custom settings in `~/.profile` or `~/.bashrc`. To ensure the correct processing of these files, it is necessary to copy the basic settings from `/etc/skel/.profile` or `/etc/skel/.bashrc` into the home directory of the user. It is recommended to copy the settings from `/etc/skel` after an update. Execute the following shell commands to prevent the loss of personal adjustments:

```
> mv ~/.bashrc ~/.bashrc.old
```

```
> cp /etc/skel/.bashrc ~/.bashrc
> mv ~/.profile ~/.profile.old
> cp /etc/skel/.profile ~/.profile
```

Then copy personal adjustments back from the `*.old` files.

30.1.2 The cron package

Use `cron` to automatically run commands in the background at predefined times. `cron` uses specially formatted time tables, and the tool comes with several default ones. Users can also specify custom tables, if needed.

The cron tables are located in `/var/spool/cron/tabs`. `/etc/crontab` serves as a systemwide cron table. Enter the user name to run the command directly after the time table and before the command. In *Example 30.1, "Entry in /etc/crontab"*, `root` is entered. Package-specific tables, located in `/etc/cron.d`, have the same format. See the **`cron`** man page (**`man cron`**).

EXAMPLE 30.1: ENTRY IN /ETC/CRONTAB

```
1-59/5 * * * *   root   test -x /usr/sbin/atrun && /usr/sbin/atrun
```

You cannot edit `/etc/crontab` by calling the command **`crontab -e`**. This file must be loaded directly into an editor, then modified and saved.

Several packages install shell scripts to the directories `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly` and `/etc/cron.monthly`, whose execution is controlled by `/usr/lib/cron/run-crons`. `/usr/lib/cron/run-crons` is run every 15 minutes from the main table (`/etc/crontab`). This guarantees that processes that may have been neglected can be run at the proper time.

To run the `hourly`, `daily` or other periodic maintenance scripts at custom times, remove the time stamp files regularly using `/etc/crontab` entries (see *Example 30.2, "/etc/crontab: remove time stamp files"*, which removes the `hourly` one before every full hour, the `daily` one once a day at 2:14 a.m., etc.).

EXAMPLE 30.2: /ETC/CRONTAB: REMOVE TIME STAMP FILES

```
59 *  * * *     root  rm -f /var/spool/cron/lastrun/cron.hourly
14 2  * * *     root  rm -f /var/spool/cron/lastrun/cron.daily
29 2  * * 6     root  rm -f /var/spool/cron/lastrun/cron.weekly
44 2  1 * *     root  rm -f /var/spool/cron/lastrun/cron.monthly
```

Or you can set DAILY_TIME in /etc/sysconfig/cron to the time at which cron.daily should start. The setting of MAX_NOT_RUN ensures that the daily tasks get triggered to run, even if the user did not turn on the computer at the specified DAILY_TIME for a longer time. The maximum value of MAX_NOT_RUN is 14 days.

30.1.3 Stopping cron status messages

To avoid the mail flood caused by cron status messages, the default value of SEND_MAIL_ON_NO_ERROR in /etc/sysconfig/cron is set to "no" for new installations. Even with this setting to "no", cron data output will still be sent to the MAILTO address, as documented in the cron man page.

In the update case it is recommended to set these values according to your needs.

30.1.4 Log files: package logrotate

There are several system services (*daemons*) that, along with the kernel itself, regularly record the system status and specific events onto log files. This way, the administrator can regularly check the status of the system at a certain point in time, recognize errors or faulty functions and troubleshoot them with pinpoint precision. These log files are normally stored in /var/log as specified by FHS and grow on a daily basis. The logrotate package helps control the growth of these files. For more details refer to *Book "System Analysis and Tuning Guide", Chapter 3 "System log files", Section 3.3 "Managing log files with* **logrotate***"*.

30.1.5 The **locate** command

locate, a command for quickly finding files, is not included in the standard scope of installed software. If desired, install the package mlocate, the successor of the package findutils-locate. The updatedb process is started automatically every night or about 15 minutes after booting the system.

30.1.6 The **ulimit** command

With the **ulimit** (*user limits*) command, it is possible to set limits for the use of system resources and to have these displayed. **ulimit** is especially useful for limiting available memory for applications. With this, an application can be prevented from co-opting too much of the system resources and slowing or even hanging up the operating system.

ulimit can be used with various options. To limit memory usage, use the options listed in *Table 30.1, "**ulimit**: Setting resources for the user"*.

TABLE 30.1: **ulimit**: SETTING RESOURCES FOR THE USER

`-m`	The maximum resident set size
`-v`	The maximum amount of virtual memory available to the shell
`-s`	The maximum size of the stack
`-c`	The maximum size of core files created
`-a`	All current limits are reported

Systemwide default entries are set in `/etc/profile`. Editing this file directly is not recommended, because changes will be overwritten during system upgrades. To customize systemwide profile settings, use `/etc/profile.local`. Per-user settings should be made in *~USER*`/.profile`.

EXAMPLE 30.3: **ulimit**: SETTINGS IN `~/.bashrc`

```
# Limits maximum resident set size (physical memory):
ulimit -m 98304

# Limits of virtual memory:
ulimit -v 98304
```

Memory allocations must be specified in KB. For more detailed information, see **man bash**.

Important: **ulimit** support

Not all shells support **ulimit** directives. PAM (for example, `pam_limits`) offers comprehensive adjustment possibilities as an alternative to **ulimit**.

30.1.7 The **free** command

The **free** command displays the total amount of free and used physical memory and swap space in the system and the buffers and cache consumed by the kernel. The concept of *available RAM* dates back to before the days of unified memory management. The slogan *free memory is bad memory* applies well to Linux. As a result, Linux has always made the effort to balance out caches without allowing free or unused memory.

The kernel does not have direct knowledge of any applications or user data. Instead, it manages applications and user data in a *page cache*. If memory runs short, parts of it are written to the swap partition or to files, from which they can initially be read using the **mmap** command (see **man mmap**).

The kernel also contains other caches, such as the *slab cache*, where the caches used for network access are stored. This may explain the differences between the counters in `/proc/meminfo`. Most, but not all, of them can be accessed via `/proc/slabinfo`.

However, if your goal is to find out how much RAM is currently being used, find this information in `/proc/meminfo`.

30.1.8 Man pages and info pages

For some GNU applications (such as tar), the man pages are no longer maintained. For these commands, use the `--help` option to get a quick overview of the info pages, which provide more in-depth instructions. Info is GNU's hypertext system. Read an introduction to this system by entering **info** `info`. Info pages can be viewed with Emacs by entering **emacs** `-f info` or directly in a console with **info**. You can also use tkinfo, xinfo or the help system to view info pages.

30.1.9 Selecting man pages using the **man** command

To read a man page enter **man** *MAN_PAGE*. If a man page with the same name exists in different sections, they will all be listed with the corresponding section numbers. Select the one to display. If you do not enter a section number within a few seconds, the first man page will be displayed.

To change this to the default system behavior, set `MAN_POSIXLY_CORRECT=1` in a shell initialization file such as `~/.bashrc`.

30.1.10 Settings for GNU Emacs

GNU Emacs is a complex work environment. The following sections cover the configuration files processed when GNU Emacs is started. More information is available at https://www.gnu.org/software/emacs/.

On start-up, Emacs reads several files containing the settings of the user, system administrator and distributor for customization or preconfiguration. The initialization file `~/.emacs` is installed to the home directories of the individual users from `/etc/skel`. `.emacs`, in turn, reads the file `/etc/skel/.gnu-emacs`. To customize the program, copy `.gnu-emacs` to the home directory (with **`cp /etc/skel/.gnu-emacs ~/.gnu-emacs`**) and make the desired settings there.

`.gnu-emacs` defines the file `~/.gnu-emacs-custom` as `custom-file`. If users make settings with the `customize` options in Emacs, the settings are saved to `~/.gnu-emacs-custom`.

With SUSE Linux Enterprise Desktop, the `emacs` package installs the file `site-start.el` in the directory `/usr/share/emacs/site-lisp`. The file `site-start.el` is loaded before the initialization file `~/.emacs`. Among other things, `site-start.el` ensures that special configuration files distributed with Emacs add-on packages, such as `psgml`, are loaded automatically. Configuration files of this type are located in `/usr/share/emacs/site-lisp`, too, and always begin with `suse-start-`. The local system administrator can specify systemwide settings in `default.el`.

More information about these files is available in the Emacs info file under *Init File*: `info:/emacs/InitFile`. Information about how to disable the loading of these files (if necessary) is also provided at this location.

The components of Emacs are divided into several packages:

- The base package `emacs`.
- `emacs-x11` (usually installed): the program *with* X11 support.
- `emacs-nox`: the program *without* X11 support.
- `emacs-info`: online documentation in info format.
- `emacs-el`: the uncompiled library files in Emacs Lisp. These are not required at runtime.
- Numerous add-on packages can be installed if needed: `emacs-auctex` (LaTeX), `psgml` (SGML and XML), `gnuserv` (client and server operation) and others.

30.2 Virtual consoles

Linux is a multiuser and multitasking system. The advantages of these features can be appreciated even on a stand-alone PC system. In text mode, there are six virtual consoles available. Switch between them using `Alt`–`F1` through `Alt`–`F6`. The seventh console is reserved for X and the tenth console shows kernel messages.

To switch to a console from X without shutting it down, use `Ctrl`–`Alt`–`F1` to `Ctrl`–`Alt`–`F6`. To return to X, press `Alt`–`F7`.

30.3 Keyboard mapping

To standardize the keyboard mapping of programs, changes were made to the following files:

```
/etc/inputrc
/etc/X11/Xmodmap
/etc/skel/.emacs
/etc/skel/.gnu-emacs
/etc/skel/.vimrc
/etc/csh.cshrc
/etc/termcap
/usr/share/terminfo/x/xterm
/usr/share/X11/app-defaults/XTerm
/usr/share/emacs/VERSION/site-lisp/term/*.el
```

These changes only affect applications that use **`terminfo`** entries or whose configuration files are changed directly (**`vi`**, **`emacs`**, etc.). Applications not shipped with the system should be adapted to these defaults.

Under X, the compose key (multikey) can be enabled as explained in `/etc/X11/Xmodmap`.

Further settings are possible using the X Keyboard Extension (XKB).

Tip: More information

Information about XKB is available in the documents listed in `/usr/share/doc/packages/xkeyboard-config` (part of the `xkeyboard-config` package).

30.4 Language and country-specific settings

The system is, to a very large extent, internationalized and can be modified for local needs. Internationalization (*I18N*) allows specific localization (*L10N*). The abbreviations I18N and L10N are derived from the first and last letters of the words and, in between, the number of letters omitted.

Settings are made with `LC_` variables defined in the file `/etc/sysconfig/language`. This refers not only to *native language support*, but also to the categories *Messages* (Language), *Character Set*, *Sort Order*, *Time and Date*, *Numbers* and *Money*. Each of these categories can be defined directly with its own variable or indirectly with a master variable in the file `language` (see the **`locale`** man page).

LIST OF VARIABLES

`RC_LC_MESSAGES`, `RC_LC_CTYPE`, `RC_LC_COLLATE`, `RC_LC_TIME`, `RC_LC_NUMERIC`, `RC_LC_MONETARY`

These variables are passed to the shell without the `RC_` prefix and represent the listed categories. The shell profiles concerned are listed below. The current setting can be shown with the command **`locale`**.

`RC_LC_ALL`

This variable, if set, overwrites the values of the variables already mentioned.

`RC_LANG`

If none of the previous variables are set, this is the fallback. By default, only `RC_LANG` is set. This makes it easier for users to enter their own values.

`ROOT_USES_LANG`

This variable can be set to `yes` or `ctype` (default). If set to `yes`, `root` uses language and country-specific settings, otherwise the system administrator always works in a POSIX environment.

The variables can be set with the YaST sysconfig editor. The value of such a variable contains the language code, country code, encoding and modifier. The individual components are joined by special characters:

```
LANG=<language>[[_<COUNTRY>].<Encoding>[@<Modifier>]]
```

30.4.1 System-wide locale settings

systemd reads /etc/locale.conf at early boot. The locale settings configured in this file are inherited by every service or user, unless there are individual settings.

Note: Behavior of older configuration files under SUSE Linux Enterprise Desktop

Earlier versions of SUSE Linux Enterprise Desktop read locale settings from /etc/sysconfig/language, /etc/sysconfig/keyboard, and /etc/sysconfig/console. Starting with SUSE Linux Enterprise Desktop 15 GA, these files are considered obsolete. systemd does not read settings from these files anymore. Instead, systemd reads /etc/locale.conf.

However, variables defined in /etc/sysconfig/language will still be used: They override the system-wide locale and can be used to define different locale settings for user shells (see *Section 30.4.2, "Some examples"*).

To set the system-wide locale, you can either:

- Write your settings in /etc/locale.conf. Each line is a environment-like variable assignment (see **man 5 locale.conf** for a list of variables):

  ```
  LANG=de_DE.UTF-8
  ```

  To fine-tune the settings, you can add additional variables, one variable per line.

- Use the command **localectl**:

  ```
  # localectl set-locale LANG=de_DE.UTF-8
  ```

  Same here, you can also specify additional variables after the **localectl set-locale** command.

To keep backward compatibility with old systems during the update of the systemd package, all variables mentioned will be migrated from sysconfig to their final destinations if they are not already defined there.

30.4.2 Some examples

You should always set the language and country codes together. Language settings follow the standard ISO 639 available at https://www.evertype.com/standards/iso639/iso639-en.html and https://www.loc.gov/standards/iso639-2/. Country codes are listed in ISO 3166, see https://en.wikipedia.org/wiki/ISO_3166.

It only makes sense to set values for which usable description files can be found in `/usr/lib/locale`. Additional description files can be created from the files in `/usr/share/i18n` using the command **`localedef`**. The description files are part of the `glibc-i18ndata` package. A description file for `en_US.UTF-8` (for English and United States) can be created with:

```
localedef -i en_US -f UTF-8 en_US.UTF-8
```

`LANG=en_US.UTF-8`

This is the default setting if American English is selected during installation. If you selected another language, that language is enabled but still with UTF-8 as the character encoding.

`LANG=en_US.ISO-8859-1`

This sets the language to English, country to United States and the character set to `ISO-8859-1`. This character set does not support the Euro sign, but it can be useful sometimes for programs that have not been updated to support `UTF-8`. The string defining the charset (`ISO-8859-1` in this case) is then evaluated by programs like Emacs.

`LANG=en_IE@euro`

The above example explicitly includes the Euro sign in a language setting. This setting is obsolete now, as UTF-8 also covers the Euro symbol. It is only useful if an application supports ISO-8859-15 and not UTF-8.

Changes to `/etc/sysconfig/language` are activated by the following process chain:

- For the Bash: `/etc/profile` reads `/etc/profile.d/lang.sh` which, in turn, analyzes `/etc/sysconfig/language`.
- For tcsh: At login, `/etc/csh.login` reads `/etc/profile.d/lang.csh` which, in turn, analyzes `/etc/sysconfig/language`.

This ensures that any changes to `/etc/sysconfig/language` are available at the next login to the respective shell, without having to manually activate them.

Users can override the system defaults by editing their ~/.bashrc accordingly. For example, if you do not want to use the system-wide en_US for program messages, include LC_MESSAGES=es_ES so that messages are displayed in Spanish instead.

30.4.3 Locale settings in ~/.i18n

If you are not satisfied with locale system defaults, change the settings in ~/.i18n according to the Bash scripting syntax. Entries in ~/.i18n override system defaults from /etc/sysconfig/language. Use the same variable names but without the RC_ namespace prefixes. For example, use LANG instead of RC_LANG:

```
LANG=cs_CZ.UTF-8
LC_COLLATE=C
```

30.4.4 Settings for language support

Files in the category *Messages* are, as a rule, only stored in the corresponding language directory (like en) to have a fallback. If you set LANG to en_US and the message file in /usr/share/locale/en_US/LC_MESSAGES does not exist, it falls back to /usr/share/locale/en/LC_MESSAGES.

A fallback chain can also be defined, for example, for Breton to French or for Galician to Spanish to Portuguese:

LANGUAGE="br_FR:fr_FR"

LANGUAGE="gl_ES:es_ES:pt_PT"

If desired, use the Norwegian variants Nynorsk and Bokmål instead (with additional fallback to no):

LANG="nn_NO"

LANGUAGE="nn_NO:nb_NO:no"

or

LANG="nb_NO"

LANGUAGE="nb_NO:nn_NO:no"

In Norwegian, LC_TIME is also treated differently.

One problem that can arise is a separator used to delimit groups of digits not being recognized properly. This occurs if `LANG` is set to only a two-letter language code like `de`, but the definition file glibc uses is located in `/usr/share/lib/de_DE/LC_NUMERIC`. Thus `LC_NUMERIC` must be set to `de_DE` to make the separator definition visible to the system.

30.4.5 More information

- *The GNU C Library Reference Manual*, Chapter “Locales and Internationalization”. It is included in the package `glibc-info`.
- Markus Kuhn, *UTF-8 and Unicode FAQ for Unix/Linux*, currently at https://www.cl.cam.ac.uk/~mgk25/unicode.html ↗.

31 Using NetworkManager

Revision History

2024-05-13

NetworkManager is the ideal solution for laptops and other portable computers. It supports state-of-the-art encryption types and standards for network connections, including connections to 802.1X protected networks. 802.1X is the “IEEE Standard for Local and Metropolitan Area Networks—Port-Based Network Access Control”. With NetworkManager, you need not worry about configuring network interfaces and switching between wired or wireless networks when you are on the move. NetworkManager can automatically connect to known wireless networks or manage several network connections in parallel—the fastest connection is then used as default. Furthermore, you can manually switch between available networks and manage your network connection using an applet in the system tray.

Instead of only one connection being active, multiple connections may be active at once. This enables you to unplug your laptop from an Ethernet and remain connected via a wireless connection.

Important: Support coverage

NetworkManager is only supported by SUSE for desktop workloads with SLED or the Workstation extension. All server certifications are done with **`wicked`** as the network configuration tool, and using NetworkManager may invalidate them. NetworkManager is not supported by SUSE for server workloads.

31.1 Use cases for NetworkManager

NetworkManager provides a sophisticated and intuitive user interface, which enables users to easily switch their network environment. However, NetworkManager is not a suitable solution in the following cases:

- Your computer provides network services for other computers in your network, for example, it is a DHCP or DNS server.
- Your computer is a Xen server or your system is a virtual system inside Xen.

31.2 Enabling or disabling NetworkManager

On desktop and laptop computers, NetworkManager is enabled by default. You can disable and enable it at any time using the Network Settings module in YaST.

1. Run YaST and go to *System* › *Network Settings*.
2. The *Network Settings* dialog opens. Go to the *Global Options* tab.
3. To configure and manage your network connections with NetworkManager:
   a. In the *Network Setup Method* field, select *User Controlled with NetworkManager*.
   b. Click *OK* and close YaST.
   c. Configure your network connections with NetworkManager as described in *Section 31.3, "Configuring network connections"*.
4. To deactivate NetworkManager and control the network with your own configuration:
   a. In the *Network Setup Method* field, choose *Controlled by wicked*.
   b. Click *OK*.
   c. Set up your network card with YaST using automatic configuration via DHCP or a static external IP address.
      Find a detailed description of the network configuration with YaST in *Section 23.4, "Configuring a network connection with YaST"*.

31.3 Configuring network connections

After enabling NetworkManager in YaST, configure your network connections with the NetworkManager front-end available in GNOME. It shows tabs for all types of network connections, such as wired, wireless, mobile broadband, DSL and VPN connections.

To open the network configuration dialog in GNOME, open the settings menu via the status menu and click the *Network* entry.

Note: Availability of options

Depending on your system setup, you may not be allowed to configure certain connections. In a secured environment, certain options may be locked or require `root` permission. Ask your system administrator for details.

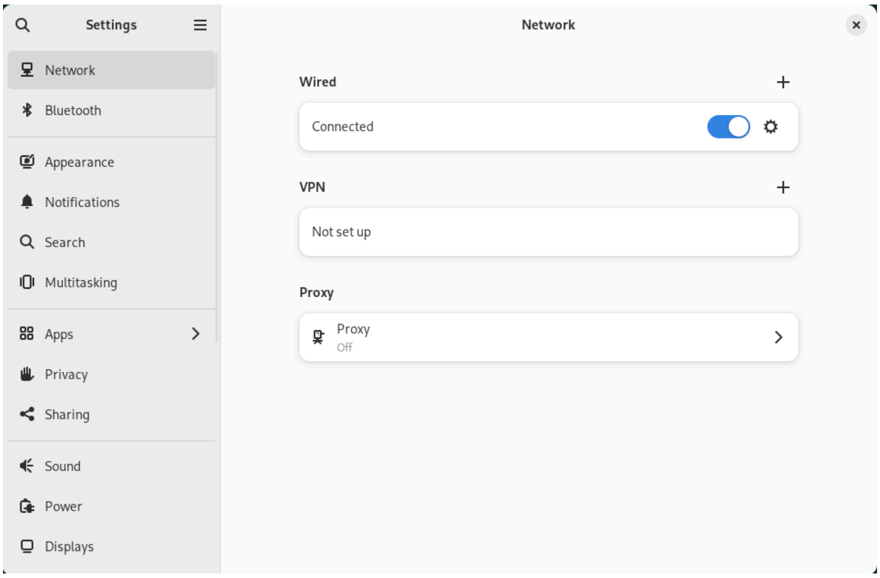


FIGURE 31.1: GNOME NETWORK CONNECTIONS DIALOG

PROCEDURE 31.1: ADDING AND EDITING CONNECTIONS

1. Open the Status Menu, click the gear icon to open *Settings* and click *Network* from the left menu.
2. To add a Connection:
   a. Click the + icon next to the connection type tab that you want to add.
   b. Depending on the connection type, fill the required fields in the corresponding dialog.
   c. When you are finished click *Add*.
   d. After confirming your changes, the newly configured network connection appears in the list of available networks in the Status Menu.
3. To edit a connection:
   a. Click the gear icon on the right side of the connection type tab that you want to edit.

b. Insert your changes and click *Apply* to save them.

c. To make your connection available as a system connection go to the *Details* tab and set the check box *Make available to other users*. For more information about user and system connections, see *Section 31.4.1, “User and system connections”*.

31.3.1 Managing wired network connections

If your computer is connected to a wired network, use the NetworkManager applet to manage the connection.

1. Open the Status Menu and click *Wired* to switch it off or click the related right arrow to change the connection details.
2. To change the settings click *Wired Settings* and then click the gear icon.
3. To switch off all network connections, activate the *Airplane Mode* setting.

31.3.2 Managing wireless network connections

Visible wireless networks are listed in the GNOME NetworkManager applet menu under *Wireless Networks*. The signal strength of each network is also shown in the menu. Encrypted wireless networks are marked with a shield icon.

PROCEDURE 31.2: CONNECTING TO A VISIBLE WIRELESS NETWORK

1. To connect to a visible wireless network, open the Status Menu and click *Wi-Fi*.
2. Click *Turn On* to enable it.
3. Click *Select Network*, select your Wi-Fi Network and click *Connect*.
4. If the network is encrypted, a configuration dialog opens. It shows the type of encryption the network uses and text boxes for entering the login credentials.

PROCEDURE 31.3: CONNECTING TO AN INVISIBLE WIRELESS NETWORK

1. To connect to a network that does not broadcast its service set identifier (SSID or ESSID) and therefore cannot be detected automatically, open the Status Menu and click *Wi-Fi*.

2. Click *Wi-Fi Settings* to open the detailed settings menu.
3. Make sure your Wi-Fi is enabled and click *Connect to Hidden Network*.
4. In the dialog that opens, enter the SSID or ESSID in *Network Name* and set encryption parameters if necessary.

A wireless network that has been chosen explicitly remains connected as long as possible. If a network cable is plugged in during that time, any connections that have been set to *Stay connected when possible* will be connected, while the wireless connection remains up.

31.3.3 Enabling wireless captive portal detection

On the initial connection, many public wireless hotspots force users to visit a landing page (the *captive portal*). Before you have logged in or agreed to the terms and conditions, all your HTTP requests are redirected to the provider's captive portal.

When connecting to a wireless network with a captive portal, NetworkManager and GNOME automatically show the login page as part of the connection process. This ensures that you always know when you are connected, and helps you to get set up as quickly as possible without using the browser to login.

To enable this feature, install the package `NetworkManager-branding-SLE` and restart NetworkManager with:

```
> sudo systemctl restart network
```

Whenever you connect to a network with a captive portal, NetworkManager (or GNOME) opens the captive portal login page for you. Login with your credentials to get access to the Internet.

31.3.4 Configuring your Wi-Fi/Bluetooth card as an access point

If your Wi-Fi/Bluetooth card supports access point mode, you can use NetworkManager for the configuration.

1. Open the Status Menu and click *Wi-Fi*.
2. Click *Wi-Fi Settings* to open the detailed settings menu.
3. Click *Use as Hotspot* and follow the instructions.

4. Use the credentials shown in the resulting dialog to connect to the hotspot from a remote machine.

31.3.5 NetworkManager and VPN

NetworkManager supports several Virtual Private Network (VPN) technologies. For each technology, SUSE Linux Enterprise Desktop comes with a base package providing the generic support for NetworkManager. Besides that, you also need to install the respective desktop-specific package for your applet.

OpenVPN

To use this VPN technology, install:

- `NetworkManager-openvpn`
- `NetworkManager-openvpn-gnome`

OpenConnect

To use this VPN technology, install:

- `NetworkManager-openconnect`
- `NetworkManager-openconnect-gnome`

PPTP (point-to-point tunneling protocol)

To use this VPN technology, install:

- `NetworkManager-pptp`
- `NetworkManager-pptp-gnome`

The following procedure describes how to set up your computer as an OpenVPN client using NetworkManager. Setting up other types of VPNs works analogously.

Before you begin, make sure that the package `NetworkManager-openvpn-gnome` is installed and all dependencies have been resolved.

PROCEDURE 31.4: SETTING UP OPENVPN WITH NETWORKMANAGER

1. Open the application *Settings* by clicking the status icons at the right end of the panel and clicking the *wrench and screwdriver* icon. In the window *All Settings*, choose *Network*.
2. Click the + icon.

3. Select *VPN* and then *OpenVPN*.
4. Choose the *Authentication* type. Depending on the setup of your OpenVPN server, choose *Certificates (TLS)* or *Password with Certificates (TLS)*.
5. Insert the necessary values into the respective text boxes. For our example configuration, these are:

Gateway	The remote endpoint of the VPN server
User name	The user (only available when you have selected *Password with Certificates (TLS)*)
Password	The password for the user (only available when you have selected *Password with Certificates (TLS)*)
User Certificate	`/etc/openvpn/client1.crt`
CA Certificate	`/etc/openvpn/ca.crt`
Private Key	`/etc/openvpn/client1.key`

6. Finish the configuration with *Add*.
7. To enable the connection, in the *Network* panel of the *Settings* application click the switch button. Alternatively, click the status icons at the right end of the panel, click the name of your VPN and then *Connect*.

31.4 NetworkManager and security

NetworkManager distinguishes two types of wireless connections: trusted and untrusted. A trusted connection is any network that you explicitly selected in the past. All others are untrusted. Trusted connections are identified by the name and MAC address of the access point. Using the MAC address ensures that you cannot use a different access point with the name of your trusted connection.

NetworkManager periodically scans for available wireless networks. If multiple trusted networks are found, the most recently used is automatically selected. NetworkManager waits for your selection in case if all networks are untrusted.

If the encryption setting changes but the name and MAC address remain the same, NetworkManager attempts to connect, but first you are asked to confirm the new encryption settings and provide any updates, such as a new key.

If you switch from using a wireless connection to offline mode, NetworkManager blanks the SSID or ESSID. This ensures that the card is disconnected.

31.4.1 User and system connections

NetworkManager knows two types of connections: `user` and `system` connections.

User connections require every user to authenticate in NetworkManager, which stores the user's credentials in their local GNOME keyring so that they do not need to re-enter them every time they connect.

System connections are available to all users automatically. The first user to create the connection enters any necessary credentials, and then all other users have access without needing to know the credentials. The difference in configuring a user or system connection is a single check box, *Make available to other users*. For information on how to configure user or system connections with NetworkManager, refer to *Section 31.3, “Configuring network connections”*.

31.4.2 Storing passwords and credentials

If you do not want to re-enter your credentials each time you want to connect to an encrypted network, you can use the GNOME Keyring Manager to store your credentials encrypted on the disk, secured by a master password.

31.4.3 Firewall zones

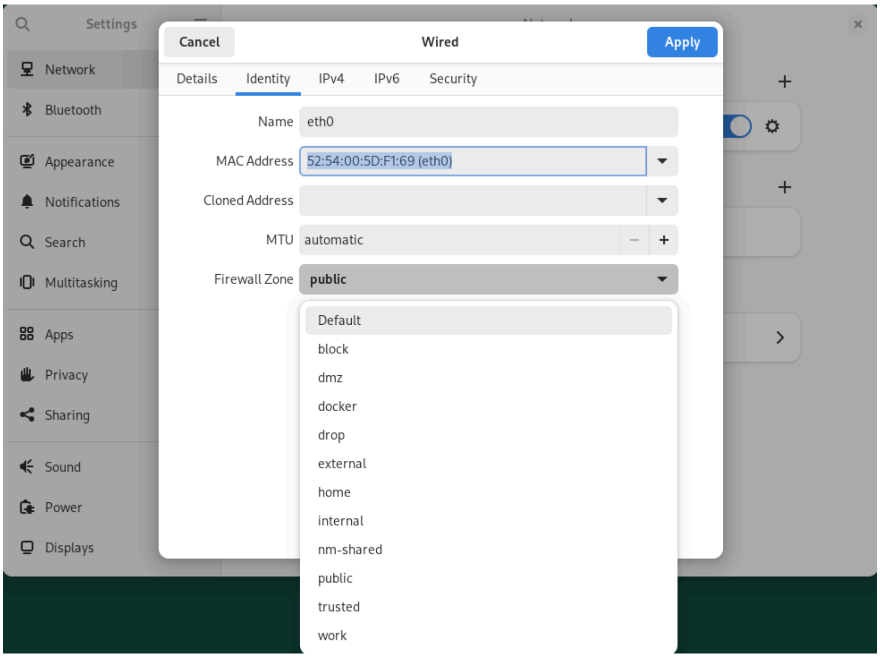


FIGURE 31.2: `firewalld` ZONES IN NETWORKMANAGER

The firewall zones set general rules about which network connections are allowed. To configure the zone of *firewalld* for a wired connection, go to the *Identity* tab of the connection settings. To configure the zone of *firewalld* for a Wi-Fi connection, go to the *Security* tab of the connection settings.

If you are in your home network, use the zone `home`. For public wireless networks, switch to `public`. If you are in a secure environment and want to allow all connections, use the zone `trusted`.

For details about firewalld, see *Book “Security and Hardening Guide”, Chapter 23 “Masquerading and firewalls”, Section 23.4 “*`firewalld`*”*.

31.5 Frequently asked questions

In the following, find several frequently asked questions about configuring special network options with NetworkManager.

5. *How to tie a connection to a specific device?*

   By default, connections in NetworkManager are device type-specific: they apply to all physical devices with the same type. If more than one physical device per connection type is available (for example, your machine is equipped with two Ethernet cards), you can tie a connection to a certain device.

   To do this in GNOME, first look up the MAC address of your device (use the *Connection Information* available from the applet, or use the output of command line tools like **`nm-tool`** or **`wicked show all`**). Then start the dialog for configuring network connections and choose the connection you want to modify. On the *Wired* or *Wireless* tab, enter the *MAC Address* of the device and confirm your changes.

6. *How to specify a certain access point in case multiple access points with the same ESSID are detected?*

   When multiple access points with different wireless bands (a/b/g/n) are available, the access point with the strongest signal is automatically chosen by default. To override this, use the *BSSID* field when configuring wireless connections.

   The Basic Service Set Identifier (BSSID) uniquely identifies each Basic Service Set. In an infrastructure Basic Service Set, the BSSID is the MAC address of the wireless access point. In an independent (ad-hoc) Basic Service Set, the BSSID is a locally administered MAC address generated from a 46-bit random number.

   Start the dialog for configuring network connections as described in *Section 31.3, "Configuring network connections"*. Choose the wireless connection you want to modify and click *Edit*. On the *Wireless* tab, enter the BSSID.

7. *How to share network connections with other computers?*

   The primary device (the device which is connected to the Internet) does not need any special configuration. However, you need to configure the device that is connected to the local hub or machine as follows:

   1. Start the dialog for configuring network connections as described in *Section 31.3, "Configuring network connections"*. Choose the connection you want to modify and click *Edit*. Switch to the *IPv4 Settings* tab and from the *Method* drop-down list, activate *Shared to other computers*. That will enable IP traffic forwarding and run a DHCP server on the device. Confirm your changes in NetworkManager.

   2. As the DHCP server uses port `67`, make sure that it is not blocked by the firewall: On the machine sharing the connections, start YaST and select *Security and Users* › *Firewall*. Switch to the *Allowed Services* category. If *DHCP Server* is not already shown as *Allowed Service*, select *DHCP Server* from *Services to Allow* and click *Add*. Confirm your changes in YaST.

8. *How to provide static DNS information with automatic (DHCP, PPP, VPN) addresses?*

   In case a DHCP server provides invalid DNS information (and/or routes), you can override it. Start the dialog for configuring network connections as described in *Section 31.3, "Configuring network connections"*. Choose the connection you want to modify and click *Edit*. Switch to the *IPv4 Settings* tab, and from the *Method* drop-down box, activate *Automatic (DHCP) addresses only*. Enter the DNS information in the *DNS Servers* and *Search Domains* fields. To *Ignore automatically obtained routes* click *Routes* and activate the respective check box. Confirm your changes.

9. *How to make NetworkManager connect to password protected networks before a user logs in?*

   Define a `system connection` that can be used for such purposes. For more information, refer to *Section 31.4.1, "User and system connections"*.

31.6 Troubleshooting

Connection problems can occur. Common problems related to NetworkManager include the applet not starting or a missing VPN option. Methods for resolving and preventing these problems depend on the tool used.

NetworkManager desktop applet does not start

The applets starts automatically if the network is set up for NetworkManager control. If the applet does not start, check if NetworkManager is enabled in YaST as described in *Section 31.2, "Enabling or disabling NetworkManager"*. Then make sure that the NetworkManager-gnome package is also installed.

If the desktop applet is installed but is not running, start it manually with the command **`nm-applet`**.

NetworkManager applet does not include the VPN option

Support for NetworkManager, applets, and VPN for NetworkManager is distributed in separate packages. If your NetworkManager applet does not include the VPN option, check if the packages with NetworkManager support for your VPN technology are installed. For more information, see *Section 31.3.5, "NetworkManager and VPN"*.

No network connection available

If you have configured your network connection correctly and all other components for the network connection (router, etc.) are also up and running, it sometimes helps to restart the network interfaces on your computer. To do so, log in to a command line as `root` and run **`systemctl restart wickeds`**.

31.7 More information

More information about NetworkManager can be found on the following Web sites and directories:

NetworkManager project page

https://gitlab.freedesktop.org/NetworkManager/NetworkManager

Package documentation

Also check out the information in the following directories for the latest information about NetworkManager and the GNOME applet:

- `/usr/share/doc/packages/NetworkManager/`,
- `/usr/share/doc/packages/NetworkManager-gnome/`.

IV Hardware configuration

32 Setting up your system keyboard layout 457

33 Setting up sound cards 458

34 Setting up a printer 461

35 Setting up a scanner 467

36 Power management 469

37 Persistent memory 475

32 Setting up your system keyboard layout

Revision History

2022-03-29

The YaST *System Keyboard Layout* module lets you define the default keyboard layout for the system (also used for the console). Users can modify the keyboard layout in their individual X sessions, using the desktop's tools.

1. Start the YaST *System Keyboard Configuration* dialog by clicking *Hardware* › *System Keyboard Layout* in YaST. Alternatively, start the module from the command line with **`sudo yast2 keyboard`**.
2. Select the desired *Keyboard Layout* from the list.
3. Try the selected keyboard layout in the *Test* text box.
4. If the result is as expected, confirm your changes and close the dialog.
5. The result is stored in the files `/etc/vconsole.conf` (for text consoles) and `/etc/X11/xorg.conf.d/00-keyboard.conf` (for X11).
6. Advanced keyboard settings can be configured in *System* › *Sysconfig Editor* › *Hardware* › *Keyboard*. Here you can specify the keyboard rate and delay settings, and enable or disable NumLock, CapsLock, and ScrollLock. These settings are stored in `/etc/sysconfig/keyboard`.

33 Setting up sound cards

Revision History

2023-12-22

YaST detects most sound cards automatically and configures them with the appropriate values. To change the default settings, or to set up a sound card that could not be configured automatically, use the YaST sound module. There, you can also set up additional sound cards or switch their order.

Warning

If you do not know all details about the setup of your sound system, do not change its settings manually. Instead, let your sound subsystem—PipeWire or PulseAudio— configure it for you. Use dedicated desktop application to switch audio devices. As a fallback, use the **`pavucontrol`** graphical application.

To start the sound module, start YaST and click *Hardware* › *Sound*. Alternatively, start the *Sound Configuration* dialog directly by running **`yast2 sound &`** as user `root` from a command line. If the sound module is not available, install it using the **`sudo zypper install yast2-sound`** command.

PROCEDURE 33.1: CONFIGURING SOUND CARDS

If you have added a new sound card or YaST could not automatically configure an existing sound card, follow the steps below. For configuring a new sound card, you need to know your sound card vendor and model. If in doubt, refer to your sound card documentation for the required information. For a reference list of sound cards supported by ALSA with their corresponding sound modules, see https://www.alsa-project.org/main/index.php/Matrix:Main.

During configuration, you can choose between the following setup options:

Quick automatic setup

You are not required to go through any of the further configuration steps—the sound card is configured automatically. You can set the volume or any options you want to change later.

Normal setup

Allows you to adjust the output volume and play a test sound during the configuration.

Advanced setup with possibility to change options

For experts only. Allows you to customize all parameters of the sound card.

> **Important: Advanced configuration**
>
> Only use this option if you know exactly what you are doing. Otherwise leave the parameters untouched and use the normal or the automatic setup options.

1. Start the YaST sound module.
2. To configure a detected, but *Not Configured* sound card, select the respective entry from the list and click *Edit*.
   To configure a new sound card, click *Add*. Select your sound card vendor and model and click *Next*.
3. Choose one of the setup options and click *Next*.
4. If you have chosen *Normal Setup*, you can now *Test* your sound configuration and make adjustments to the volume. You should start at about ten percent volume to avoid damage to your hearing or the speakers.
5. If all options are set according to your wishes, click *Next*.
   The *Sound Configuration* dialog shows the newly configured or modified sound card.
6. To remove a sound card configuration that you no longer need, select the respective entry and click *Delete*.
7. Click *OK* to save the changes and leave the YaST sound module.

PROCEDURE 33.2: MODIFYING SOUND CARD CONFIGURATIONS

1. To change the configuration of an individual sound card (for experts only!), select the sound card entry in the *Sound Configuration* dialog and click *Edit*.
   This takes you to the *Sound Card Advanced Options* where you can fine-tune several parameters. For more information, click *Help*.

2. To adjust the volume of an already configured sound card or to test the sound card, select the sound card entry in the *Sound Configuration* dialog and click *Other*. Select the respective menu item.

   ### Note: YaST mixer

   The YaST mixer settings provide only basic options. They are intended for troubleshooting (for example, if the test sound is not audible). Access the YaST mixer settings from *Other* › *Volume*. For everyday use and fine-tuning of sound options, use the mixer applet provided by your desktop or the **`alsasound`** command line tool.

3. For playback of MIDI files, select *Other* › *Start Sequencer*.

4. When a supported sound card is detected, you can install SoundFonts for playback of MIDI files:

   a. Insert the original driver CD-ROM into your CD or DVD drive.

   b. Select *Other* › *Install SoundFonts* to copy SF2 SoundFonts™ to your hard disk. The SoundFonts are saved in the directory `/usr/share/sfbank/creative/`.

5. If you have configured more than one sound card in your system you can adjust the order of your sound cards. To set a sound card as primary device, select the sound card in the *Sound Configuration* and click *Other* › *Set as the Primary Card*. The sound device with index `0` is the default device and thus used by the system and the applications.

6. By default, SUSE Linux Enterprise Desktop uses the PulseAudio sound system. This is an abstraction layer that helps to mix multiple audio streams, bypassing any restrictions the hardware may have. To enable or disable the PulseAudio sound system, click *Other* › *PulseAudio Configuration*. If enabled, PulseAudio daemon is used to play sounds. Disable *PulseAudio Support* to use something else system-wide.

The volume and configuration of all sound cards are saved when you click *OK* and leave the YaST sound module. The mixer settings are saved to the file `/etc/asound.state`. The ALSA configuration data is appended to the end of the file `/etc/modprobe.d/sound` and written to `/etc/sysconfig/sound`.

34 Setting up a printer

Revision History

2024-05-13

YaST can be used to configure local and network printers. Further information about printing (general information, technical details, and troubleshooting) is available in *Chapter 24, Printer operation*.

In YaST, click *Hardware* › *Printer* to start the printer module. By default it opens in the *Printer Configurations* view, displaying a list of all printers that are available and configured. This is especially useful when having access to a lot of printers via the network. From here you can also *Print a Test Page* and configure printers.

Note: Starting CUPS

To use the printer connected to your machine, you must have CUPS installed and running on your system. If CUPS is not running, you are prompted to start it. In case CUPS is not started at boot time, you are also prompted to enable it (recommended).

34.1 Configuring printers

Normally USB printers are detected automatically. If that does not happen, check whether the printer is switched on and connected to the machine.

Configuring a printer is a three-step process: specify the connection type, choose a driver, and name the print queue for this setup.

Many printer models may have several available drivers. When configuring the printer, YaST defaults to those marked `recommended`. Normally, it is not necessary to change the driver. However, if you want a color printer to print only in black and white, you can use a driver that does not support color printing. If you experience performance problems with a PostScript printer when printing graphics, try switching from a PostScript to a PCL driver (provided your printer understands PCL).

If no driver for your printer is listed, try selecting a generic driver with an appropriate standard language from the list. Refer to your printer's documentation to find out what language (the set of commands controlling the printer) your printer supports. If this does not work, refer to *Section 34.1.1, "Adding drivers with YaST"* for an alternative solution.

A printer is always used through a print queue. This ensures that simultaneous jobs can be queued and processed one by one. Each print queue is assigned to a specific driver, and a printer can have multiple queues. As an example, this makes it possible to set up a second queue on a color printer that prints black and white only. Refer to *Section 24.1, "The CUPS workflow"* for more information about print queues.

PROCEDURE 34.1: ADDING A NEW PRINTER

1. Start the YaST printer module with *Hardware* › *Printer*.
2. In the *Printer Configurations* screen click *Add*.
3. If the printer is already listed under `Specify the Connection`, proceed with the next step. Otherwise, try *Detect More* or start the *Connection Wizard*.
4. In the text box under `Find and Assign a Driver` enter the vendor name and the model name and click *Search for*.
5. Choose a driver that matches your printer. It is recommended to choose the driver listed first. If no suitable driver is displayed, try the following.
   a. Check the search term.
   b. Expand the search by clicking *Find More*.
   c. Add a driver as described in *Section 34.1.1, "Adding drivers with YaST"*.
6. Specify the `Default paper size`.
7. In the *Set Arbitrary Name* field, enter a unique name for the print queue.
8. The printer is now configured with the default settings and ready to use. Click *OK* to return to the *Printer Configurations* view. The newly configured printer is now visible in the list of printers.

34.1.1 Adding drivers with YaST

Not all printer drivers available for SUSE Linux Enterprise Desktop are installed by default. When adding a printer, if no suitable driver is available in the *Find and Assign a Driver* dialog, install a driver package containing drivers for the printer:

PROCEDURE 34.2: INSTALLING ADDITIONAL DRIVER PACKAGES

1. Start the YaST printer module with *Hardware* › *Printer*.

2. In the *Printer Configurations* screen, click *Add*.

3. In the `Find and Assign a Driver` section, click *Driver Packages*.

4. Choose one or more suitable driver packages from the list. Do *not* specify the path to a printer description file.

5. Choose *OK* and confirm the package installation.

6. To directly use these drivers, proceed as described in *Procedure 34.1, "Adding a new printer"*.

PostScript printers do not need printer driver software. PostScript printers need only a PostScript Printer Description (PPD) file which matches the particular model. PPD files are provided by the printer manufacturer.

If no suitable PPD file is available in the *Find and Assign a Driver* dialog when adding a PostScript printer, install a PPD file for your printer:

There are several sources of PPD files. It is recommended to first try additional driver packages that are shipped with SUSE Linux Enterprise Desktop but not installed by default (see below for installation instructions). If these packages do not contain suitable drivers for your printer, get PPD files directly from your printer vendor or from the driver CD of a PostScript printer. For details, see *Section 24.8.2, "No suitable PPD file available for a PostScript printer"*. Alternatively, find PPD files at https://www.openprinting.org/printers, the "OpenPrinting.org printer database". When using PPD files from OpenPrinting, keep in mind that they may not be supported by SUSE Linux Enterprise Desktop.

PROCEDURE 34.3: ADDING A PPD FILE FOR POSTSCRIPT PRINTERS

1. Start the YaST printer module with *Hardware* › *Printer*.

2. In the *Printer Configurations* screen, click *Add*.

3. In the `Find and Assign a Driver` section, click *Driver Packages*.

4. Enter the full path to the PPD file into the text box under `Make a Printer Description File Available`.

5. Click *OK* to return to the `Add New Printer Configuration` screen.

6. To directly use this PPD file, proceed as described in *Procedure 34.1, "Adding a new printer"*.

34.1.2 Editing a local printer configuration

By editing an existing configuration for a printer you can change basic settings such as connection type and driver. It is also possible to adjust the default settings for paper size, resolution, media source, etc. You can change identifiers of the printer by altering the printer description or location.

1. Start the YaST printer module with *Hardware* › *Printer*.
2. In the *Printer Configurations* screen, choose a local printer configuration from the list and click *Edit*.
3. Change the connection type or the driver as described in *Procedure 34.1, "Adding a new printer"*. This should only be necessary in case you have problems with the current configuration.
4. Optionally, make this printer the default by checking *Default Printer*.
5. Adjust default settings by clicking *All Options for the Current Driver*. To change a setting, expand the list of options by clicking the + sign. Change the default by clicking an option. Apply your changes with *OK*.

34.2 Configuring printing via the network with YaST

Network printers are not detected automatically. They must be configured manually using the YaST printer module. Depending on your network setup, you can print to a print server (CUPS, LPD, SMB or IPX) or directly to a network printer (preferably via TCP). Access the configuration view for network printing by choosing *Printing via Network* from the left pane in the YaST printer module.

34.2.1 Using CUPS

In a Linux environment CUPS is used to print via the network. The simplest setup is to only print via a single CUPS server which can directly be accessed by all clients. Printing via more than one CUPS server requires a running local CUPS daemon that communicates with the remote CUPS servers.

Important: Browsing network print queues

CUPS servers announce their print queues over the network either via the traditional CUPS browsing protocol or via Bonjour/DNS-SD. Clients need to browse these lists so users can select specific printers to send their print jobs to. To browse network print queues, the service `cups-browsed` provided by the package `cups-filters-cups-browsed` must run on all clients that print via CUPS servers. `cups-browsed` is started automatically when configuring network printing with YaST.

If browsing does not work after starting `cups-browsed`, the CUPS servers may announce the network print queues via Bonjour/DNS-SD. In this case you need to additionally install the package `avahi` and start the associated service with **`sudo systemctl start avahi-daemon`** on all clients.

PROCEDURE 34.4: PRINTING VIA A SINGLE CUPS SERVER

1. Start the YaST printer module with *Hardware* › *Printer*.
2. From the left pane, select *Print via Network*.
3. Check *Do All Your Printing Directly via One Single CUPS Server* and specify the name or IP address of the server.
4. Click *Test Server* to make sure you have chosen the correct name or IP address.
5. Click *OK* to return to the *Printer Configurations* screen. All printers available via the CUPS server are now listed.

PROCEDURE 34.5: PRINTING VIA MULTIPLE CUPS SERVERS

1. Start the YaST printer module with *Hardware* › *Printer*.
2. From the left pane, select *Print via Network*.
3. Check *Accept Printer Announcements from CUPS Servers*.
4. Under `General Settings` specify which servers to use. You may accept connections from all networks available or from specific hosts. If you choose the latter option, you need to specify the host names or IP addresses.
5. Click *OK* and then *Yes* when prompted to start a local CUPS server. After the server has started, YaST returns to the *Printer Configurations* screen. Click *Refresh list* to see the printers detected so far.

34.2.2 Using print servers other than CUPS

If your network offers print services via print servers other than CUPS, start the YaST printer module with *Hardware* › *Printer* and select *Print via Network* from the left pane. Start the *Connection Wizard* and choose the appropriate *Connection Type*. Ask your network administrator for details on configuring a network printer in your environment.

34.3 Sharing printers over the network

Printers managed by a local CUPS daemon can be shared over the network, effectively turning your machine into a CUPS server. You share a printer by enabling so-called “browsing mode” in CUPS. If browsing is enabled, the local print queues are made available on the network for listening to remote CUPS daemons. It is also possible to set up a dedicated CUPS server that manages all print queues and can be accessed by remote clients directly. In this case, enabling browsing is not required.

PROCEDURE 34.6: SHARING PRINTERS

1. Start the YaST printer module with *Hardware* › *Printer*.
2. Select *Share Printers* from the left pane.
3. Select *Allow Remote Access*. Also check *For computers within the local network* and enable browsing mode by also checking *Publish printers by default within the local network*.
4. Click *OK* to restart the CUPS server and to return to the *Printer Configurations* screen.
5. Regarding CUPS and firewall settings, see https://en.opensuse.org/SDB:CUPS_and_SANE_Firewall_settings.

35 Setting up a scanner

Revision History

2022-03-29

You can configure a USB or SCSI scanner with YaST. The `sane-backends` package contains hardware drivers and other essentials needed to use a scanner. If you own an HP All-In-One device, see *Section 35.1, "Configuring an HP all-in-one device"*, instructions on how to configure a network scanner are available at *Section 35.3, "Scanning over the network"*.

PROCEDURE 35.1: CONFIGURING A USB OR SCSI SCANNER

1. Connect your USB or SCSI scanner to your computer and turn it on.
2. Start YaST and select *Hardware* › *Scanner*. YaST builds the scanner database and tries to detect your scanner model automatically.
   If a USB or SCSI scanner is not properly detected, try *Other* › *Restart Detection*.
3. To activate the scanner select it from the list of detected scanners and click *Edit*.
4. Choose your model form the list and click *Next* and *Finish*.
5. Use *Other* › *Test* to make sure you have chosen the correct driver.
6. Leave the configuration screen with *OK*.

35.1 Configuring an HP all-in-one device

An HP All-In-One device can be configured with YaST even if it is made available via the network. If you own a USB HP All-In-One device, start configuring as described in *Procedure 35.1, "Configuring a USB or SCSI scanner"*. If it is detected properly and the *Test* succeeds, it is ready to use.

If your USB device is not properly detected, or your HP All-In-One device is connected to the network, run the HP Device Manager:

1. Start YaST and select *Hardware* › *Scanner*. YaST loads the scanner database.
2. Start the HP Device Manager with *Other* › *Run hp-setup* and follow the on-screen instructions. After having finished the HP Device Manager, the YaST scanner module automatically restarts the auto detection.
3. Test it by choosing *Other* › *Test*.

4. Leave the configuration screen with *OK*.

35.2 Sharing a scanner over the network

SUSE Linux Enterprise Desktop allows the sharing of a scanner over the network. To do so, configure your scanner as follows:

1. Configure the scanner as described in *Chapter 35, Setting up a scanner*.
2. Choose *Other* › *Scanning via Network*.
3. Enter the host names of the clients (separated by a comma) that should be allowed to use the scanner under *Server Settings* › *Permitted Clients for saned* and leave the configuration dialog with *OK*.

35.3 Scanning over the network

To use a scanner that is shared over the network, proceed as follows:

1. Start YaST and select *Hardware* › *Scanner*.
2. Open the network scanner configuration menu by *Other* › *Scanning via Network*.
3. Enter the host name of the machine the scanner is connected to under *Client Settings* › *Servers Used for the net Metadriver*
4. Leave with *OK*. The network scanner is now listed in the Scanner Configuration window and is ready to use.

36 Power management

Revision History

2024-05-13

Power management is especially important on laptop computers, but is also useful on other systems. ACPI (Advanced Configuration and Power Interface) is available on all modern computers (laptops, desktops and servers). Power management technologies require suitable hardware and BIOS routines. Most laptops and many modern desktops and servers meet these requirements. It is also possible to control CPU frequency scaling to save power or decrease noise.

36.1 Power saving functions

Power saving functions are not only significant for the mobile use of laptops, but also for desktop systems. The main functions and their use in ACPI are:

Standby

Not supported.

Suspend (to memory)

This mode writes the entire system state to the RAM. Subsequently, the entire system except the RAM is put to sleep. In this state, the computer consumes little power. The advantage of this state is the possibility of resuming work at the same point within a few seconds without having to boot and restart applications. This function corresponds to the ACPI state S3.

Hibernation (suspend to disk)

In this operating mode, the entire system state is written to the hard disk and the system is powered off. There must be a swap partition at least as big as the RAM to write all the active data. Reactivation from this state takes about 30 to 90 seconds. The state before the suspend is restored. Several manufacturers offer useful hybrid variants of this mode, such as RediSafe in IBM Thinkpads. The corresponding ACPI state is S4. In Linux, suspend to disk is performed by kernel routines that are independent from ACPI.

Note: Changed UUID for swap partitions when formatting via **mkswap**

Do not reformat existing swap partitions with **mkswap** if possible. Reformatting with **mkswap** will change the UUID value of the swap partition. Either reformat via YaST (which will update /etc/fstab) or adjust /etc/fstab manually.

Battery monitor

ACPI checks the battery charge status and provides information about it. Additionally, it coordinates actions to perform when a critical charge status is reached.

Automatic power-off

Following a shutdown, the computer is powered off. This is especially important when an automatic shutdown is performed shortly before the battery is empty.

Processor speed control

In connection with the CPU, energy can be saved in three different ways: frequency and voltage scaling (also known as PowerNow! or Speedstep), throttling and putting the processor to sleep (C-states). Depending on the operating mode of the computer, these methods can also be combined.

36.2 Advanced configuration and power interface (ACPI)

ACPI was designed to enable the operating system to set up and control the individual hardware components. ACPI supersedes both Power Management Plug and Play (PnP) and Advanced Power Management (APM). It delivers information about the battery, AC adapter, temperature, fan and system events, like “close lid” or “battery low.”

The BIOS provides tables containing information about the individual components and hardware access methods. The operating system uses this information for tasks like assigning interrupts or activating and deactivating components. Because the operating system executes commands stored into the BIOS, the functionality depends on the BIOS implementation. The tables ACPI can detect and load are reported in journald. See *Chapter 21, **journalctl**: query the systemd journal* for more information on viewing the journal log messages. See *Section 36.2.2, “Troubleshooting”* for more information about troubleshooting ACPI problems.

36.2.1 Controlling the CPU performance

The CPU can save energy in three ways:

- Frequency and Voltage Scaling
- Throttling the Clock Frequency (T-states)
- Putting the Processor to Sleep (C-states)

Depending on the operating mode of the computer, these methods can be combined. Saving energy also means that the system heats up less and the fans are activated less frequently.

Frequency scaling and throttling are only relevant if the processor is busy, because the most economic C-state is applied anyway when the processor is idle. If the CPU is busy, frequency scaling is the recommended power saving method. Often the processor only works with a partial load. In this case, it can be run with a lower frequency. Dynamic frequency scaling controlled by the kernel on-demand governor is the best approach.

Throttling should be used as the last resort, for example, to extend the battery operation time despite a high system load. However, certain systems do not run smoothly when they are throttled too much. Moreover, CPU throttling does not make sense if the CPU has little to do.

For in-depth information, refer to *Book "System Analysis and Tuning Guide", Chapter 12 "Power management"*.

36.2.2 Troubleshooting

There are two different types of problems. On one hand, the ACPI code of the kernel may contain bugs that were not detected in time. In this case, a solution will be made available for download. More often, the problems are caused by the BIOS. Sometimes, deviations from the ACPI specification are purposely integrated in the BIOS to circumvent errors in the ACPI implementation of other widespread operating systems. Hardware components that have serious errors in the ACPI implementation are recorded in a blacklist that prevents the Linux kernel from using ACPI for these components.

The first thing to do when problems are encountered is to update the BIOS. If the computer does not boot, one of the following boot parameters may be helpful:

pci=noacpi

Do not use ACPI for configuring the PCI devices.

acpi=ht

Only perform a simple resource configuration. Do not use ACPI for other purposes.

acpi=off

Disable ACPI.

Warning: Problems booting without ACPI

Certain newer machines (especially SMP systems and AMD64 systems) need ACPI for configuring the hardware correctly. On these machines, disabling ACPI can cause problems.

Sometimes, the machine is confused by hardware that is attached over USB or FireWire. If a machine refuses to boot, unplug all unneeded hardware and try again.

Monitor the boot messages of the system with the command `dmesg -T | grep -2i acpi` (or all messages, because the problem may not be caused by ACPI) after booting. If an error occurs while parsing an ACPI table, the most important table—the DSDT (*Differentiated System Description Table*)—can be replaced with an improved version. In this case, the faulty DSDT of the BIOS is ignored. The procedure is described in *Section 36.4, "Troubleshooting"*.

In the kernel configuration, there is a switch for activating ACPI debug messages. If a kernel with ACPI debugging is compiled and installed, detailed information is issued.

If you experience BIOS or hardware problems, it is always advisable to contact the manufacturers. Especially if they do not always provide assistance for Linux, they should be confronted with the problems. Manufacturers only take the issue seriously if they realize that an adequate number of their customers use Linux.

36.2.2.1 More information

- https://tldp.org/HOWTO/ACPI-HOWTO/ (detailed ACPI HOWTO, contains DSDT patches)
- https://uefi.org/specifications (Advanced Configuration & Power Interface Specification)

36.3 Rest for the hard disk

In Linux, the hard disk can be put to sleep entirely if it is not needed or it can be run in a more economic or quieter mode. On modern laptops, you do not need to switch off the hard disks manually, because they automatically enter an economic operating mode whenever they are not needed. However, if you want to maximize power savings, test the following methods, using the **`hdparm`** command.

It can be used to modify hard disk settings. The option `-y` instantly switches the hard disk to the standby mode. `-Y` puts it to sleep. **`hdparm`** `-S` *`X`* causes the hard disk to be spun down after a certain period of inactivity. Replace *`X`* as follows: `0` disables this mechanism, causing the hard disk to run continuously. Values from `1` to `240` are multiplied by 5 seconds. Values from `241` to `251` correspond to 1 to 11 times 30 minutes.

Internal power saving options of the hard disk can be controlled with the option `-B`. Select a value from `0` to `255` for maximum saving to maximum throughput. The result depends on the hard disk used and is difficult to assess. To make a hard disk quieter, use the option `-M`. Select a value from `128` to `254` for quiet to fast.

Often, it is not so easy to put the hard disk to sleep. In Linux, multiple processes write to the hard disk, waking it up repeatedly. Therefore, it is important to understand how Linux handles data that needs to be written to the hard disk. First, all data is buffered in the RAM. This buffer is monitored by the `pdflush` daemon. When the data reaches a certain age limit or when the buffer is filled to a certain degree, the buffer content is flushed to the hard disk. The buffer size is dynamic and depends on the size of the memory and the system load. By default, pdflush is set to short intervals to achieve maximum data integrity. It checks the buffer every 5 seconds and writes the data to the hard disk. The following variables are interesting:

`/proc/sys/vm/dirty_writeback_centisecs`
: Contains the delay until a pdflush thread wakes up (in hundredths of a second).

`/proc/sys/vm/dirty_expire_centisecs`
: Defines after which timeframe a dirty page should be written at latest. Default is `3000`, which means 30 seconds.

`/proc/sys/vm/dirty_background_ratio`
: Maximum percentage of dirty pages until pdflush begins to write them. Default is `5`%.

`/proc/sys/vm/dirty_ratio`
: When the dirty pages exceed this percentage of the total memory, processes are forced to write dirty buffers during their time slice instead of continuing to write.

Warning: Data integrity risk

Changes to the `pdflush` daemon settings can compromise data integrity.

Apart from these processes, journaling file systems, like `Btrfs`, `Ext3`, `Ext4` and others write their metadata independently from `pdflush`, which also prevents the hard disk from spinning down. To avoid this, a special kernel extension has been developed for mobile devices. To use the extension, install the `laptop-mode-tools` package and see `/usr/src/linux/Documentation/laptops/laptop-mode.txt` for details.

Another important factor is the way active programs behave. For example, good editors regularly write hidden backups of the currently modified file to the hard disk, causing the disk to wake up. Features like this can be disabled at the expense of data integrity.

In this connection, the mail daemon postfix uses the variable `POSTFIX_LAPTOP`. If this variable is set to `yes`, postfix accesses the hard disk far less frequently.

In SUSE Linux Enterprise Desktop these technologies are controlled by `laptop-mode-tools`.

36.4 Troubleshooting

All error messages and alerts are logged in the system journal, which can be queried with the command **`journalctl`** (see *Chapter 21, **`journalctl`**: query the `systemd` journal* for more information). The following sections cover the most common problems.

36.4.1 CPU frequency does not work

Refer to the kernel sources to see if your processor is supported. You may need a special kernel module or module option to activate CPU frequency control. If the `kernel-source` package is installed, this information is available in `/usr/src/linux/Documentation/cpu-freq/*`.

37 Persistent memory

Revision History

2024-05-13

This chapter contains additional information about using SUSE Linux Enterprise Desktop with non-volatile main memory, also known as *Persistent Memory*, comprising one or more NVDIMMs.

37.1 Introduction

Persistent memory is a new type of computer storage, combining speeds approaching those of dynamic RAM (DRAM) along with RAM's byte-by-byte addressability, plus the permanence of solid-state drives (SSDs).

SUSE currently supports the use of persistent memory with SUSE Linux Enterprise Desktop on machines with the AMD64/Intel 64 and POWER architectures.

Like conventional RAM, persistent memory is installed directly into mainboard memory slots. As such, it is supplied in the same physical form factor as RAM—as DIMMs. These are known as NVDIMMs: non-volatile dual inline memory modules.

Unlike RAM, though, persistent memory is also similar to flash-based SSDs in several ways. Both are based on forms of solid-state memory circuitry, but despite this, both provide non-volatile storage: Their contents are retained when the system is powered off or restarted. For both forms of medium, writing data is slower than reading it, and both support a limited number of rewrite cycles. Finally, also like SSDs, sector-level access to persistent memory is possible if that is more suitable for a particular application.

Different models use different forms of electronic storage medium, such as Intel 3D XPoint, or a combination of NAND-flash and DRAM. New forms of non-volatile RAM are also in development. This means that different vendors and models of NVDIMM offer different performance and durability characteristics.

Because the storage technologies involved are in an early stage of development, different vendors' hardware may impose different limitations. Thus, the following statements are generalizations.

Persistent memory is up to ten times slower than DRAM, but around a thousand times faster than flash storage. It can be rewritten on a byte-by-byte basis rather than flash memory's whole-sector erase-and-rewrite process. Finally, while rewrite cycles are limited, most forms of persistent memory can handle millions of rewrites, compared to the thousands of cycles of flash storage.

This has two important consequences:

- It is not possible with current technology to run a system with only persistent memory and thus achieve non-volatile main memory. You must use a mixture of both conventional RAM and NVDIMMs. The operating system and applications will execute in conventional RAM, with the NVDIMMs providing fast supplementary storage.
- The performance characteristics of different vendors' persistent memory mean that it may be necessary for programmers to be aware of the hardware specifications of the NVDIMMs in a particular server, including how many NVDIMMs there are and in which memory slots they are fitted. This will impact hypervisor use, migration of software between different host machines, and so on.

This new storage subsystem is defined in version 6 of the ACPI standard. However, `libnvdimm` supports pre-standard NVDIMMs and they can be used in the same way.

Tip: Intel Optane DC Persistent Memory

Intel Optane DIMMs memory can be used in specific modes:

- In *App Direct Mode*, the Intel Optane memory is used as fast persistent storage, an alternative to SSDs and NVMe devices. Data in this mode is kept when the system is powered off.
  App Direct Mode has been supported since SLES 12 SP4.
- In *Memory Mode*, the Intel Optane memory serves as a cost-effective, high-capacity alternative to DRAM. In this mode, separate DRAM DIMMs act as a cache for the most frequently accessed data while the Optane DIMMs memory provides large memory capacity. However, compared with DRAM-only systems, this mode is slower under random access workloads. If you run applications without Optane-specific enhancements that take advantage of this mode, memory performance may decrease. Data in this mode is lost when the system is powered off.

Memory Mode has been supported since SLES 15 SP1.

- In *Mixed Mode*, the Intel Optane memory is partitioned, so it can serve in both modes simultaneously.
  Mixed Mode has been supported since SLES 15 SP1.

37.2 Terms

Region

A *region* is a block of persistent memory that can be divided up into one or more *namespaces*. You cannot access the persistent memory of a region without first allocating it to a namespace.

Namespace

A single contiguously addressed range of non-volatile storage, comparable to NVM Express SSD namespaces, or to SCSI Logical Units (LUNs). Namespaces appear in the server's `/dev` directory as separate block devices. Depending on the method of access required, namespaces can either amalgamate storage from multiple NVDIMMs into larger volumes, or allow it to be partitioned into smaller volumes.

Mode

Each namespace also has a *mode* that defines which NVDIMM features are enabled for that namespace. Sibling namespaces of the same parent region always have the same type, but might be configured to have different modes. Namespace modes include:

devdax

Device-DAX mode. Creates a single-character device file (`/dev/daxX.Y`). Does *not* require file system creation.

fsdax

File system-DAX mode. Default if no other mode is specified. Creates a block device (`/dev/pmemX [.Y]`) which supports DAX for `ext4` or `XFS`.

sector

For legacy file systems which do not checksum metadata. Suitable for small boot volumes. Compatible with other operating systems.

raw

A memory disk without a label or metadata. Does not support DAX. Compatible with other operating systems.

> **Note**
>
> `raw` mode is not supported by SUSE. It is not possible to mount file systems on `raw` namespaces.

Type

Each namespace and region has a *type* that defines how the persistent memory associated with that namespace or region can be accessed. A namespace always has the same type as its parent region. There are two different types: Persistent Memory, which can be configured in two different ways, and the deprecated Block Mode.

Persistent memory (PMEM)

PMEM storage offers byte-level access, similar to RAM. Using PMEM, a single namespace can include multiple interleaved NVDIMMs, allowing them all to be used as a single device.

There are two ways to configure a PMEM namespace.

PMEM with DAX

A PMEM namespace configured for Direct Access (DAX) means that accessing the memory bypasses the kernel's page cache and goes direct to the medium. Software can directly read or write every byte of the namespace separately.

PMEM with block translation table (BTT)

A PMEM namespace configured to operate in BTT mode is accessed on a sector-by-sector basis, like a conventional disk drive, rather than the more RAM-like byte-addressable model. A translation table mechanism batches accesses into sector-sized units.

The advantage of BTT is data protection. The storage subsystem ensures that each sector is completely written to the underlying medium. If a sector cannot be completely written (that is, if the write operation fails for some reason), then the whole sector will be rolled back to its previous state. Thus a given sector cannot be partially written.

Additionally, access to BTT namespaces is cached by the kernel.

The drawback is that DAX is not possible for BTT namespaces.

Block mode (BLK)

Block mode storage addresses each NVDIMM as a separate device. Its use is deprecated and no longer supported.

Apart from `devdax` namespaces, all other types must be formatted with a file system, just as with a conventional drive. SUSE Linux Enterprise Desktop supports the `ext2`, `ext4` and `XFS` file systems for this.

Direct access (DAX)

DAX allows persistent memory to be directly mapped into a process's address space, for example, using the `mmap` system call.

DIMM physical address (DPA)

A memory address as an offset into a single DIMM's memory; that is, starting from zero as the lowest addressable byte on that DIMM.

Label

Metadata stored on the NVDIMM, such as namespace definitions. This can be accessed using DSMs.

Device-specific method (DSM)

ACPI method to access the firmware on an NVDIMM.

37.3 Use cases

37.3.1 PMEM with DAX

This form of memory access is *not* transactional. In the event of a power outage or other system failure, data may not be written into storage. PMEM storage is only suitable if the application can handle the situation of partially written data.

37.3.1.1 Applications that benefit from large amounts of byte-addressable storage

If the server will host an application that can directly use large amounts of fast storage on a byte-by-byte basis, the programmer can use the `mmap` system call to place blocks of persistent memory directly into the application's address space, without using any additional system RAM.

37.3.1.2 Avoiding use of the kernel page cache

Avoid using the kernel page cache to conserve the use of RAM for the page cache, and instead give it to your applications. For instance, non-volatile memory could be dedicated to holding virtual machine (VM) images. As these would not be cached, this would reduce the cache usage on the host, allowing more VMs per host.

37.3.2 PMEM with BTT

This is useful when you want to use the persistent memory on a set of NVDIMMs as a disk-like pool of fast storage. For example, placing the file system journal on PMEM with BTT increases the reliability of file system recovery after a power failure or other sudden interruption (see *Section 37.5.3, "Creating a PMEM namespace with BTT"*).

To applications, such devices appear as fast SSDs and can be used like any other storage device. For example, LVM can be layered on top of the persistent memory and will work as normal.

The advantage of BTT is that sector write atomicity is guaranteed, so even sophisticated applications that depend on data integrity will keep working. Media error reporting works through standard error-reporting channels.

37.4 Tools for managing persistent memory

To manage persistent memory, it is necessary to install the `ndctl` package. This also installs the `libndctl` package, which provides a set of user space libraries to configure NVDIMMs.

These tools work via the `libnvdimm` library, which supports three types of NVDIMM:

- PMEM
- BLK
- Simultaneous PMEM and BLK

The **`ndctl`** utility has a helpful set of **`man`** pages, accessible with the command:

```
> ndctl help subcommand
```

To see a list of available subcommands, use:

```
> ndctl --list-cmds
```

The available subcommands include:

version
: Displays the current version of the NVDIMM support tools.

enable-namespace
: Makes the specified namespace available for use.

disable-namespace
: Prevents the specified namespace from being used.

create-namespace
: Creates a new namespace from the specified storage devices.

destroy-namespace
: Removes the specified namespace.

enable-region
: Makes the specified region available for use.

disable-region
: Prevents the specified region from being used.

zero-labels
: Erases the metadata from a device.

read-labels
: Retrieves the metadata of the specified device.

list
: Displays available devices.

help
: Displays information about using the tool.

37.5 Setting up persistent memory

37.5.1 Viewing available NVDIMM storage

The **`ndctl`** `list` command can be used to list all available NVDIMMs in a system.

In the following example, the system has three NVDIMMs, which are in a single, triple-channel interleaved set.

```
# ndctl list --dimms

[
 {
  "dev":"nmem2",
  "id":"8089-00-0000-12325476"
 },
 {
  "dev":"nmem1",
  "id":"8089-00-0000-11325476"
 },
 {
  "dev":"nmem0",
  "id":"8089-00-0000-10325476"
 }
]
```

With a different parameter, **ndctl** list will also list the available regions.

Note

Regions may not appear in numerical order.

Note that although there are only three NVDIMMs, they appear as four regions.

```
# ndctl list --regions

[
 {
  "dev":"region1",
  "size":68182605824,
  "available_size":68182605824,
  "type":"blk"
 },
 {
  "dev":"region3",
  "size":202937204736,
  "available_size":202937204736,
  "type":"pmem",
  "iset_id":5903239628671731251
  },
  {
   "dev":"region0",
```

```
    "size":68182605824,
    "available_size":68182605824,
    "type":"blk"
  },
  {
    "dev":"region2",
    "size":68182605824,
    "available_size":68182605824,
    "type":"blk"
  }
]
```

The space is available in two different forms: either as three separate 64 regions of type BLK, or as one combined 189 GB region of type PMEM which presents all the space on the three interleaved NVDIMMs as a single volume.

Note that the displayed value for `available_size` is the same as that for `size`. This means that none of the space has been allocated yet.

37.5.2 Configuring the storage as a single PMEM namespace with DAX

For the first example, we will configure our three NVDIMMs into a single PMEM namespace with Direct Access (DAX).

The first step is to create a new namespace.

```
# ndctl create-namespace --type=pmem --mode=fsdax --map=memory
{
 "dev":"namespace3.0",
 "mode":"memory",
 "size":199764213760,
 "uuid":"dc8ebb84-c564-4248-9e8d-e18543c39b69",
 "blockdev":"pmem3"
}
```

This creates a block device `/dev/pmem3`, which supports DAX. The `3` in the device name is inherited from the parent region number, in this case `region3`.

The `--map=memory` option sets aside part of the PMEM storage space on the NVDIMMs so that it can be used to allocate internal kernel data structures called `struct pages`. This allows the new PMEM namespace to be used with features such as `O_DIRECT I/O` and `RDMA`.

The reservation of some persistent memory for kernel data structures is why the resulting PMEM namespace has a smaller capacity than the parent PMEM region.

Next, we verify that the new block device is available to the operating system:

```
# fdisk -l /dev/pmem3
Disk /dev/pmem3: 186 GiB, 199764213760 bytes, 390164480 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```

Before it can be used, like any other drive, it must be formatted. In this example, we format it with XFS:

```
# mkfs.xfs /dev/pmem3
meta-data=/dev/pmem3      isize=256    agcount=4, agsize=12192640 blks
         =                sectsz=4096  attr=2, projid32bit=1
         =                crc=0        finobt=0, sparse=0
data     =                bsize=4096   blocks=48770560, imaxpct=25
         =                sunit=0      swidth=0 blks
naming   =version 2       bsize=4096   ascii-ci=0 ftype=1
log      =internal log    bsize=4096   blocks=23813, version=2
         =                sectsz=4096  sunit=1 blks, lazy-count=1
realtime =none            extsz=4096   blocks=0, rtextents=0
```

Next, we can mount the new drive onto a directory:

```
# mount -o dax /dev/pmem3 /mnt/pmem3
```

Then we can verify that we now have a DAX-capable device:

```
# mount | grep dax
/dev/pmem3 on /mnt/pmem3 type xfs (rw,relatime,attr2,dax,inode64,noquota)
```

The result is that we now have a PMEM namespace formatted with the XFS file system and mounted with DAX.

Any `mmap()` calls to files in that file system will return virtual addresses that directly map to the persistent memory on our NVDIMMs, bypassing the page cache.

Any `fsync` or `msync` calls on files in that file system will still ensure that modified data has been fully written to the NVDIMMs. These calls flush the processor cache lines associated with any pages that have been modified in user space via `mmap` mappings.

37.5.2.1 Removing a namespace

Before creating any other type of volume that uses the same storage, we must unmount and then remove this PMEM volume.

First, unmount it:

```
# umount /mnt/pmem3
```

Then disable the namespace:

```
# ndctl disable-namespace namespace3.0
disabled 1 namespace
```

Then delete it:

```
# ndctl destroy-namespace namespace3.0
destroyed 1 namespace
```

37.5.3 Creating a PMEM namespace with BTT

BTT provides sector write atomicity, which makes it a good choice when you need data protection, , for Ext4 and XFS journals. If there is a power failure, the journals are protected and should be recoverable. The following examples show how to create a PMEM namespace with BTT in sector mode, and how to place the file system journal in this namespace.

```
# ndctl create-namespace --type=pmem --mode=sector
{
 "dev":"namespace3.0",
 "mode":"sector",
 "uuid":"51ab652d-7f20-44ea-b51d-5670454f8b9b",
 "sector_size":4096,
 "blockdev":"pmem3s"
}
```

Next, verify that the new device is present:

```
# fdisk -l /dev/pmem3s
Disk /dev/pmem3s: 188.8 GiB, 202738135040 bytes, 49496615 sectors
Units: sectors of 1 * 4096 = 4096 bytes
Sector size (logical/physical): 4096 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```

Like the DAX-capable PMEM namespace we previously configured, this BTT-capable PMEM namespace consumes all the available storage on the NVDIMMs.

Note

The trailing `s` in the device name (`/dev/pmem3s`) stands for `sector` and can be used to easily distinguish namespaces that are configured to use the BTT.

The volume can be formatted and mounted as in the previous example.

The PMEM namespace shown here cannot use DAX. Instead it uses the BTT to provide *sector write atomicity*. On each sector write through the PMEM block driver, the BTT will allocate a new sector to receive the new data. The BTT atomically updates its internal mapping structures after the new data is fully written so the newly written data will be available to applications. If the power fails at any point during this process, the write will be lost and the application will have access to its old data, still intact. This prevents the condition known as “torn sectors”.

This BTT-enabled PMEM namespace can be formatted and used with a file system same as any other standard block device. It cannot be used with DAX. However, `mmap` mappings for files on this block device will use the page cache.

37.5.4 Placing the file system journal on PMEM/BTT

When you place the file system journal on a separate device, it must use the same file system block size as the file system. Most likely this is 4096, and you can find the block size with this command:

```
# blockdev --getbsz /dev/sda3
```

The following example creates a new Ext4 journal on a separate NVDIMM device, creates the file system on a SATA device, then attaches the new file system to the journal:

```
# mke2fs -b 4096 -O journal_dev /dev/pmem3s
# mkfs.ext4 -J device=/dev/pmem3s /dev/sda3
```

The following example creates a new XFS file system on a SATA drive, and creates the journal on a separate NVDIMM device:

```
# mkfs.xfs -l logdev=/dev/pmem3s  /dev/sda3
```

See **`man 8 mkfs.ext4`** and **`man 8 mkfs.ext4`** for detailed information about options.

37.6 More information

More about this topic can be found in the following list:

- Persistent Memory Wiki (https://nvdimm.wiki.kernel.org/)

  Contains instructions for configuring NVDIMM systems, information about testing, and links to specifications related to NVDIMM enabling. This site is developing as NVDIMM support in Linux is developing.

- Persistent Memory Programming (https://pmem.io/)

  Information about configuring, using and programming systems with non-volatile memory under Linux and other operating systems. Covers the NVM Library (NVML), which aims to provide useful APIs for programming with persistent memory in user space.

- LIBNVDIMM: Non-Volatile Devices (https://www.kernel.org/doc/Documentation/nvdimm/nvdimm.txt)

  Aimed at kernel developers, this is part of the Documentation directory in the current Linux kernel tree. It talks about the different kernel modules involved in NVDIMM enablement, lays out technical details of the kernel implementation, and talks about the `sysfs`interface to the kernel that is used by the **`ndctl`** tool.

- GitHub: pmem/ndctl (https://github.com/pmem/ndctl)

  Utility library for managing the **`libnvdimm`** subsystem in the Linux kernel. Also contains user space libraries, as well as unit tests and documentation.

V Services

38 Service management with YaST 489

39 Time synchronization with NTP 491

38 Service management with YaST

Revision History

2022-04-27

YaST provides a service manager for controlling the default system target, services, displaying service status, and reading the log file. New in SUSE Linux Enterprise Desktop 15 SP7 is YaST support for `systemd` socket-based services activation, which configures services to start on demand.

`systemd` supports starting services with socket-based activation, for starting services on demand. These services have two unit types: service and socket. For example, CUPS is controlled by `cups.service` and `cups.socket`. YaST allows you to select the type of service start-up you want to use.

Figure 38.1, "YaST service manager" shows the options in the Start Mode drop-down box: *On Boot*, *On Demand*, and *Manually*. Select *On Demand* for socket-based activation. This opens a listening network socket, and the service starts when there is a request.

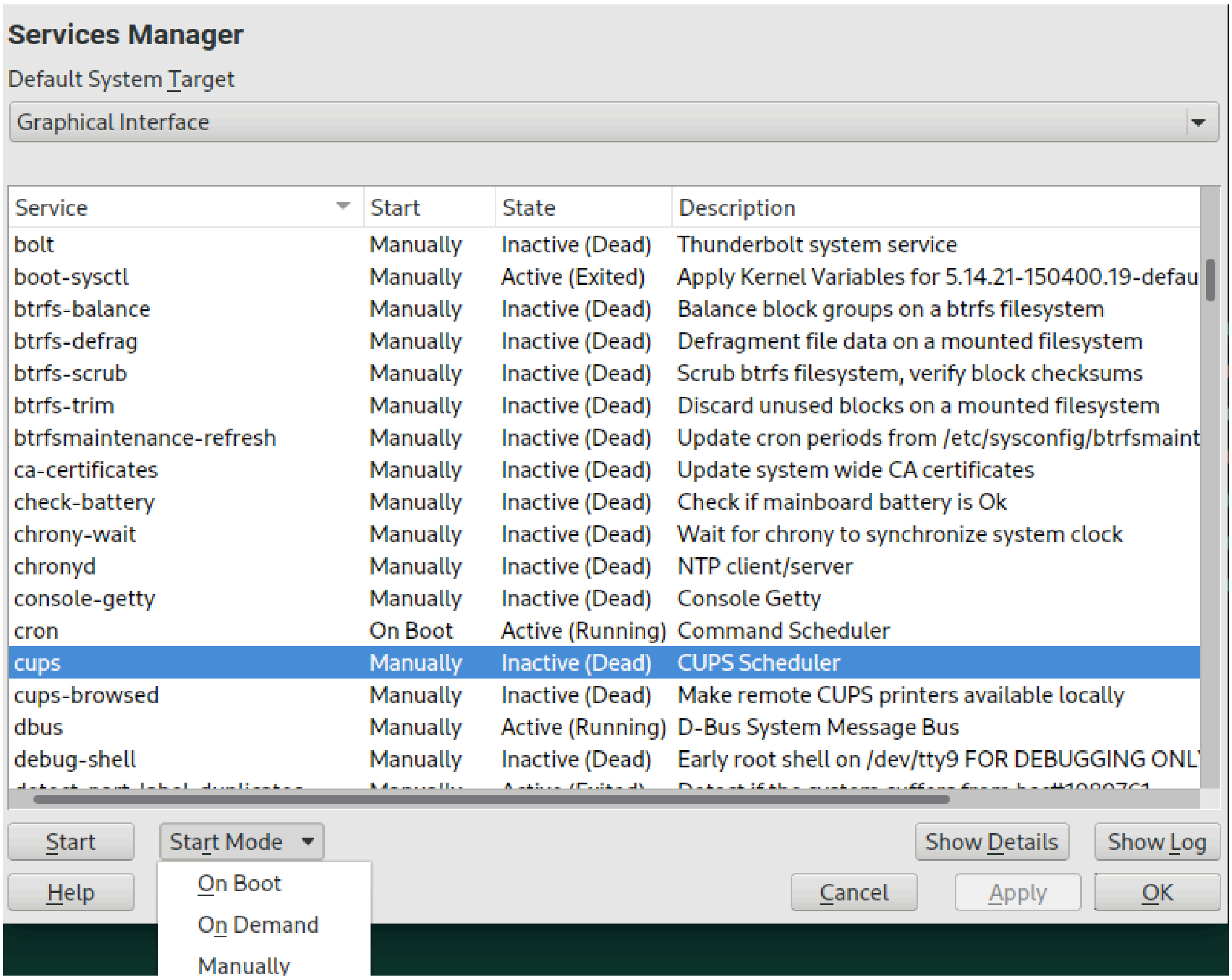


FIGURE 38.1: YAST SERVICE MANAGER

The *On Demand* option is visible only for services that support it. Currently this is a small subset of services, such as CUPS, dbus, iscsid, iscsiuio, multipathd, pcscd, rpcbind, TFTP, virtlockd, virtlogd. See **`man 5 systemd.socket`** for detailed information on how socket activation works.

39 Time synchronization with NTP

Revision History

2025-04-25

The NTP (network time protocol) mechanism is a protocol for synchronizing the system time over the network. First, a machine can obtain the time from a server that is a reliable time source. Second, a machine can itself act as a time source for other computers in the network. The goal is twofold—maintaining the absolute time and synchronizing the system time of all machines within a network.

Maintaining an exact system time is important in many situations. The built-in hardware clock does often not meet the requirements of applications such as databases or clusters. Manual correction of the system time would lead to severe problems because, for example, a backward leap can cause malfunction of critical applications. Within a network, it is usually necessary to synchronize the system time of all machines, but manual time adjustment is a bad approach. NTP provides a mechanism to solve these problems. The NTP service continuously adjusts the system time with reliable time servers in the network. It further enables the management of local reference clocks, such as radio-controlled clocks.

Since SUSE Linux Enterprise Desktop 15, `chrony` is the default implementation of NTP. `chrony` includes two parts; `chronyd` is a daemon that can be started at boot time and **`chronyc`** is a command line interface program to monitor the performance of `chronyd`, and to change operating parameters at runtime.

Starting with SUSE Linux Enterprise Desktop 15.2, the YaST module for NTP client configuration configures the systemd-timer instead of the cron daemon to execute `chrony`, when it is not configured to run as a daemon.

Note

To enable time synchronization via active directory, follow the instructions found at *Book “Security and Hardening Guide”, Chapter 7 “Active Directory support”, Section 7.3.3 “Joining Active Directory using Windows domain membership”, Joining an Active Directory domain using Windows domain membership*.

39.1 Configuring an NTP client with YaST

The NTP daemon (`chronyd`) coming with the `chrony` package is preset to use the local computer hardware clock as a time reference. The precision of the hardware clock heavily depends on its time source. For example, an atomic clock or GPS receiver is a precise time source, while a common RTC chip is not a reliable time source. YaST simplifies the configuration of an NTP client.

In the YaST NTP client configuration (*Network Services* › *NTP Configuration*) window, you can specify when to start the NTP daemon, the type of the configuration source, and add custom time servers.

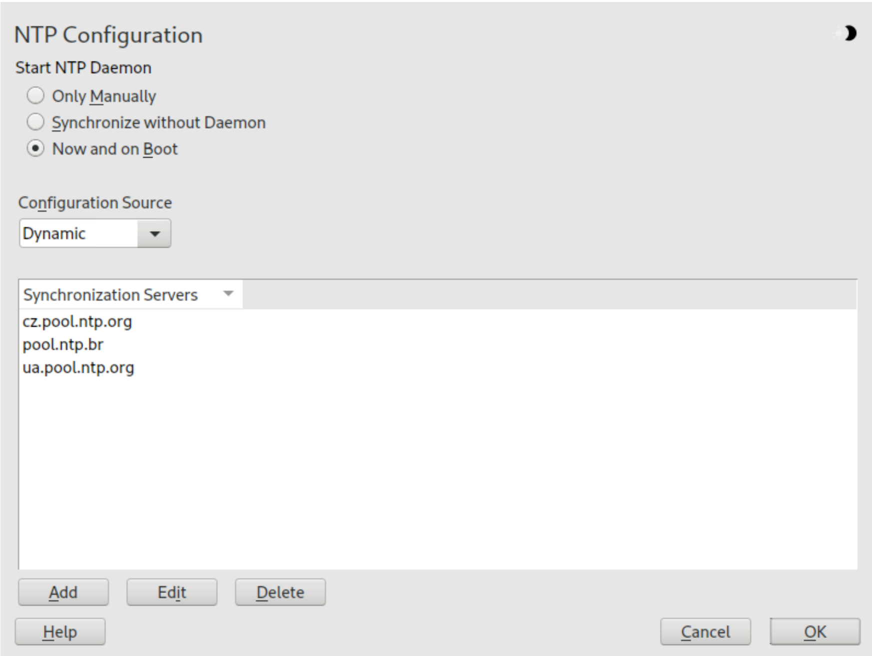


FIGURE 39.1: NTP CONFIGURATION WINDOW

39.1.1 NTP daemon start

You can choose from three options for when to start the NTP daemon:

Only manually

Select *Only Manually* to manually start the `chrony` daemon.

Synchronize without daemon

Select *Synchronize without Daemon* to set the system time periodically without a permanently running `chrony`. You can set the *Interval of the Synchronization in Minutes*.

Now and on boot

Select *Now and On Boot* to start `chronyd` automatically when the system is booted. This setting is recommended.

39.1.2 Type of the configuration source

In the *Configuration Source* drop-down box, select either *Dynamic* or *Static*. Set *Static* if your server uses only a fixed set of (public) NTP servers, while *Dynamic* is better if your internal network offers NTP servers via DHCP.

39.1.3 Configure time servers

Time servers for the client to query are listed in the lower part of the *NTP Configuration* window. Modify this list as needed with *Add*, *Edit*, and *Delete*.

Click *Add* to add a new time server:

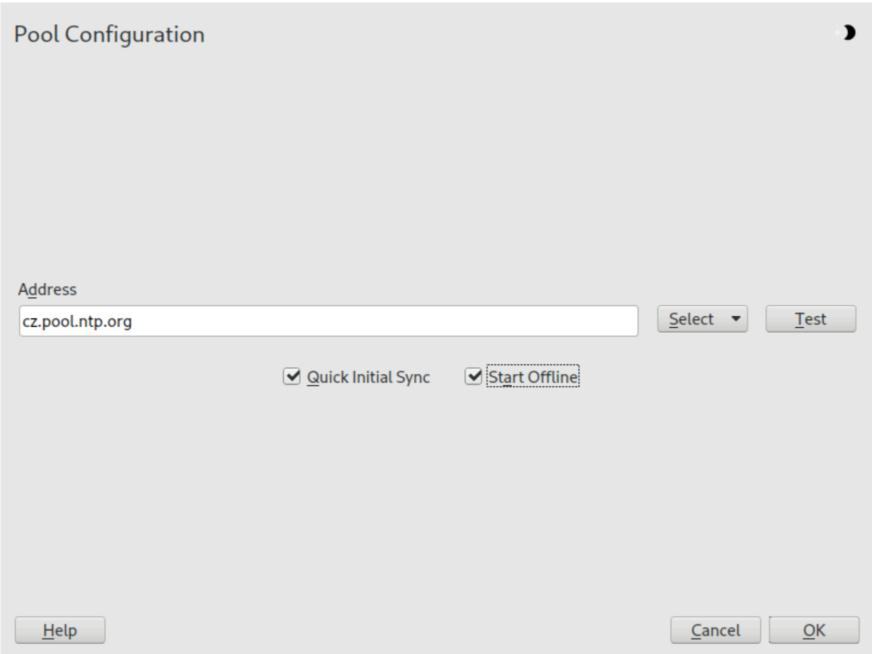


FIGURE 39.2: ADDING A TIME SERVER

1. In the *Address* field, type the URL of the time server or pool of time servers with which you want to synchronize the machine time. After the URL is complete, click *Test* to verify that it points to a valid time source.

2. Activate *Quick Initial Sync* to speed up the time synchronization by sending more requests at the `chronyd` daemon start.
3. Activate *Start Offline* to speed up the boot time on systems that start the `chronyd` daemon automatically and may not have an Internet connection at boot time. This option is useful, for example, for laptops with network connections managed by NetworkManager.
4. Confirm with *OK*.

39.2 Manually configuring NTP in the network

`chrony` reads its configuration from the `/etc/chrony.conf` file. To keep the computer clock synchronized, you need to tell `chrony` what time servers to use. You can use specific server names or IP addresses, for example:

```
0.suse.pool.ntp.org
1.suse.pool.ntp.org
2.suse.pool.ntp.org
3.suse.pool.ntp.org
```

You can also specify a *pool* name. Pool name resolves to several IP addresses:

```
pool pool.ntp.org
```

Tip: Computers on the same network

To synchronize time on multiple computers on the same network, we do not recommend to synchronize them all with an external server. A good practice is to make one computer the time server which is synchronized with an external time server, and the other computers act as its clients. Add a `local` directive to the server's `/etc/chrony.conf` to distinguish it from an authoritative time server:

```
local stratum 10
```

To start `chrony`, run:

```
systemctl start chronyd.service
```

After initializing `chronyd`, it takes a while before the time is stabilized and the drift file for correcting the local computer clock is created. With the drift file, the systematic error of the hardware clock can be computed when the computer is powered on. The correction is used immediately, resulting in a higher stability of the system time.

To enable the service so that `chrony` starts automatically at boot time, run:

```
systemctl enable chronyd.service
```

Warning: Conflicting `yast-timesync.service` service

Besides the `chronyd.service` service, SLED includes `yast-timesync.service`. `yast-timesync.service` is triggered by a timer every 5 minutes and runs `chronyd` with the `-q` option to set the system time and exit. Because only one instance of `chronyd` can be running at any given time, do not enable or start both `chronyd`-related services at the same time.

39.3 Configuring NTS

Network Time Protocol (NTP) is a protocol used to synchronize and keep accurate the system time of one or more hosts in the network. This article describes how to secure NTP using Network Time Security (NTS).

NTP protocol does not introduce any security mechanism to make the communication between the time server and client authenticated and encrypted. Network Time Security (NTS) is an extension that improves the security of NTP. `chrony` supports NTS and can authenticate time sources and protect against certain network attacks.

The following procedures outline how to configure the time server and client machine for secure time synchronization.

PROCEDURE 39.1: CONFIGURING NTS TIME SERVER

1. *(Optional)* It is a good idea to configure the time server to update its time via NTS. This ensures secure time synchronization from the very beginning of the synchronization chain. Comment out any existing time sources in `/etc/chrony.conf` that do not support NTS and add at least one that supports NTS, for example:

   ```
   server time.cloudflare.com iburst nts
   ```

Tip

The `nts` option requests NTS connection if it is available, otherwise it falls back to NTP if NTS is not available.

2. Restart the `chronyd` service.

```
> sudo systemctl restart chronyd.srvice
```

3. Verify the configured time sources.

```
> chronyc sources -v
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^? time.cloudflare.com           3   6     1     2   -947ms[ -947ms] +/-   12ms
^? pyrrha.fi.muni.cz             2   6     1     1   -948ms[ -948ms] +/-   39ms
^* whitesoft-intex16.c.cbsn>     1   6     1     2   -948ms[ -948ms] +/- 5444us
^? mail.combatostrich.dev        2   6     1     1   -948ms[ -948ms] +/-   28ms
```

Note

The line that starts with `^*` includes the time source that was selected as best.

Verify that the configured time source uses the NTS mode.

```
> chronyc -N authdata
Name/IP address             Mode KeyID Type KLen Last Atmp  NAK Cook CLen
=========================================================================
[...]
time.cloudflare.com          NTS     1   15  256    3    0    0    8   96
```

4. Verify that the server configuration includes the `allow` option that specifies which clients can synchronize time with the time server, for example:

```
allow 192.168.1.0/24
```

5. *(Optional)* If the time server is running behind a firewall, allow communication on ports for both NTP and NTS. They are 123 and 4460 by default.

6. Obtain a TLS certificate and a corresponding private key and copy them to `/var/lib/chrony/`. Verify that they are readable by `chrony`, for example:

```
> sudo install -m 0440 -o chrony -g chrony nts.key /var/lib/chrony/
> sudo install -m 0440 -o chrony -g chrony nts.crt /var/lib/chrony/
```

Tip

Find detailed information about TLS certificates in a dedicated article (https://documentation.suse.com/smart/security/html/tls-certificates/index.html).

7. Edit `/etc/chrony.conf` and verify that the `ntsdumpdir /var/lib/chrony` option is active. Then append the paths to the TLS key and certificate.

```
ntsdumpdir /var/lib/chrony
ntsserverkey /var/lib/chrony/nts.key
ntsservercert /var/lib/chrony/nts.crt
```

8. Restart the `chronyd` service.

```
> sudo systemctl restart chronyd.service
```

PROCEDURE 39.2: CONFIGURING NTS CLIENTS

1. Disable existing NTP sources, for example:

```
#server 192.168.1.1 iburst
```

The sources configurations are included in `/etc/chrony.conf` or in files under `/etc/chrony.d/`.

2. The client host needs to trust the Root CA that signed the TLS certificate. Find details on how to manage the CA certificate store in a dedicated article (https://documentation.suse.com/smart/security/html/tls-certificates/index.html#tls-certificates-store).

3. Add the NTS time server source that you configured in *Procedure 39.1, "Configuring NTS time server"* to the client `chrony` configuration in `/etc/chrony.conf`, for example:

```
server nts1.example.com iburst nts
```

4. Restart the `chronyd` service.

```
> sudo systemctl restart chronyd.service
```

5. Verify configured time sources on the client and confirm that the connection is authenticated.

```
> sudo chronyc sources -v
> sudo chronyc -N authdata
```

6. On the NTS time server, verify per client statistics about NTS connections.

```
> sudo chronyc -N clients -k
```

39.4 Configure chronyd at runtime using **chronyc**

You can use **chronyc** to change the behavior of chronyd at runtime. It also generates status reports about the operation of chronyd.

You can run **chronyc** either in interactive or non-interactive mode. To run **chronyc** interactively, enter **chronyc** on the command line. It displays a prompt and waits for your command input. For example, to check how many NTP sources are online or offline, run:

```
# chronyc
chronyc> activity
200 OK
4 sources online
2 sources offline
1 sources doing burst (return to online)
1 sources doing burst (return to offline)
0 sources with unknown address
```

To exit **chronyc**'s prompt, enter **quit** or **exit**.

If you do not need to use the interactive prompt, enter the command directly:

```
# chronyc activity
```

Note: Temporary changes

Changes made using **chronyc** are not permanent. They will be lost after the next chronyd restart. For permanent changes, modify /etc/chrony.conf.

For a complete list of **chronyc** commands, see its man page (**man 1 chronyc**).

39.5 Dynamic time synchronization at runtime

Although `chronyd` starts up normally on a system that boots without a network connection, the tool cannot resolve the DNS names of the time servers specified in the configuration file.

`chronyd` keeps trying to resolve the time server names specified by the `server`, `pool`, and `peer` directives in an increasing time interval until it succeeds.

If the time server will not be reachable when `chronyd` is started, you can specify the `offline` option:

```
server server_address offline
```

`chronyd` does not try to poll the server until it is enabled using the following command:

```
# chronyc online server_address
```

When the `auto_offline` option is set, `chronyd` assumes that the time server has gone offline when two requests have been sent to it without receiving a response. This option avoids the need to run the `offline` command from **`chronyc`** when disconnecting the network link.

39.6 Setting up a local reference clock

The software package `chrony` relies on other programs (such as `gpsd`) to access the timing data via the SHM or SOCK driver. Use the `refclock` directive in `/etc/chrony.conf` to specify a hardware reference clock to be used as a time source. It has two mandatory parameters: a driver name and a driver-specific parameter. The two parameters are followed by zero or more `refclock` options. `chronyd` includes the following drivers:

- PPS - driver for the kernel `pulse per second` API. For example:

  ```
  refclock PPS /dev/pps0 lock NMEA refid GPS
  ```

- SHM - NTP shared memory driver. For example:

  ```
  refclock SHM 0 poll 3 refid GPS1
  refclock SHM 1:perm=0644 refid GPS2
  ```

- SOCK - Unix domain socket driver. For example:

```
refclock SOCK /var/run/chrony.ttyS0.sock
```

- PHC - PTP hardware clock driver. For example:

```
refclock PHC /dev/ptp0 poll 0 dpoll -2 offset -37
refclock PHC /dev/ptp1:nocrossts poll 3 pps
```

For more information on individual drivers' options, see **`man 8 chrony.conf`**.

39.7 Clock synchronization to an external time reference (ETR)

Support for clock synchronization to an external time reference (ETR) is available. The external time reference sends an oscillator signal and a synchronization signal every 2**20 (2 to the power of 20) microseconds to keep TOD clocks of all connected servers synchronized.

For availability two ETR units can be connected to a machine. If the clock deviates for more than the sync-check tolerance all CPUs get a machine check that indicates that the clock is not synchronized. If this happens, all DASD I/O to XRC enabled devices is stopped until the clock is synchronized again.

The ETR support is activated via two `sysfs` attributes; run the following commands as `root`:

```
echo 1 > /sys/devices/system/etr/etr0/online
echo 1 > /sys/devices/system/etr/etr1/online
```

VI Troubleshooting

40 Help and documentation 502

41 Gathering system information for support 507

42 Common problems and their solutions 537

40 Help and documentation

Revision History

2024-05-13

SUSE® Linux Enterprise Desktop comes with several sources of information and documentation, available online or integrated into your installed system.

Product Documentation

Extensive documentation for SUSE Linux Enterprise Desktop is available at https://documentation.suse.com/#sled. Topics covered range from deployment, upgrade and system administration to virtualization, system tuning and security, among others.

Documentation in `/usr/share/doc`

This directory holds release notes for your system (in the subdirectory `release-notes`). It also contains information of installed packages in the subdirectory `packages`. Find more detailed information in *Section 40.1, “Documentation directory”*.

Man pages and info pages for shell commands

When working with the shell, you do not need to know the options of the commands by heart. Traditionally, the shell provides integrated help by means of man pages and info pages. Read more in *Section 40.2, “Man pages”* and *Section 40.3, “Info pages”*.

Desktop help center

The help center of the GNOME desktop (Help) provides central access to the GNOME desktop documentation.

Separate help packages for certain applications

When installing new software with YaST, the software documentation is normally installed automatically and appears in the help center of your desktop. However, certain applications, such as GIMP, may have different online help packages that can be installed separately with YaST and do not integrate into the help centers.

40.1 Documentation directory

The traditional directory to find documentation on your installed Linux system is `/usr/share/doc`. The directory contains the release notes and information about the packages installed on your system, plus manuals and more.

> **Note: Contents depend on installed packages**
>
> In the Linux world, manuals and other kinds of documentation are available in the form of packages, like software. How much and which information you find in `/usr/share/doc` also depends on the (documentation) packages installed. If you cannot find the subdirectories mentioned here, check if the respective packages are installed on your system and add them with YaST, if needed.

40.1.1 Release notes

We provide HTML, PDF, RTF and text versions of SUSE Linux Enterprise Desktop release notes. They are available on your installed system under `/usr/share/doc/release-notes/` or online at your product-specific Web page at https://www.suse.com/releasenotes/index.html.

40.1.2 Package documentation

Under `packages`, find the documentation that is included in the software packages installed on your system. For every package, a subdirectory `/usr/share/doc/packages/PACKAGENAME` is created. It often contains README files for the package and sometimes examples, configuration files, or additional scripts. The following list introduces typical files to be found under `/usr/share/doc/packages`. None of these entries are mandatory and many packages only include a few of them.

`AUTHORS`

List of the main developers.

`BUGS`

Known bugs or malfunctions. May also contain a link to a Bugzilla Web page where you can search all bugs.

`CHANGES` ,
`ChangeLog`

Summary of changes from version to version. It is interesting for developers, because it is detailed.

`COPYING`,
`LICENSE`
: Licensing information.

`FAQ`
: Question and answers collected from mailing lists or newsgroups.

`INSTALL`
: How to install this package on your system. As the package is already installed by the time you get to read this file, you can safely ignore the contents of this file.

`README`, `README.*`
: General information on the software. For example, for what purpose and how to use it.

`TODO`
: Features planned for the future.

`MANIFEST`
: List of files with a brief summary.

`NEWS`
: Description of what is new in this version.

40.2 Man pages

Man pages are an essential part of any Linux system. They explain the usage of a command and all available options and parameters. Man pages can be accessed with **`man`** followed by the name of the command, for example, **`man ls`**.

Man pages are displayed directly in the shell. To navigate them, move up and down with `Page ↑` and `Page ↓`. Move between the beginning and the end of a document with `Home` and `End`. End this viewing mode by pressing `Q`. Learn more about the **`man`** command itself with **`man man`**. Man pages are sorted in categories as shown in *Table 40.1, "Man pages—categories and descriptions"* (taken from the man page for man itself).

TABLE 40.1: MAN PAGES—CATEGORIES AND DESCRIPTIONS

Number	Description
1	Executable programs or shell commands

Number	Description
2	System calls (functions provided by the kernel)
3	Library calls (functions within program libraries)
4	Special files (normally found in `/dev`)
5	File formats and conventions (`/etc/fstab`)
6	Games
7	Miscellaneous (including macro packages and conventions), for example, man(7), groff(7)
8	System administration commands (normally only for `root`)
9	Kernel routines (nonstandard)

Each man page consists of several parts labeled *NAME*, *SYNOPSIS*, *DESCRIPTION*, *SEE ALSO*, *LICENSING*, and *AUTHOR*. There may be additional sections available depending on the type of command.

40.3 Info pages

Info pages are another important source of information on your system. normally, they are more detailed than man pages. They consist of more than command line options and contain sometimes whole tutorials or reference documentation. To view the info page for a certain command, enter **`info`** followed by the name of the command, for example, **`info ls`**. You can browse an info page with a viewer directly in the shell and display the different sections, called “nodes”. Use `Space` to move forward and `<–` to move backward. Within a node, you can also browse with `Page ↑` and `Page ↓` but only `Space` and `<–` takes you also to the previous or subsequent node. Press `Q` to end the viewing mode. Not every command comes with an info page and vice versa.

40.4 Online resources

For an overview of all documentation available for SUSE Linux Enterprise Desktop check out the product-specific documentation Web pages at https://documentation.suse.com/.

If you are searching for additional product-related information, you can also refer to the following Web sites:

SUSE technical support

The SUSE Technical Support can be found at https://www.suse.com/support/ if you have questions or need solutions for technical problems.

User community

SUSE and Rancher Community (https://www.rancher.com/community)

SUSE blog

The SUSE blog offers articles, tips, Q and A: https://www.suse.com/c/blog/

GNOME documentation

Documentation for GNOME users, administrators and developers is available at https://help.gnome.org/.

The Linux documentation project

The Linux Documentation Project (TLDP) is run by a team of volunteers who write Linux-related documentation (see https://tldp.org/). It is a comprehensive documentation resource for Linux. The set of documents contains tutorials for beginners, but is mainly focused on experienced users and professional system administrators. TLDP publishes HOWTOs, FAQs and guides (handbooks) under a free license. Parts of the documentation from TLDP are also available on SUSE Linux Enterprise Desktop.

41 Gathering system information for support

Revision History

2025-04-09

For a quick overview of all relevant system information of a machine, SUSE Linux Enterprise Desktop offers the `hostinfo` package. It also helps system administrators to check for tainted kernels (that are not supported) or any third-party packages installed on a machine.

In case of problems, a detailed system report may be created with either the **`supportconfig`** command line tool or the YaST *Support* module. Both collect information about the system such as: current kernel version, hardware, installed packages, partition setup, and much more. The result is a TAR archive of files. After opening a Service Request (SR), you can upload the TAR archive to Global Technical Support. It helps to locate the issue you reported and to assist you in solving the problem.

Additionally, you can analyze the **`supportconfig`** output for known issues to help resolve problems faster. For this purpose, SUSE Linux Enterprise Desktop provides both an appliance and a command line tool for `Supportconfig Analysis` (SCA).

41.1 Displaying current system information

For a quick and easy overview of all relevant system information when logging in to a server, use the package `hostinfo`. After it has been installed on a machine, the console displays the following information to any `root` user that logs in to this machine:

EXAMPLE 41.1: OUTPUT OF **`hostinfo`** WHEN LOGGING IN AS `root`

```
Welcome to SUSE Linux Enterprise Server 15 SP2 Snapshot8 (x86_64) - Kernel \r (\l).


Distribution:              SUSE Linux Enterprise Server 15 SP7
Current As Of:             Mon 11 March 2024 10:11:51 AM CET
Hostname:                  localhost
Kernel Version:            6.4.0-150700.38-default
 Architecture:             x86_64
```

```
 Installed:              Fri 08 March 2024 04:45:50 PM CET
 Status:                 Not Tainted
Last Installed Package:  Mon 11 March 2024 10:02:13 AM CET
 Patches Needed:         0
 Security:               0
 3rd Party Packages:     6
Network Interfaces
 eth0:                   192.168.2/24 2002:c0a8:20a::/64
Memory
 Total/Free/Avail:       7.4Gi/6.4Gi/6.8Gi (91% Avail)
CPU Load Average:        7 (3%) with 2 CPUs
```

In case the output shows a `tainted` kernel status, see *Section 41.6, "Support of kernel modules"* for more details.

41.2 Collecting system information with supportconfig

To create a TAR archive with detailed system information that you can hand over to Global Technical Support, use either:

- the command **`supportconfig`** or,
- the YaST *Support* module.

The command line tool is provided by the package `supportutils` which is installed by default. The YaST *Support* module is also based on the command line tool.

Certain packages integrate Supportconfig plug-ins. When Supportconfig is executed, all plug-ins are executed as well and create one or more result files for the archive. The benefit is that only topics with a specific plug-in are checked. Supportconfig plug-ins are stored in the directory `/usr/lib/supportconfig/plugins/`.

41.2.1 Creating a service request number

Supportconfig archives can be generated at any time. However, for handing over the Supportconfig data to Global Technical Support, you need to generate a service request number first. You need it to upload the archive to support.

To create a service request, go to https://scc.suse.com/support/requests and follow the instructions on the screen. Write down the service request number.

> **Note: Privacy statement**
>
> SUSE treats system reports as confidential data. For details about our privacy commitment, see https://www.suse.com/company/policies/privacy/ .

41.2.2 Upload targets

After having created a service request number, you can upload your Supportconfig archives to Global Technical Support as described in *Procedure 41.1, "Submitting information to support with YaST"* or *Procedure 41.2, "Submitting information to support from command line"*. Use one of the following upload targets:

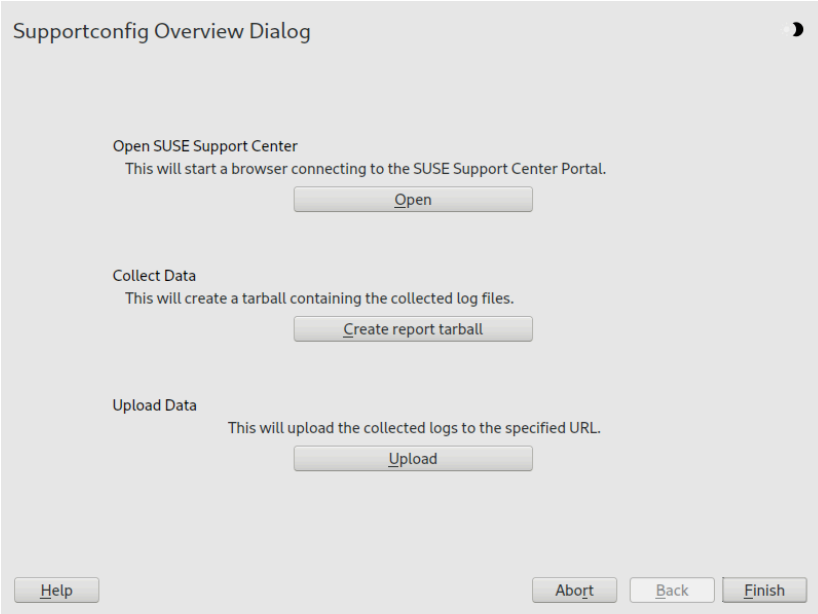
- North America: FTP ftp://support-ftp.us.suse.com/incoming/
- EMEA, Europe, the Middle East and Africa: FTP ftp://support-ftp.emea.suse.com/incoming

Alternatively, you can manually attach the TAR archive to your service request using the service request URL: https://scc.suse.com/support/requests .

41.2.3 Creating a supportconfig archive with YaST

To use YaST to gather your system information, proceed as follows:

1. Start YaST and open the *Support* module.



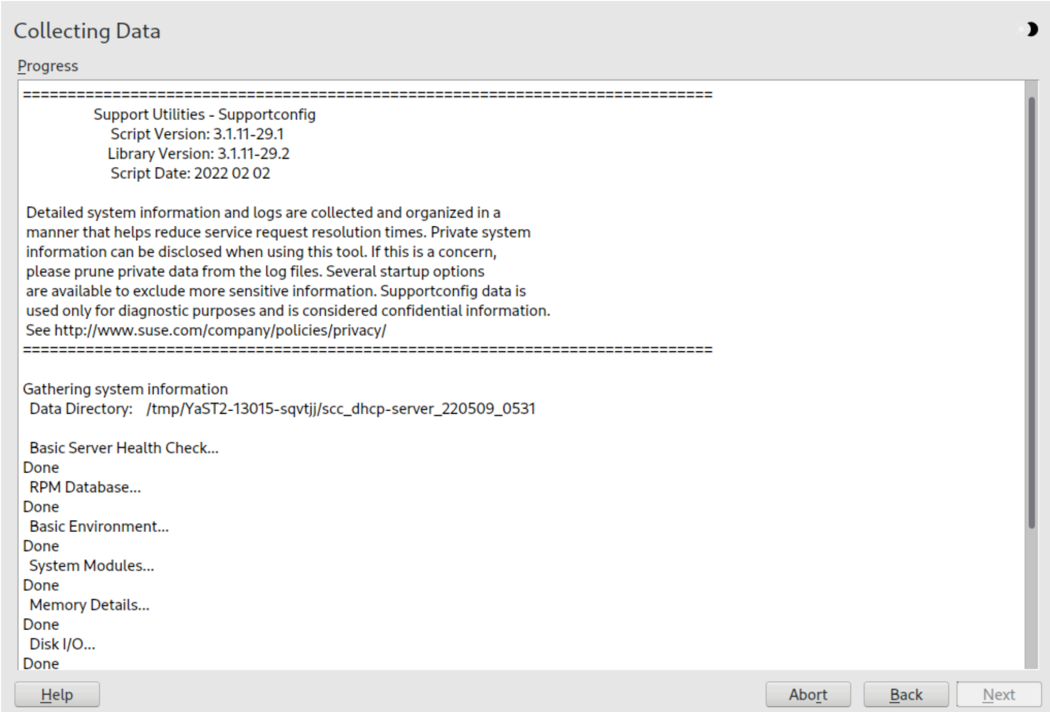
2. Click *Create report tarball*.

3. In the next window, select one of the Supportconfig options from the radio button list. *Use Custom (Expert) Settings* is preselected by default. To test the report function first, use *Only gather a minimum amount of info*. For additional information on the other options, refer to the **`supportconfig`** man page.
   Click *Next*.

4. Enter your contact information. It is saved in the `basic-environment.txt` file and included in the created archive.

5. To submit the archive to Global Technical Support, provide the required *Upload Information*. YaST automatically suggests an upload server. To modify it, refer to *Section 41.2.2, "Upload targets"* for details of which upload servers are available.
   To submit the archive later, leave the *Upload Information* empty.

6. Click *Next* to start the information collection process.



After the process is finished, click *Next*.

7. To review the collected data, select the desired file from *File Name* to view its contents in YaST. To remove a file from the TAR archive before submitting it to support, use *Remove from Data*. Press *Next*.

8. Save the TAR archive. If you started the YaST module as `root` user, YaST prompts to save the archive to `/var/log` (otherwise, to your home directory). The file name format is `scc_HOST_DATE_TIME.tbz`.

9. To upload the archive to support directly, make sure *Upload log files tarball to URL* is activated. The *Upload Target* shown here is the one that YaST suggests in *Step 5*. To modify the upload target, check which upload servers are available in *Section 41.2.2, "Upload targets"*.

10. To skip the upload, deactivate *Upload log files tarball to URL*.

11. Confirm the changes to close the YaST module.

41.2.4 Creating a supportconfig archive from command line

The following procedure shows how to create a Supportconfig archive, but without submitting it to support directly. For uploading it, you need to run the command with certain options as described in *Procedure 41.2, "Submitting information to support from command line"*.

1. Open a shell and become `root`.
2. Run **`supportconfig`**. It is enough to run this tool without any options. However, the most common options are displayed in the following list:

   `-E` *`MAIL`*,
   `-N` *`NAME`*,
   `-O` *`COMPANY`*,
   `-P` *`PHONE`*
   : Sets your contact data: e-mail address (`-E`), company name (`-O`), your name (`-N`), and your phone number (`-P`).

   `-i` *`KEYWORDS`*,
   `-F`
   : Limits the features to check. The placeholder *`KEYWORDS`* is a comma separated list of case-sensitive keywords. Get a list of all keywords with **`supportconfig -F`**.

   `-r` *`SRNUMBER`*
   : Defines your service request number when uploading the generated TAR archive.

3. Wait for the tool to complete the operation.
4. The default archive location is `/var/log`, with the file name format being `scc_`*`HOST_DATE_TIME`*`.tbz`

41.2.5 Understanding the output of `supportconfig`

Whether you run **`supportconfig`** through YaST or directly, the script gives you a summary of what it did.

```
                         Support Utilities - Supportconfig
                              Script Version: 3.0-98
                              Script Date: 2017 06 01
[...]
```

```
Gathering system information
  Data Directory:    /var/log/scc_d251_180201_1525 ❶

  Basic Server Health Check...                      Done  ❷
  RPM Database...                                   Done  ❷
  Basic Environment...                              Done  ❷
  System Modules...                                 Done  ❷
[...]
  File System List...                               Skipped  ❸
[...]
  Command History...                                Excluded  ❹
[...]
  Supportconfig Plugins:                            1  ❺
    Plugin: pstree...                               Done
[...]
Creating Tar Ball

==[ DONE ]===================================================================
  Log file tar ball: /var/log/scc_d251_180201_1525.txz ❻
  Log file size:     732K
  Log file md5sum:   bf23e0e15e9382c49f92cbce46000d8b
=============================================================================
```

❶ The temporary data directory to store the results. This directory is archived as tar file, see ❻.

❷ The feature was enabled (either by default or selected manually) and executed successfully. The result is stored in a file (see *Table 41.1, "Comparison of features and file names in the TAR archive"*).

❸ The feature was skipped because files of one or more RPM packages were changed.

❹ The feature was excluded because it was deselected via the `-x` option.

❺ The script found one plug-in and executes the plug-in **`pstree`**. The plug-in was found in the directory `/usr/lib/supportconfig/plugins/`. See the man page for details.

❻ The tar file name of the archive, by default compressed with **`xz`**.

41.2.6 Common supportconfig options

The **supportconfig** utility is usually called without any options. Display a list of all options with **supportconfig** -h or refer to the man page. The following list gives a brief overview of common use cases:

Reducing the size of the information being gathered

Use the minimal option (-m):

```
> sudo supportconfig -m
```

Limiting the information to a specific topic

If you have already localized a problem that relates to a specific area or feature set only, you should limit the collected information to the specific area for the next **supportconfig** run. For example, if you detected problems with LVM and want to test a recent change that you did to the LVM configuration. In that case it makes sense to gather the minimum Supportconfig information around LVM only:

```
> sudo supportconfig -i LVM
```

Additional keywords can be separated through commas. For example, an additional disk test:

```
> sudo supportconfig -i LVM,DISK
```

For a complete list of feature keywords that you can use for limiting the collected information to a specific area, run:

```
> sudo supportconfig -F
```

Including additional contact information in the output:

```
> sudo supportconfig -E tux@example.org -N "Tux Penguin" -O "Penguin Inc." ...
```

(all in one line)

Collecting already rotated log files

```
> sudo supportconfig -l
```

This is especially useful in high logging environments or after a kernel crash when syslog rotates the log files after a reboot.

41.2.7 Overview of the archive content

The TAR archive contains all the results from the features. The set of features can be limited through the `-i` option (see *Section 41.2.6, "Common supportconfig options"*).

To list the content of the archive, use the following **`tar`** command:

```
# tar xf /var/log/scc_earth_180131_1545.tbz
```

The following file names are always available inside the TAR archive:

MINIMUM FILES IN ARCHIVE

`basic-environment.txt`
: Contains the date when this script was executed and system information like version of the distribution, hypervisor information, and more.

`basic-health-check.txt`
: Contains basic health checks like uptime, virtual memory statistics, free memory and hard disk, checks for zombie processes, and more.

`hardware.txt`
: Contains basic hardware checks like information about the CPU architecture, list of all connected hardware, interrupts, I/O ports, kernel boot messages, and more.

`messages.txt`
: Contains log messages from the system journal.

`rpm.txt`
: Contains a list of all installed RPM packages, the name, where they are coming from, and their versions.

`summary.xml`
: Contains information in XML format like distribution, the version, and product specific fragments.

`supportconfig.txt`
: Contains information about the **`supportconfig`** script itself.

`y2log.txt`
: Contains YaST specific information like specific packages, configuration files, and log files.

Table 41.1, "Comparison of features and file names in the TAR archive" lists all available features and their file names. Further service packs can extend the list, as can plug-ins.

TABLE 41.1: COMPARISON OF FEATURES AND FILE NAMES IN THE TAR ARCHIVE

Feature	File name
APPARMOR	security-apparmor.txt
AUDIT	security-audit.txt
AUTOFS	fs-autofs.txt
BOOT	boot.txt
BTRFS	fs-btrfs.txt
DAEMONS	systemd.txt
CIMOM	cimom.txt
CRASH	crash.txt
CRON	cron.txt
DHCP	dhcp.txt
DISK	fs-diskio.txt
DNS	dns.txt
DOCKER	docker.txt
DRBD	drbd.txt
ENV	env.txt
ETC	etc.txt
HA	ha.txt
HAPROXY	haproxy.txt
HISTORY	shell_history.txt
IB	ib.txt
IMAN	novell-iman.txt
ISCSI	fs-iscsi.txt
LDAP	ldap.txt

Feature	File name
LIVEPATCH	kernel-livepatch.txt
LVM	lvm.txt
MEM	memory.txt
MOD	modules.txt
MPIO	mpio.txt
NET	network-*.txt
NFS	nfs.txt
NTP	ntp.txt
NVME	nvme.txt
OCFS2	ocfs2.txt
OFILES	open-files.txt
PRINT	print.txt
PROC	proc.txt
SAR	sar.txt
SLERT	slert.txt
SLP	slp.txt
SMT	smt.txt
SMART	fs-smartmon.txt
SMB	samba.txt
SRAID	fs-softraid.txt
SSH	ssh.txt
SSSD	sssd.txt
SYSCONFIG	sysconfig.txt

Feature	File name
`SYSFS`	`sysfs.txt`
`TRANSACTIONAL`	`transactional-update.txt`
`TUNED`	`tuned.txt`
`UDEV`	`udev.txt`
`UFILES`	`fs-files-additional.txt`
`UP`	`updates.txt`
`WEB`	`web.txt`
`X`	`x.txt`

41.3 Submitting information to Global Technical Support

Use the YaST *Support* module or the **`supportconfig`** command line utility to submit system information to the Global Technical Support. When you experience a server issue and want the support's assistance, you will need to open a service request first. For details, see *Section 41.2.1, "Creating a service request number"*.

The following examples use *`12345678901`* as a placeholder for your service request number. Replace *`12345678901`* with the service request number you created in *Section 41.2.1, "Creating a service request number"*.

PROCEDURE 41.1: SUBMITTING INFORMATION TO SUPPORT WITH YAST

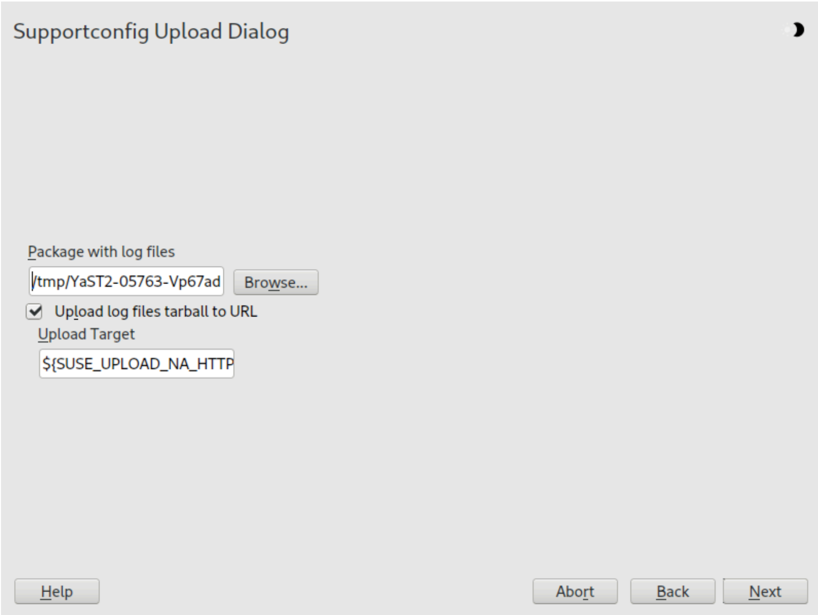
The following procedure assumes that you have already created a Supportconfig archive, but have not uploaded it yet. Make sure to have included your contact information in the archive as described in *Section 41.2.3, "Creating a supportconfig archive with YaST"*, *Step 4*. For instructions on how to generate and submit a Supportconfig archive in one go, see *Section 41.2.3, "Creating a supportconfig archive with YaST"*.

1. Start YaST and open the *Support* module.

2. Click *Upload*.

3. In *Package with log files* specify the path to the existing Supportconfig archive or *Browse* for it.

4. YaST automatically proposes an upload server. To modify it, refer to *Section 41.2.2, “Upload targets”* for details of which upload servers are available.



Proceed with *Next*.

5. Click *Finish*.

PROCEDURE 41.2: SUBMITTING INFORMATION TO SUPPORT FROM COMMAND LINE

The following procedure assumes that you have already created a Supportconfig archive, but have not uploaded it yet. For instructions on how to generate and submit a Supportconfig archive in one go, see *Section 41.2.3, “Creating a supportconfig archive with YaST”*.

1. Servers with Internet connectivity:

   a. To use the default upload target, run:

   ```
   > sudo supportconfig -ur 12345678901
   ```

   b. For the secure upload target, use the following:

   ```
   > sudo supportconfig -ar 12345678901
   ```

2. Servers *without* Internet connectivity

a. Run the following:

```
> sudo supportconfig -r 12345678901
```

b. Manually upload the `/var/log/scc_SR12345678901*tbz` archive to one of our FTP servers. Which one to use depends on your location in the world. For an overview, see *Section 41.2.2, “Upload targets”*.

3. After the TAR archive arrives in the incoming directory of our FTP server, it becomes automatically attached to your service request.

41.4 Analyzing system information

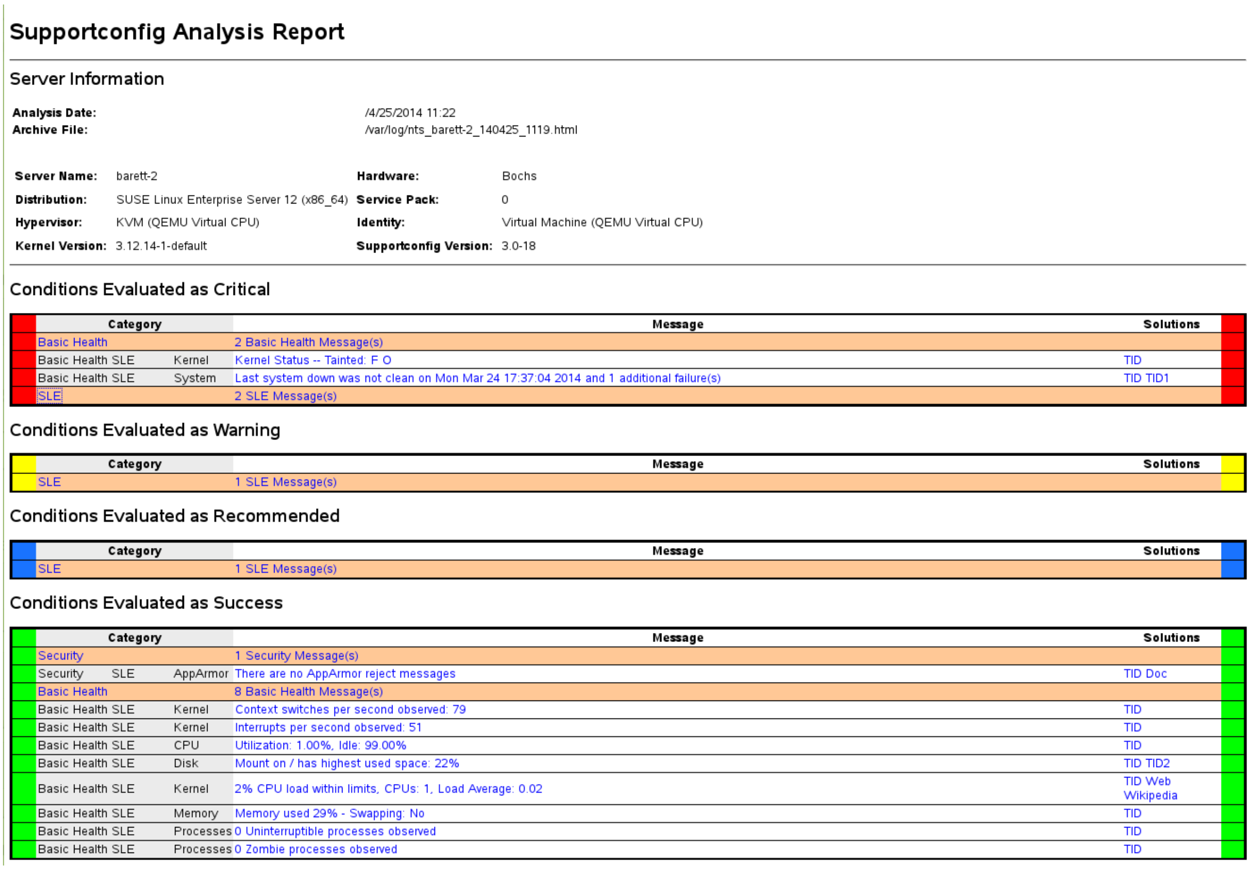
System reports created with **supportconfig** can be analyzed for known issues to help resolve problems faster. For this purpose, SUSE Linux Enterprise Desktop provides both an appliance and a command line tool for `Supportconfig Analysis` (SCA). The SCA appliance is a server-side tool which is non-interactive. The SCA tool (**scatool** provided by the package `sca-server-report`) runs on the client-side and is executed from command line. Both tools analyze Supportconfig archives from affected servers. The initial server analysis takes place on the SCA appliance or the workstation on which **scatool** is running. No analysis cycles happen on the production server.

Both the appliance and the command line tool additionally need product-specific patterns that enable them to analyze the Supportconfig output for the associated products. Each pattern is a script that parses and evaluates a Supportconfig archive for one known issue. The patterns are available as RPM packages.

You can also develop your own patterns as briefly described in *Section 41.4.3, “Developing custom analysis patterns”*.

41.4.1 SCA command line tool

The SCA command line tool lets you analyze a local machine using both **supportconfig** and the analysis patterns for the specific product that is installed on the local machine. The tool creates an HTML report showing its analysis results. For an example, see *Figure 41.1, “HTML report generated by SCA tool”*.



Supportconfig Analysis Report

Server Information

Analysis Date: /4/25/2014 11:22
Archive File: /var/log/nts_barett-2_140425_1119.html

Server Name:	barett-2	**Hardware:**	Bochs
Distribution:	SUSE Linux Enterprise Server 12 (x86_64)	**Service Pack:**	0
Hypervisor:	KVM (QEMU Virtual CPU)	**Identity:**	Virtual Machine (QEMU Virtual CPU)
Kernel Version:	3.12.14-1-default	**Supportconfig Version:**	3.0-18

Conditions Evaluated as Critical

Category			Message	Solutions
Basic Health			2 Basic Health Message(s)	
Basic Health	SLE	Kernel	Kernel Status -- Tainted: F O	TID
Basic Health	SLE	System	Last system down was not clean on Mon Mar 24 17:37:04 2014 and 1 additional failure(s)	TID TID1
SLE			2 SLE Message(s)	

Conditions Evaluated as Warning

Category	Message	Solutions
SLE	1 SLE Message(s)	

Conditions Evaluated as Recommended

Category	Message	Solutions
SLE	1 SLE Message(s)	

Conditions Evaluated as Success

Category			Message	Solutions
Security			1 Security Message(s)	
Security	SLE	AppArmor	There are no AppArmor reject messages	TID Doc
Basic Health			8 Basic Health Message(s)	
Basic Health	SLE	Kernel	Context switches per second observed: 79	TID
Basic Health	SLE	Kernel	Interrupts per second observed: 51	TID
Basic Health	SLE	CPU	Utilization: 1.00%, Idle: 99.00%	TID
Basic Health	SLE	Disk	Mount on / has highest used space: 22%	TID TID2
Basic Health	SLE	Kernel	2% CPU load within limits, CPUs: 1, Load Average: 0.02	TID Web Wikipedia
Basic Health	SLE	Memory	Memory used 29% - Swapping: No	TID
Basic Health	SLE	Processes	0 Uninterruptible processes observed	TID
Basic Health	SLE	Processes	0 Zombie processes observed	TID

FIGURE 41.1: HTML REPORT GENERATED BY SCA TOOL

The **scatool** command is provided by the `sca-server-report` package. It is not installed by default. Additionally, you need the `sca-patterns-base` package and any of the product-specific `sca-patterns-*` packages that matches the product installed on the machine where you want to run the **scatool** command.

Execute the **scatool** command either as `root` user or with **sudo**. When calling the SCA tool, either analyze an existing **supportconfig** TAR archive or let it generate and analyze a new archive in one go. The tool also provides an interactive console with tab completion. It is possible to run **supportconfig** on an external machine and to execute the subsequent analysis on the local machine.

Find a few example commands below:

`sudo scatool` `-s`
: Calls **`supportconfig`** and generates a new Supportconfig archive on the local machine. Analyzes the archive for known issues by applying the SCA analysis patterns that match the installed product. Displays the path to the HTML report that is generated from the results of the analysis. It is written to the same directory where the Supportconfig archive can be found.

`sudo scatool` `-s` `-o` *`/opt/sca/reports/`*
: Same as **`sudo scatool`** `-s`, only that the HTML report is written to the path specified with `-o`.

`sudo scatool` `-a` *`PATH_TO_TARBALL_OR_DIR`*
: Analyzes the specified Supportconfig archive file (or the specified directory to where the Supportconfig archive has been extracted). The generated HTML report is saved in the same location as the Supportconfig archive or directory.

`sudo scatool` `-a` *`SLES_SERVER.COMPANY.COM`*
: Establishes an SSH connection to an external server *`SLES_SERVER.COMPANY.COM`* and runs **`supportconfig`** on the server. The Supportconfig archive is then copied back to the local machine and is analyzed there. The generated HTML report is saved to the default `/var/log` directory. (Only the Supportconfig archive is created on *`SLES_SERVER.COMPANY.COM`*).

`sudo scatool` `-c`
: Starts the interactive console for **`scatool`**. Press →| twice to see the available commands.

For further options and information, run **`sudo scatool -h`** or see the **`scatool`** man page.

41.4.2 SCA appliance

If you decide to use the SCA appliance for analyzing the Supportconfig archives, configure a dedicated server (or virtual machine) as the SCA appliance server. The SCA appliance server can then be used to analyze Supportconfig archives from all machines in your enterprise running SUSE Linux Enterprise Server or SUSE Linux Enterprise Desktop. You can simply upload Supportconfig archives to the appliance server for analysis. Interaction is not required. In a MariaDB database, the SCA appliance keeps track of all Supportconfig archives that have been analyzed .

You can read the SCA reports directly from the appliance Web interface. Alternatively, you can have the appliance send the HTML report to any administrative user via e-mail. For details, see *Section 41.4.2.5.4, “Sending SCA reports via e-mail”*.

41.4.2.1 Installation quick start

To install and set up the SCA appliance in a fast way from the command line, follow the instructions here. The procedure is intended for experts and focuses on the bare installation and setup commands. For more information, refer to the more detailed description in *Section 41.4.2.2, “Prerequisites”* to *Section 41.4.2.3, “Installation and basic setup”*.

PREREQUISITES

- Web and LAMP Pattern
- Web and Scripting Module (you must register the machine to be able to select this module).

Note: `root` privileges required

All commands in the following procedure must be run as `root`.

PROCEDURE 41.3: INSTALLATION USING ANONYMOUS FTP FOR UPLOAD

After the appliance is set up and running, no more manual interaction is required. This way of setting up the appliance is therefore ideal for using cron jobs to create and upload Supportconfig archives.

1. On the machine on which to install the appliance, log in to a console and execute the following commands (make sure to accept the recommended packages):

```
> sudo zypper install sca-appliance-* sca-patterns-* \
vsftpd yast2 yast2-ftp-server
> sudo systemctl enable apache2
> sudo systemctl start apache2
> sudo systemctl enable vsftpd
> sudo systemctl start vsftpd
> sudo yast ftp-server
```

2. In YaST FTP Server, select *Authentication* › *Enable Upload* › *Anonymous Can Upload* › *Finish* › *Yes* to *Create /srv/ftp/upload*.

3. Execute the following commands:

```
> sudo systemctl enable mysql
> sudo systemctl start mysql
> sudo mysql_secure_installation
> sudo setup-sca -f
```

The mysql_secure_installation will create a MariaDB `root` password.

PROCEDURE 41.4: INSTALLATION USING SCP/TMP FOR UPLOAD

This way of setting up the appliance requires manual interaction when typing the SSH password.

1. On the machine on which to install the appliance, log in to a console.

2. Execute the following commands:

```
> sudo zypper install sca-appliance-* sca-patterns-*
> sudo systemctl enable apache2
> sudo systemctl start apache2
> sudo sudo systemctl enable mysql
> sudo systemctl start mysql
> sudo mysql_secure_installation
> sudo setup-sca
```

41.4.2.2 Prerequisites

To run an SCA appliance server, you need the following prerequisites:

- All `sca-appliance-*` packages.
- The `sca-patterns-base` package. Additionally, any of the product-specific `sca-patterns-*` for the type of Supportconfig archives that you want to analyze with the appliance.
- Apache
- PHP
- MariaDB
- anonymous FTP server (optional)

41.4.2.3 Installation and basic setup

As listed in *Section 41.4.2.2, "Prerequisites"*, the SCA appliance has several dependencies on other packages. Therefore, you need to take preparatory steps before installing and setting up the SCA appliance server:

1. For Apache and MariaDB, install the `Web` and `LAMP` installation patterns.
2. Set up Apache, MariaDB, and optionally an anonymous FTP server.
3. Configure Apache and MariaDB to start at boot time:

```
> sudo systemctl enable apache2 mysql
```

4. Start both services:

```
> sudo systemctl start apache2 mysql
```

Now you can install the SCA appliance and set it up as described in *Procedure 41.5, "Installing and configuring the SCA appliance"*.

PROCEDURE 41.5: INSTALLING AND CONFIGURING THE SCA APPLIANCE

After installing the packages, use the **`setup-sca`** script for the basic configuration of the MariaDB administration and report database that is used by the SCA appliance.

It can be used to configure the following options you have for uploading the Supportconfig archives from your machines to the SCA appliance:

- **`scp`**
- anonymous FTP server

1. Install the appliance and the SCA base-pattern library:

```
> sudo zypper install sca-appliance-* sca-patterns-base
```

2. Additionally, install the pattern packages for the types of Supportconfig archives you want to analyze. For example, if you have SUSE Linux Enterprise Server 12 and SUSE Linux Enterprise Server 15 servers in your environment, install both the `sca-patterns-sle12` and `sca-patterns-sle15` packages.
To install all available patterns:

```
> sudo zypper install sca-patterns-*
```

3. For basic setup of the SCA appliance, use the **`setup-sca`** script. How to call it depends on how you want to upload the Supportconfig archives to the SCA appliance server:

   - If you have configured an anonymous FTP server that uses the `/srv/ftp/upload` directory, execute the setup script with the `-f` option. Follow the instructions on the screen:

     ```
     > sudo setup-sca -f
     ```

     ### Note: FTP server using another directory

     If your FTP server uses another directory than `/srv/ftp/upload`, adjust the following configuration files first to make them point to the correct directory: `/etc/sca/sdagent.conf` and `/etc/sca/sdbroker.conf` .

   - To upload Supportconfig files to the `/tmp` directory of the SCA appliance server via **`scp`**, call the setup script without any parameters. Follow the instructions on the screen:

     ```
     > sudo setup-sca
     ```

   The setup script runs a few checks regarding its requirements and configures the needed components. It will prompt you for two passwords: the MySQL `root` password of the MariaDB that you have set up, and a Web user password with which to log in to the Web interface of the SCA appliance.

4. Enter the existing MariaDB `root` password. It will allow the SCA appliance to connect to the MariaDB.

5. Define a password for the Web user. It will be written to `/srv/www/htdocs/sca/web-config.php` and will be set as the password for the user `scdiag`. Both user name and password can be changed at any time later, see *Section 41.4.2.5.1, "Password for the Web interface"*.

After successful installation and setup, the SCA appliance is ready for use, see *Section 41.4.2.4, "Using the SCA appliance"*. However, you should modify options such as changing the password for the Web interface, changing the source for the SCA pattern updates, enabling archiving mode or configuring e-mail notifications. For details on that, see *Section 41.4.2.5, "Customizing the SCA appliance"*.

Warning: Data protection

As the reports on the SCA appliance server contain security-relevant information, make sure to protect the data on the SCA appliance server against unauthorized access.

41.4.2.4 Using the SCA appliance

You can upload existing Supportconfig archives to the SCA appliance manually or create new Supportconfig archives and upload them to the SCA appliance in one step. Uploading can be done via FTP or SCP. For both, you need to know the URL where the SCA appliance can be reached. For upload via FTP, an FTP server needs to be configured for the SCA appliance, see *Procedure 41.5, “Installing and configuring the SCA appliance”*.

41.4.2.4.1 Uploading supportconfig archives to the SCA appliance

- For creating a Supportconfig archive and uploading it via (anonymous) FTP:

  ```
  > sudo supportconfig -U “ftp://SCA-APPLIANCE.COMPANY.COM/upload”
  ```

- For creating a Supportconfig archive and uploading it via SCP:

  ```
  > sudo supportconfig -U “scp://SCA-APPLIANCE.COMPANY.COM/tmp”
  ```

  You will be prompted for the `root` user password of the server running the SCA appliance.

- To manually upload one or multiple archives, copy the existing archive files (located at `/var/log/scc_*.tbz`) to the SCA appliance. As target, use either the appliance server's `/tmp` directory or the `/srv/ftp/upload` directory (if FTP is configured for the SCA appliance server).

41.4.2.4.2 Viewing SCA reports

SCA reports can be viewed from any machine that has a browser installed and can access the report index page of the SCA appliance.

1. Start a Web browser and make sure that JavaScript and cookies are enabled.

2. As a URL, enter the report index page of the SCA appliance.

```
https://sca-appliance.company.com/sca
```

If in doubt, ask your system administrator.

3. You will be prompted for a user name and a password to log in.

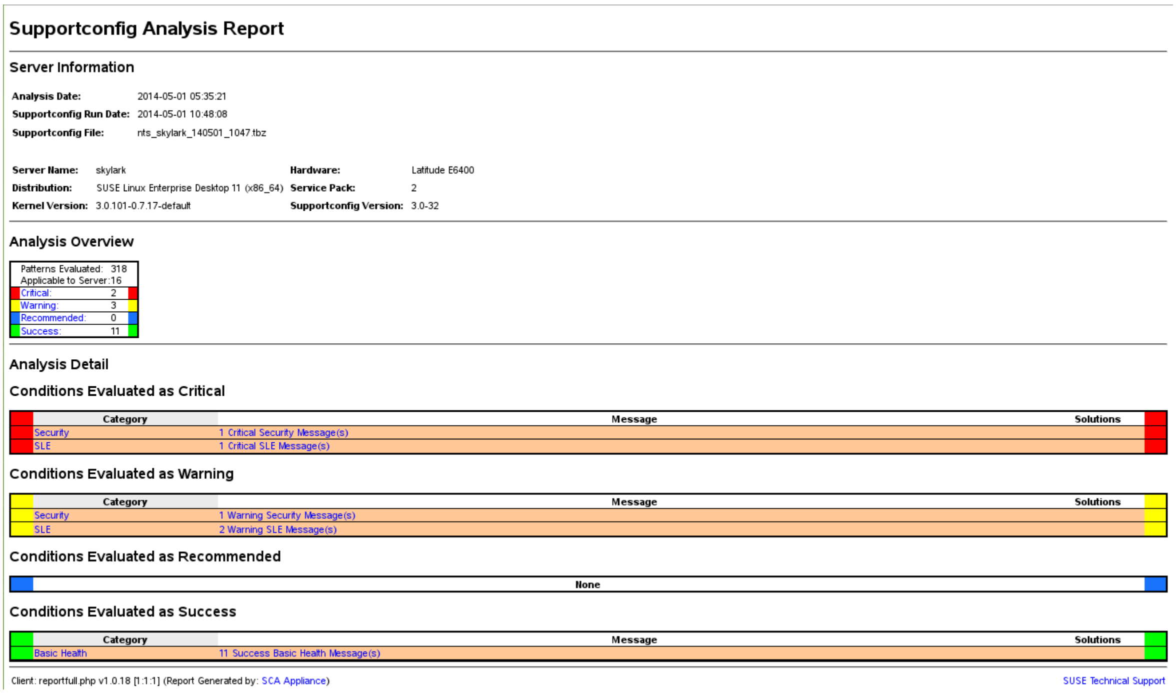


FIGURE 41.2: HTML REPORT GENERATED BY SCA APPLIANCE

4. After logging in, click the date of the report you want to read.

5. Click the *Basic Health* category first to expand it.

6. In the *Message* column, click an individual entry. This opens the corresponding article in the SUSE Knowledge base. Read the proposed solution and follow the instructions.

7. If the *Solutions* column of the *Supportconfig Analysis Report* shows any additional entries, click them. Read the proposed solution and follow the instructions.

8. Check the SUSE Knowledge base (https://www.suse.com/support/kb/) for results that directly relate to the problem identified by SCA. Work at resolving them.

9. Check for results that can be addressed proactively to avoid future problems.

41.4.2.5 Customizing the SCA appliance

The following sections show how to change the password for the Web interface, how to change the source for the SCA pattern updates, how to enable archiving mode, and how to configure e-mail notifications.

41.4.2.5.1 Password for the Web interface

The SCA Appliance Web interface requires a user name and password for logging in. The default user name is `scdiag` and the default password is `linux` (if not specified otherwise, see *Procedure 41.5, "Installing and configuring the SCA appliance"*). Change the default password to a secure password at the earliest possibility. You can also modify the user name.

PROCEDURE 41.6: CHANGING USER NAME OR PASSWORD FOR THE WEB INTERFACE

1. Log in as `root` user at the system console of the SCA appliance server.
2. Open `/srv/www/htdocs/sca/web-config.php` in an editor.
3. Change the values of `$username` and `$password` as desired.
4. Save the file and exit.

41.4.2.5.2 Updates of SCA patterns

By default, all `sca-patterns-*` packages are updated regularly by a `root` cron job that executes the `sdagent-patterns` script nightly, which in turn runs **`zypper update sca-patterns-*`**. A regular system update will update all SCA appliance and pattern packages. To update the SCA appliance and patterns manually, run:

```
> sudo zypper update sca-*
```

The updates are installed from the SUSE Linux Enterprise 15 SP7 update repository by default. You can change the source for the updates to an RMT server, if desired. When `sdagent-patterns` runs **`zypper update sca-patterns-*`**, it gets the updates from the currently configured update channel. If that channel is located on an RMT server, the packages will be pulled from there.

PROCEDURE 41.7: DISABLING AUTOMATIC UPDATES OF SCA PATTERNS

1. Log in as `root` user at the system console of the SCA appliance server.

2. Open `/etc/sca/sdagent-patterns.conf` in an editor.
3. Change the entry

   ```
   UPDATE_FROM_PATTERN_REPO=1
   ```

   to

   ```
   UPDATE_FROM_PATTERN_REPO=0
   ```

4. Save the file and exit. The machine does not require any restart to apply the change.

41.4.2.5.3 Archiving mode

All Supportconfig archives are deleted from the SCA appliance after they have been analyzed and their results have been stored in the MariaDB database. However, for troubleshooting purposes it can be useful to keep copies of Supportconfig archives from a machine. By default, archiving mode is disabled.

PROCEDURE 41.8: ENABLING ARCHIVING MODE IN THE SCA APPLIANCE

1. Log in as `root` user at the system console of the SCA appliance server.
2. Open `/etc/sca/sdagent.conf` in an editor.
3. Change the entry

   ```
   ARCHIVE_MODE=0
   ```

   to

   ```
   ARCHIVE_MODE=1
   ```

4. Save the file and exit. The machine does not require any restart to apply the change.

After having enabled archive mode, the SCA appliance will save the Supportconfig files to the `/var/log/archives/saved` directory, instead of deleting them.

41.4.2.5.4 Sending SCA reports via e-mail

The SCA appliance can e-mail a report HTML file for each Supportconfig analyzed. This feature is disabled by default. When enabling it, you can define a list of e-mail addresses to which the reports should be sent. Define a level of status messages that trigger the sending of reports (`STATUS_NOTIFY_LEVEL`).

POSSIBLE VALUES FOR STATUS_NOTIFY_LEVEL

$STATUS_OFF

Deactivate sending of HTML reports.

$STATUS_CRITICAL

Send only SCA reports that include a CRITICAL.

$STATUS_WARNING

Send only SCA reports that include a WARNING or CRITICAL.

$STATUS_RECOMMEND

Send only SCA reports that include a RECOMMEND, WARNING or CRITICAL.

$STATUS_SUCCESS

Send SCA reports that include a SUCCESS, RECOMMEND, WARNING or CRITICAL.

PROCEDURE 41.9: CONFIGURING E-MAIL NOTIFICATIONS FOR SCA REPORTS

1. Log in as `root` user at the system console of the SCA appliance server.
2. Open `/etc/sca/sdagent.conf` in an editor.
3. Search for the entry `STATUS_NOTIFY_LEVEL`. By default, it is set to `$STATUS_OFF` (e-mail notifications are disabled).
4. To enable e-mail notifications, change `$STATUS_OFF` to the level of status messages that you want to have e-mail reports for, for example:

   ```
   STATUS_NOTIFY_LEVEL=$STATUS_SUCCESS
   ```

   For details, see *Possible values for* `STATUS_NOTIFY_LEVEL`.
5. To define the list of recipients to which the reports should be sent:
   a. Search for the entry `EMAIL_REPORT='root'`.

b. Replace `root` with a list of e-mail addresses to which SCA reports should be sent. The e-mail addresses must be separated by spaces. For example:

```
EMAIL_REPORT='tux@my.company.com wilber@your.company.com'
```

6. Save the file and exit. The machine does not require any restart to apply the changes. All future SCA reports will be e-mailed to the specified addresses.

41.4.2.6 Backing up and restoring the database

To back up and restore the MariaDB database that stores the SCA reports, use the **`scadb`** command as described below. **`scadb`** is provided by the package `sca-appliance-broker`.

PROCEDURE 41.10: BACKING UP THE DATABASE

1. Log in as `root` user at the system console of the server running the SCA appliance.
2. Put the appliance into maintenance mode by executing:

```
# scadb maint
```

3. Start the backup with:

```
# scadb backup
```

The data is saved to a TAR archive: `sca-backup-*sql.gz`.

4. If you are using the pattern creation database to develop your own patterns (see *Section 41.4.3, "Developing custom analysis patterns"*), back up this data, too:

```
# sdpdb backup
```

The data is saved to a TAR archive: `sdp-backup-*sql.gz`.

5. Copy the following data to another machine or an external storage medium:

   - `sca-backup-*sql.gz`
   - `sdp-backup-*sql.gz`
   - `/usr/lib/sca/patterns/local` (only needed if you have created custom patterns)

6. Reactivate the SCA appliance with:

```
# scadb reset agents
```

PROCEDURE 41.11: RESTORING THE DATABASE

To restore the database from your backup, proceed as follows:

1. Log in as `root` user at the system console of the server running the SCA appliance.

2. Copy the newest `sca-backup-*sql.gz` and `sdp-backup-*sql.gz` TAR archives to the SCA appliance server.

3. To decompress the files, run:

```
# gzip -d *-backup-*sql.gz
```

4. To import the data into the database, execute:

```
# scadb import sca-backup-*sql
```

5. If you are using the pattern creation database to create your own patterns, also import the following data with:

```
# sdpdb import sdp-backup-*sql
```

6. If you are using custom patterns, also restore `/usr/lib/sca/patterns/local` from your backup data.

7. Reactivate the SCA appliance with:

```
# scadb reset agents
```

8. Update the pattern modules in the database with:

```
# sdagent-patterns -u
```

41.4.3 Developing custom analysis patterns

The SCA appliance comes with a complete pattern development environment (the SCA Pattern Database) that enables you to develop your own, custom patterns. Patterns can be written in any programming language. To make them available for the Supportconfig analysis process, they need to be saved to `/usr/lib/sca/patterns/local` and to be made executable. Both the

SCA appliance and the SCA tool will then run the custom patterns against new Supportconfig archives as part of the analysis report. For detailed instructions on how to create (and test) your own patterns, see https://www.suse.com/c/sca-pattern-development/ ↗.

41.5 Gathering information during the installation

During the installation, **`supportconfig`** is not available. However, you can collect log files from YaST by using **`save_y2logs`**. This command will create a `.tar.xz` archive in the directory `/tmp`.

If issues appear early during installation, you may be able to gather information from the log file created by **`linuxrc`**. **`linuxrc`** is a small command that runs before YaST starts. This log file is available at `/var/log/linuxrc.log`.

Important: Installation log files not available in the installed system

The log files available during the installation are not available in the installed system anymore. Properly save the installation log files while the installer is still running.

41.6 Support of kernel modules

An important requirement for every enterprise operating system is the level of support you receive for your environment. Kernel modules are the most relevant connector between hardware ("controllers") and the operating system. Every kernel module in SUSE Linux Enterprise has a `supported` flag that can take three possible values:

- "yes", thus `supported`
- "external", thus `supported`
- (empty, not set), thus `unsupported`

The following rules apply:

- All modules of a self-recompiled kernel are by default marked as unsupported.
- Kernel modules supported by SUSE partners and delivered using `SUSE SolidDriver Program` are marked "external".

- If the `supported` flag is not set, loading this module will taint the kernel. Tainted kernels are not supported. Unsupported Kernel modules are included in an extra RPM package (`kernel-FLAVOR-extra`). That package is only available for SUSE Linux Enterprise Desktop and the SUSE Linux Enterprise Workstation Extension. Those kernels will not be loaded by default (*FLAVOR*=`default`|`xen`|...). Besides, these unsupported modules are not available in the installer, and the `kernel-FLAVOR-extra` package is not part of the SUSE Linux Enterprise media.
- Kernel modules not provided under a license compatible to the license of the Linux kernel also taint the kernel. For details, see the state of `/proc/sys/kernel/tainted`.

41.6.1 Technical background

- Linux kernel: The value of `/proc/sys/kernel/unsupported` defaults to `2` on SUSE Linux Enterprise 15 SP7 (`do not warn in syslog when loading unsupported modules`). This default is used in the installer and in the installed system.
- **`modprobe`**: The **`modprobe`** utility for checking module dependencies and loading modules appropriately checks for the value of the `supported` flag. If the value is “yes” or “external” the module will be loaded, otherwise it will not. For information on how to override this behavior, see *Section 41.6.2, “Working with unsupported modules”*.

Note: Support

SUSE does not generally support the removal of storage modules via **`modprobe -r`**.

41.6.2 Working with unsupported modules

While general supportability is important, situations can occur where loading an unsupported module is required. For example, for testing or debugging purposes, or if your hardware vendor provides a hotfix.

- To override the default, copy `/lib/modprobe.d/10-unsupported-modules.conf` to `/etc/modprobe.d/10-unsupported-modules.conf` and change the value of the variable `allow_unsupported_modules` from `0` to `1`. Do not edit `/lib/modprobe.d/10-unsupported-modules.conf` directly; any changes will be overwritten whenever the `suse-module-tools` package is updated.

If an unsupported module is needed in the initrd, do not forget to run **dracut -f** to update the initrd.

If you only want to try loading a module once, you can use the `--allow-unsupported-modules` option with **modprobe**. For more information, see the comments in `/lib/modprobe.d/10-unsupported-modules.conf` and the **modprobe** man page.

- During installation, unsupported modules may be added through driver update disks, and they will be loaded. To enforce loading of unsupported modules during boot and afterward, use the kernel command line option `oem-modules`. While installing and initializing the `suse-module-tools` package, the kernel flag `TAINT_NO_SUPPORT` (`/proc/sys/kernel/tainted`) will be evaluated. If the kernel is already tainted, `allow_unsupported_modules` will be enabled. This will prevent unsupported modules from failing in the system being installed. If no unsupported modules are present during installation and the other special kernel command line option (`oem-modules=1`) is not used, the default still is to disallow unsupported modules.

Remember that loading and running unsupported modules will make the kernel and the whole system unsupported by SUSE.

41.7 More information

- **man supportconfig**—The **supportconfig** man page.
- **man supportconfig.conf**—The man page of the Supportconfig configuration file.
- **man scatool**—The **scatool** man page.
- **man scadb**—The **scadb** man page.
- **man setup-sca**—The **setup-sca** man page.
- https://mariadb.com/kb/en/—The MariaDB documentation.
- https://www.suse.com/c/sca-pattern-development/—Instructions on how to create (and test) your own SCA patterns.
- https://community.microfocus.com/img/gw/groupwise/w/tips/34308/create-your-own-supportconfig-plugin—Create Your Own Supportconfig Plugin.
- https://www.suse.com/c/creating-a-central-supportconfig-repository/—Creating a Central Supportconfig Repository.

42 Common problems and their solutions

Revision History

2024-05-13

This chapter describes a range of potential problems and their solutions. Even if your situation is not precisely listed, there may be one similar enough to offer hints to the solution of your problem.

42.1 Finding and gathering information

Linux reports things in a detailed way. There are several places to look when you encounter problems with your system. Most of them are standard to Linux systems, and several are relevant to SUSE Linux Enterprise Desktop systems. Most log files can be viewed with YaST (*Miscellaneous* › *Start-Up Log*).

YaST offers the possibility to collect all system information needed by the support team. Use *Other* › *Support* and select the problem category. When all information is gathered, attach it to your support request.

A list of the most frequently checked log files follows with the description of their typical purpose. Paths containing `~` refer to the current user's home directory.

TABLE 42.1: LOG FILES

Log File	Description
`~/.xsession-errors`	Messages from the desktop applications currently running.
`/var/log/apparmor/`	Log files from AppArmor, see *Book “Security and Hardening Guide”* for detailed information.
`/var/log/audit/audit.log`	Log file from Audit to track any access to files, directories, or resources of your system, and trace system calls. See *Book “Security and Hardening Guide”* for detailed information.
`/var/log/mail.*`	Messages from the mail system.

Log File	Description
`/var/log/NetworkManager`	Log file from NetworkManager to collect problems with network connectivity
`/var/log/samba/`	Directory containing Samba server and client log messages.
`/var/log/warn`	All messages from the kernel and system log daemon with the “warning” level or higher.
`/var/log/wtmp`	Binary file containing user login records for the current machine session. View it with **`last`**.
`/var/log/Xorg.*.log`	Start-up and runtime log files from the X Window System. It is useful for debugging failed X start-ups.
`/var/log/YaST2/`	Directory containing YaST's actions and their results.
`/var/log/zypper.log`	Log file of Zypper.

Apart from log files, your machine also supplies you with information about the running system. See *Table 42.2: System information with the `/proc` file system*

TABLE 42.2: SYSTEM INFORMATION WITH THE `/proc` FILE SYSTEM

File	Description
`/proc/cpuinfo`	Contains processor information, including its type, make, model, and performance.
`/proc/dma`	Shows which DMA channels are currently being used.
`/proc/interrupts`	Shows which interrupts are in use, and how many of each have been in use.

File	Description
`/proc/iomem`	Displays the status of I/O (input/output) memory.
`/proc/ioports`	Shows which I/O ports are in use at the moment.
`/proc/meminfo`	Displays memory status.
`/proc/modules`	Displays the individual modules.
`/proc/mounts`	Displays devices currently mounted.
`/proc/partitions`	Shows the partitioning of all hard disks.
`/proc/version`	Displays the current version of Linux.

Apart from the `/proc` file system, the Linux kernel exports information with the `sysfs` module, an in-memory file system. This module represents kernel objects, their attributes and relationships. For more information about `sysfs`, see the context of udev in *Chapter 29, Dynamic kernel device management with* `udev`. *Table 42.3* contains an overview of the most common directories under `/sys`.

TABLE 42.3: SYSTEM INFORMATION WITH THE `/sys` FILE SYSTEM

File	Description
`/sys/block`	Contains subdirectories for each block device discovered in the system. Generally, these are mostly disk type devices.
`/sys/bus`	Contains subdirectories for each physical bus type.
`/sys/class`	Contains subdirectories grouped together as a functional types of devices (like graphics, net, printer, etc.)
`/sys/device`	Contains the global device hierarchy.

Linux comes with several tools for system analysis and monitoring. See *Book "System Analysis and Tuning Guide", Chapter 2 "System monitoring utilities"* for a selection of the most important ones used in system diagnostics.

Each of the following scenarios begins with a header describing the problem followed by a paragraph or two offering suggested solutions, available references for more detailed solutions, and cross-references to other scenarios that are related.

42.2 Boot problems

Boot problems are situations when your system does not boot properly (does not boot to the expected target and login screen).

42.2.1 The GRUB 2 boot loader fails to load

If the hardware is functioning properly, it is possible that the boot loader is corrupted and Linux cannot start on the machine. In this case, it is necessary to repair the boot loader. To do so, you need to start the Rescue System as described in *Section 42.5.2, "Using the rescue system"* and follow the instructions in *Section 42.5.2.4, "Modifying and re-installing the boot loader"*.

Alternatively, you can use the Rescue System to fix the boot loader as follows. Boot your machine from the installation media. In the boot screen, choose *More* › *Boot Linux System*. Select the disk containing the installed system and kernel with the default kernel options.

When the system is booted, start YaST and switch to *System* › *Boot Loader*. Make sure that the *Write generic Boot Code to MBR* option is enabled, and click *OK*. This fixes the corrupted boot loader by overwriting it, or installs the boot loader if it is missing.

Other reasons for the machine not booting may be BIOS-related:

BIOS settings

Check your BIOS for references to your hard disk. GRUB 2 may simply not be started if the hard disk itself cannot be found with the current BIOS settings.

BIOS boot order

Check whether your system's boot order includes the hard disk. If the hard disk option was not enabled, your system may install properly, but fails to boot when access to the hard disk is required.

42.2.2 No login or prompt appears

This behavior typically occurs after a failed kernel upgrade and it is known as a *kernel panic* because of the type of error on the system console that sometimes can be seen at the final stage of the process. If, in fact, the machine has just been rebooted following a software update, the immediate goal is to reboot it using the old, proven version of the Linux kernel and associated files. This can be done in the GRUB 2 boot loader screen during the boot process as follows:

1. Reboot the computer using the reset button, or switch it off and on again.

2. When the GRUB 2 boot screen becomes visible, select the *Advanced Options* entry and choose the previous kernel from the menu. The machine will boot using the prior version of the kernel and its associated files.

3. After the boot process has completed, remove the newly installed kernel and, if necessary, set the default boot entry to the old kernel using the YaST *Boot Loader* module. For more information, refer to *Section 18.3, "Configuring the boot loader with YaST"*. However, doing this is not necessary because automated update tools normally modify it for you during the rollback process.

4. Reboot.

If this does not fix the problem, boot the computer using the installation media. After the machine has booted, continue with *Step 3*.

42.2.3 No graphical login

If the machine starts, but does not boot into the graphical login manager, anticipate problems either with the choice of the default systemd target or the configuration of the X Window System. To check the current systemd default target run the command **`sudo systemctl get-default`**. If the value returned is *not* `graphical.target`, run the command **`sudo systemctl isolate graphical.target`**. If the graphical login screen starts, log in and start *YaST* › *System* › *Services Manager* and set the *Default System Target* to *Graphical Interface*. From now on the system should boot into the graphical login screen.

If the graphical login screen does not start even if having booted or switched to the graphical target, your desktop or X Window software may be misconfigured or corrupted. Examine the log files at `/var/log/Xorg.*.log` for detailed messages from the X server as it attempted to start. If the desktop fails during start, it may log error messages to the system journal that can be queried with the command **journalctl** (see *Chapter 21, **journalctl**: query the systemd journal* for more information). If these error messages hint at a configuration problem in the X server, try to fix these issues. If the graphical system still does not come up, consider reinstalling the graphical desktop.

42.2.4 Root Btrfs partition cannot be mounted

If a `btrfs` root partition becomes corrupted, try the following options:

- Mount the partition with the `-o recovery` option.
- If that fails, run **btrfs-zero-log** on your root partition.

42.2.5 Force checking root partitions

If the root partition becomes corrupted, use the parameter `forcefsck` on the boot prompt. This passes the option `-f` (force) to the **fsck** command.

42.2.6 Disable swap to enable booting

When a swap device is not available and the system cannot enable it during boot, booting may fail. Try disabling all swap devices by appending the following options to the kernel command line:

```
systemd.device_wants_unit=off systemd.mask=swap.target
```

You may also try disabling specific swap devices:

```
systemd.mask=dev-sda1.swap
```

42.2.7 GRUB 2 fails during reboot on a dual-boot system

If GRUB 2 fails during reboot, disable the `Fast Boot` setting in the BIOS.

42.3 Login problems

Login problems occur when your system refuses to accept the user name and password, or accepts them but then fails to start the graphic desktop, produces errors, or drops to a command line, for example.

42.3.1 Valid user name and password combinations fail

This often occurs when the system is configured to use network authentication or directory services and cannot retrieve results from its configured servers. The `root` user is the only local user that can still log in to these machines. The following are common reasons a machine appears functional but cannot process logins correctly:

- The network is not working. For further directions on this, turn to *Section 42.4, “Network problems”*.
- DNS is not working at the moment (which prevents GNOME from working and the system from making validated requests to secure servers). One indication that this is the case is that the machine takes a long time to respond to any action. Find more information about this topic in *Section 42.4, “Network problems”*.
- If the system is configured to use Kerberos, the system's local time may have drifted past the accepted variance with the Kerberos server time (this is typically 300 seconds). If NTP (network time protocol) is not working properly or local NTP servers are not working, Kerberos authentication ceases to function because it depends on common clock synchronization across the network.
- The system's authentication configuration is misconfigured. Check the PAM configuration files involved for any typographical errors or misordering of directives. For additional background information about PAM and the syntax of the configuration files involved, refer to *Book “Security and Hardening Guide”, Chapter 2 “Authentication with PAM”*.
- The home partition is encrypted. Find more information about this topic in *Section 42.3.3, “Login to encrypted home partition fails”*.

In cases that do not involve external network problems, the solution is to log in as `root` and repair the configuration. If you cannot log in to the running system, reboot it into the rescue mode as outlined in *Procedure 18.3, “Entering rescue mode”*.

42.3.2 Valid user name and password not accepted

This is by far the most common problem users encounter, because there are many reasons this can occur. Depending on whether you use local user management and authentication or network authentication, login failures occur for different reasons.

Local user management can fail for the following reasons:

- The user may have entered the wrong password.
- The user's home directory containing the desktop configuration files is corrupted or write protected.
- There may be problems with the X Window System authenticating this particular user, especially if the user's home directory has been used with another Linux distribution before installing the current one.

To locate the reason for a local login failure, proceed as follows:

1. Check whether the user remembered their password correctly before you start debugging the whole authentication mechanism. If the user may have not have remembered their password correctly, use the YaST User Management module to change the user's password. Pay attention to the `Caps Lock` key and unlock it, if necessary.
2. Log in as `root` and check the system journal with **`journalctl -e`** for error messages of the login process and of PAM.
3. Try to log in from a console (using `Ctrl`–`Alt`–`F1`). If this is successful, the blame cannot be put on PAM, because it is possible to authenticate this user on this machine. Try to locate any problems with the X Window System or the GNOME desktop. For more information, refer to *Section 42.3.4, "GNOME desktop has issues"*.
4. If the user's home directory has been used with another Linux distribution, remove the `Xauthority` file in the user's home. Use a console login via `Ctrl`–`Alt`–`F1` and run **`rm .Xauthority`** as this user. This should eliminate X authentication problems for this user. Try graphical login again.
5. If the desktop could not start because of corrupt configuration files, proceed with *Section 42.3.4, "GNOME desktop has issues"*.

In the following, common reasons a network authentication for a particular user may fail on a specific machine are listed:

- The user may have entered the wrong password.
- The user name exists in the machine's local authentication files and is also provided by a network authentication system, causing conflicts.
- The home directory exists but is corrupt or unavailable. Perhaps it is write protected or is on a server that is inaccessible at the moment.
- The user does not have permission to log in to that particular host in the authentication system.
- The machine has changed host names, for whatever reason, and the user does not have permission to log in to that host.
- The machine cannot reach the authentication server or directory server that contains that user's information.
- There may be problems with the X Window System authenticating this particular user, especially if the user's home has been used with another Linux distribution before installing the current one.

To locate the cause of the login failures with network authentication, proceed as follows:

1. Check whether the user remembered their password correctly before you start debugging the whole authentication mechanism.
2. Determine the directory server which the machine relies on for authentication and make sure that it is up and running and properly communicating with the other machines.
3. Determine that the user's user name and password work on other machines to make sure that their authentication data exists and is properly distributed.
4. See if another user can log in to the misbehaving machine. If another user can log in without difficulty or if `root` can log in, log in and examine the system journal with the **`journalctl -e`** > file. Locate the time stamps that correspond to the login attempts and determine if PAM has produced any error messages.

5. Try to log in from a console (using `Ctrl`–`Alt`–`F1`). If this is successful, the problem is not with PAM or the directory server on which the user's home is hosted, because it is possible to authenticate this user on this machine. Try to locate any problems with the X Window System or the GNOME desktop. For more information, refer to *Section 42.3.4, “GNOME desktop has issues”*.

6. If the user's home directory has been used with another Linux distribution, remove the `Xauthority` file in the user's home. Use a console login via `Ctrl`–`Alt`–`F1` and run **`rm .Xauthority`** as this user. This should eliminate X authentication problems for this user. Try graphical login again.

7. If the desktop could not start because of corrupt configuration files, proceed with *Section 42.3.4, “GNOME desktop has issues”*.

42.3.3 Login to encrypted home partition fails

It is recommended to use an encrypted home partition for laptops. If you cannot log in to your laptop, the reason might be that your partition could not be unlocked.

During the boot time, you need to enter the passphrase to unlock your encrypted partition. If you do not enter it, the boot process continues, leaving the partition locked.

To unlock your encrypted partition, proceed as follows:

1. Switch to the text console with `Ctrl`–`Alt`–`F1`.

2. Become `root`.

3. Restart the unlocking process again with:

```
# systemctl restart home.mount
```

4. Enter your passphrase to unlock your encrypted partition.

5. Exit the text console and switch back to the login screen with `Alt`–`F7`.

6. Log in as usual.

42.3.4 GNOME desktop has issues

If you are experiencing issues with the GNOME desktop, there are several ways to troubleshoot the misbehaving graphical desktop environment. The recommended procedure described below offers the safest option to fix a broken GNOME desktop.

PROCEDURE 42.1: TROUBLESHOOTING GNOME

1. Launch YaST and switch to *Security and Users*.
2. Open the *User and Group Management* dialog and click *Add*.
3. Fill out the required fields and click *OK* to create a new user.
4. Log out and log in as the new user. This gives you a fresh GNOME environment.
5. Copy individual subdirectories from the `~/.local/` and `~/.config/` directories of the old user account to the respective directories of the new user account.
   Log out and log in again as the new user after every copy operation to check whether GNOME still works correctly.
6. Repeat the previous step until you find the configuration file that breaks GNOME.
7. Log in as the old user, and move the offending configuration file to a different location. Log out and log in again as the old user.
8. Delete the previously created user.

42.4 Network problems

Many problems of your system may be network-related, although the symptoms look different. For example, the reason for a system not allowing users to log in may be a network problem. This section introduces a simple checklist you can apply to identify the cause of any network problem encountered.

PROCEDURE 42.2: HOW TO IDENTIFY NETWORK PROBLEMS

When checking the network connection of your machine, proceed as follows:

1. If you use an Ethernet connection, check the hardware first. Make sure that your network cable is properly plugged into your computer and router (or hub, etc.). The control lights next to your Ethernet connector are normally both be active.

If the connection fails, check whether your network cable works with another machine. If it does, your network card causes the failure. If hubs or switches are included in your network setup, they may be faulty, as well.

2. If using a wireless connection, check whether the wireless link can be established by other machines. If not, contact the wireless network's administrator.

3. When you have checked your basic network connectivity, try to find out which service is not responding. Gather the address information of all network servers needed in your setup. Either look them up in the appropriate YaST module or ask your system administrator. The following list gives typical network servers involved in a setup together with the symptoms of an outage.

   DNS (name service)
   : A broken or malfunctioning name service affects the network's functionality in many ways. If the local machine relies on any network servers for authentication and these servers cannot be found because of name resolution issues, users would not even be able to log in. Machines in the network managed by a broken name server would not be able to “see” each other and communicate.

   NTP (time service)
   : A malfunctioning or broken NTP service could affect Kerberos authentication and X server functionality.

   NFS (file service)
   : If any application needs data stored in an NFS mounted directory, it cannot start or function properly if this service was down or misconfigured. In the worst case scenario, a user's personal desktop configuration would not come up if their home directory containing the `.gconf` subdirectory could not be found because of a faulty NFS server.

   Samba (file service)
   : If any application needs data stored in a directory on a faulty Samba server, it cannot start or function properly.

   NIS (user management)
   : If your SUSE Linux Enterprise Desktop system relies on a faulty NIS server to provide the user data, users cannot log in to this machine.

LDAP (user management)

If your SUSE Linux Enterprise Desktop system relies on a faulty LDAP server to provide the user data, users cannot log in to this machine.

Kerberos (authentication)

Authentication does not work and login to any machine fails.

CUPS (network printing)

Users cannot print.

4. Check whether the network servers are running and whether your network setup allows you to establish a connection:

Important: Limitations

The debugging procedure described below only applies to a simple network server/client setup that does not involve any internal routing. It assumes both server and client are members of the same subnet without the need for additional routing.

a. Use **`ping`** *`IP_ADDRESS/HOSTNAME`* (replace with the host name or IP address of the server) to check whether each one of them is up and responding to the network. If this command is successful, it tells you that the host you were looking for is up and running and that the name service for your network is configured correctly.
If ping fails with `destination host unreachable`, either your system or the desired server is not properly configured or down. Check whether your system is reachable by running **`ping`** *`IP address`* or *`YOUR_HOSTNAME`* from another machine. If you can reach your machine from another machine, it is the server that is not running or not configured correctly.
If ping fails with `unknown host`, the name service is not configured correctly or the host name used was incorrect. For further checks on this matter, refer to *Step 4.b*. If ping still fails, either your network card is not configured correctly or your network hardware is faulty.

b. Use **`host`** *`HOSTNAME`* to check whether the host name of the server you are trying to connect to is properly translated into an IP address and vice versa. If this command returns the IP address of this host, the name service is up and running. If the **`host`** command fails, check all network configuration files relating to name and address resolution on your host:

`/var/run/netconfig/resolv.conf`

This file is used to keep track of the name server and domain you are currently using. It is a symbolic link to `/run/netconfig/resolv.conf` and is usually automatically adjusted by YaST or DHCP. Make sure that this file has the following structure and all network addresses and domain names are correct:

```
search FULLY_QUALIFIED_DOMAIN_NAME
nameserver IPADDRESS_OF_NAMESERVER
```

This file can contain more than one name server address, but at least one of them must be correct to provide name resolution to your host. If needed, adjust this file using the YaST Network Settings module (Hostname/DNS tab).
If your network connection is handled via DHCP, enable DHCP to change host name and name service information by selecting *Set Hostname via DHCP* (can be set globally for any interface or per interface) and *Update Name Servers and Search List via DHCP* in the YaST Network Settings module (Hostname/DNS tab).

`/etc/nsswitch.conf`

This file tells Linux where to look for name service information. It should look like this:

```
 ...
hosts: files dns
networks: files dns
...
```

The `dns` entry is vital. It tells Linux to use an external name server. Normally, these entries are automatically managed by YaST, but it would be prudent to check.

If all the relevant entries on the host are correct, let your system administrator check the DNS server configuration for the correct zone information. If you have made sure that the DNS configuration of your host and the DNS server are correct, proceed with checking the configuration of your network and network device.

c. If your system cannot establish a connection to a network server and you have excluded name service problems from the list of possible culprits, check the configuration of your network card.
   Use the command `ip addr show` *`NETWORK_DEVICE`* to check whether this device was properly configured. Make sure that the `inet address` with the netmask (/*`MASK`*) is configured correctly. An error in the IP address or a missing bit in your network mask would render your network configuration unusable. If necessary, perform this check on the server as well.

d. If the name service and network hardware are properly configured and running, but certain external network connections still get long timeouts or fail entirely, use `traceroute` *`FULLY_QUALIFIED_DOMAIN_NAME`* (executed as `root`) to track the network route these requests are taking. This command lists any gateway (hop) that a request from your machine passes on its way to its destination. It lists the response time of each hop and whether this hop is reachable. Use a combination of traceroute and ping to track down the culprit and let the administrators know.

When you have identified the cause of your network trouble, you can resolve it yourself (if the problem is located on your machine) or let the system administrators of your network know about your findings so they can reconfigure the services or repair the necessary systems.

42.4.1 NetworkManager problems

If you have a problem with network connectivity, narrow it down as described in *Procedure 42.2, "How to identify network problems"*. If NetworkManager looks suspicious, proceed as follows to get logs providing hints on why NetworkManager fails:

1. Open a shell and log in as `root`.
2. Restart the NetworkManager:

```
> sudo systemctl restart NetworkManager
```

3. Open a Web page, for example, https://www.opensuse.org as normal user to see, if you can connect.
4. Collect any information about the state of NetworkManager in `/var/log/NetworkManager`.

For more information about NetworkManager, refer to *Chapter 31, Using NetworkManager*.

42.5 Data problems

Data problems are when the machine may or may not boot properly but, in either case, it is clear that there is data corruption on the system and that the system needs to be recovered. These situations call for a backup of your critical data, enabling you to recover the system state from before your system failed.

42.5.1 Managing partition images

Sometimes you need to perform a backup from an entire partition or even hard disk. Linux comes with the **`dd`** tool which can create an exact copy of your disk. Combined with **`gzip`** you save space.

PROCEDURE 42.3: BACKING UP AND RESTORING HARD DISKS

1. Start a Shell as user `root`.
2. Select your source device. Typically this is something like `/dev/sda` (labeled as *SOURCE*).
3. Decide where you want to store your image (labeled as *BACKUP_PATH*). It must be different from your source device. In other words: if you make a backup from `/dev/sda`, your image file must not to be stored under `/dev/sda`.
4. Run the commands to create a compressed image file:

```
# dd if=/dev/SOURCE | gzip > /BACKUP_PATH/image.gz
```

5. Restore the hard disk with the following commands:

```
# gzip -dc /BACKUP_PATH/image.gz | dd of=/dev/SOURCE
```

If you only need to back up a partition, replace the *SOURCE* placeholder with your respective partition. In this case, your image file can lie on the same hard disk, but on a different partition.

42.5.2 Using the rescue system

There are several reasons a system could fail to come up and run properly. A corrupted file system following a system crash, corrupted configuration files, or a corrupted boot loader configuration are the most common ones.

To help you to resolve these situations, SUSE Linux Enterprise Desktop contains a rescue system that you can boot. The rescue system is a small Linux system that can be loaded into a RAM disk and mounted as root file system, allowing you to access your Linux partitions from the outside. Using the rescue system, you can recover or modify any important aspect of your system.

- Manipulate any type of configuration file.
- Check the file system for defects and start automatic repair processes.
- Access the installed system in a “change root” environment.
- Check, modify, and re-install the boot loader configuration.
- Recover from a badly installed device driver or unusable kernel.
- Resize partitions using the parted command. Find more information about this tool at the GNU Parted Web site https://www.gnu.org/software/parted/parted.html.

The rescue system can be loaded from various sources and locations. The simplest option is to boot the rescue system from the original installation medium.

1. Insert the installation medium into your DVD drive.
2. Reboot the system.
3. At the boot screen, press F4 and choose *DVD-ROM*. Then choose *Rescue System* from the main menu.
4. Enter `root` at the `Rescue:` prompt. A password is not required.

If your hardware setup does not include a DVD drive, you can boot the rescue system from a network source. The following example applies to a remote boot scenario—if using another boot medium, such as a DVD, modify the `info` file accordingly and boot as you would for a normal installation.

1. Enter the configuration of your PXE boot setup and add the lines `install=PROTOCOL://INSTSOURCE` and `rescue=1`. If you need to start the repair system, use `repair=1` instead. As with a normal installation, *PROTOCOL* stands for any of the supported network protocols (NFS, HTTP, FTP, etc.) and *INSTSOURCE* for the path to your network installation source.

2. Boot the system using “Wake on LAN”, as described in *Book “Deployment Guide”, Chapter 14 “Preparing network boot environment”, Section 14.5 “Using wake-on-LAN for remote wakeups”*.

3. Enter `root` at the `Rescue:` prompt. A password is not required.

When you have entered the rescue system, you can use the virtual consoles that can be reached with Alt – F1 to Alt – F6 .

A shell and other useful utilities, such as the mount program, are available in the `/bin` directory. The `/sbin` directory contains important file and network utilities for reviewing and repairing the file system. This directory also contains the most important binaries for system maintenance, such as **fdisk**, **mkfs**, **mkswap**, **mount**, and **shutdown**, **ip** and **ss** for maintaining the network. The directory `/usr/bin` contains the vi editor, find, less, and SSH.

To see the system messages, either use the command **dmesg** or view the system log with **journalctl**.

42.5.2.1 Checking and manipulating configuration files

As an example for a configuration that might be fixed using the rescue system, imagine you have a broken configuration file that prevents the system from booting properly. You can fix this using the rescue system.

To manipulate a configuration file, proceed as follows:

1. Start the rescue system using one of the methods described above.

2. To mount a root file system located under `/dev/sda6` to the rescue system, use the following command:

```
> sudo mount /dev/sda6 /mnt
```

All directories of the system are now located under `/mnt`

3. Change the directory to the mounted root file system:

```
> sudo cd /mnt
```

4. Open the problematic configuration file in the vi editor. Adjust and save the configuration.

5. Unmount the root file system from the rescue system:

```
> sudo umount /mnt
```

6. Reboot the machine.

42.5.2.2 Repairing and checking file systems

Generally, file systems cannot be repaired on a running system. If you encounter serious problems, you may not even be able to mount your root file system and the system boot may end with a "kernel panic". In this case, the only way is to repair the system from the outside. The system contains the **`fsck`** utility to check and repair multiple file system types, such as `ext2`, `ext3`, `ext4`, `msdos`, and `vfat`. Use the `-t` option to specify which file system to check.

The following command checks all `ext4` file systems found in the `/etc/fstab` specification:

```
> sudo fsck -t ext4 -A
```

> **Tip**
>
> For Btrfs, you can use the **`btrfs check`** command found in the `btrfsprogs` package.
>
> Find topics about the Btrfs file system in the following places:
>
> - The Storage Administration Guide includes https://documentation.suse.com/sles/html/SLES-all/cha-filesystems.html#sec-filesystems-major-btrfs and https://documentation.suse.com/sles/html/SLES-all/cha-resize-fs.html#sec-resize-fs-btrfs sections.
> - The following article describes how to recover from Btrfs errors https://www.suse.com/support/kb/doc/?id=000018769.

- The following article includes links to multiple Btrfs-related topics https://www.suse.com/support/kb/doc/?id=000018779.
- The **man 8 btrfs-check** man page details all options of the **btrfs check** command.

42.5.2.3 Accessing the installed system

If you need to access the installed system from the rescue system, you need to do this in a *change root* environment. For example, to modify the boot loader configuration, or to execute a hardware configuration utility.

To set up a change root environment based on the installed system, proceed as follows:

1. **Tip: Import LVM volume groups**

   If you are using an LVM setup (refer to *Book "Deployment Guide", Chapter 7 "Expert Partitioner", Section 7.3 "LVM configuration"* for more general details), import all existing volume groups to be able to find and mount the device(s):

   ```
   rootvgimport -a
   ```

   Run **lsblk** to check which node corresponds to the root partition. It is `/dev/sda2` in our example:

   ```
   > lsblk
   NAME        MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
   sda           8:0    0 149,1G  0 disk
   ├─sda1        8:1    0     2G  0 part  [SWAP]
   ├─sda2        8:2    0    20G  0 part  /
   └─sda3        8:3    0   127G  0 part
     └─cr_home 254:0    0   127G  0 crypt /home
   ```

2. Mount the root partition from the installed system:

   ```
   > sudo mount /dev/sda2 /mnt
   ```

3. Mount `/proc`, `/dev`, and `/sys` partitions:

   ```
   > sudo mount -t proc none /mnt/proc
   ```

```
> sudo mount --rbind /dev /mnt/dev
> sudo mount --rbind /sys /mnt/sys
```

4. Now you can "change root" into the new environment, keeping the `bash` shell:

```
> chroot /mnt /bin/bash
```

5. Finally, mount the remaining partitions from the installed system:

```
> mount -a
```

6. Now you have access to the installed system. Before rebooting the system, unmount the partitions with **`umount`** `-a` and leave the "change root" environment with **`exit`**.

Warning: Limitations

Although you have full access to the files and applications of the installed system, there are some limitations. The kernel that is running is the one that was booted with the rescue system, not with the change root environment. It only supports essential hardware and it is not possible to add kernel modules from the installed system unless the kernel versions are identical. Always check the version of the currently running (rescue) kernel with **`uname -r`** and then find out if a matching subdirectory exists in the `/lib/modules` directory in the change root environment. If yes, you can use the installed modules, otherwise you need to supply their correct versions on other media, such as a flash disk. Most often the rescue kernel version differs from the installed one — then you cannot simply access a sound card, for example. It is also not possible to start a graphical user interface.

Also note that you leave the "change root" environment when you switch the console with `Alt`–`F1` to `Alt`–`F6`.

42.5.2.4 Modifying and re-installing the boot loader

Sometimes a system cannot boot because the boot loader configuration is corrupted. The start-up routines cannot, for example, translate physical drives to the actual locations in the Linux file system without a working boot loader.

To check the boot loader configuration and re-install the boot loader, proceed as follows:

1. Perform the necessary steps to access the installed system as described in *Section 42.5.2.3, "Accessing the installed system"*.

2. Check that the GRUB 2 boot loader is installed on the system. If not, install the package `grub2` and run

   ```
   > sudo grub2-install /dev/sda
   ```

3. Check whether the following files are correctly configured according to the GRUB 2 configuration principles outlined in *Chapter 18, The boot loader GRUB 2* and apply fixes if necessary.

   - `/etc/default/grub`
   - `/boot/grub2/device.map`
   - `/boot/grub2/grub.cfg` (this file is generated, do not edit)
   - `/etc/sysconfig/bootloader`

4. Re-install the boot loader using the following command sequence:

   ```
   > sudo grub2-mkconfig -o /boot/grub2/grub.cfg
   ```

5. Unmount the partitions, log out of the “change root” environment, and reboot the system:

   ```
   > umount -a
   exit
   reboot
   ```

42.5.2.5 Fixing kernel installation

A kernel update may introduce a new bug which can impact the operation of your system. For example a driver for a piece of hardware in your system may be faulty, which prevents you from accessing and using it. In this case, revert to the last working kernel (if available on the system) or install the original kernel from the installation media.

Tip: How to keep last kernels after update

To prevent failures to boot after a faulty kernel update, use the kernel multiversion feature and tell `libzypp` which kernels you want to keep after the update.

For example to always keep the last two kernels and the currently running one, add

```
multiversion.kernels = latest,latest-1,running
```

to the `/etc/zypp/zypp.conf` file. See *Chapter 27, Installing multiple kernel versions* for more information.

A similar case is when you need to re-install or update a broken driver for a device not supported by SUSE Linux Enterprise Desktop. For example when a hardware vendor uses a specific device, such as a hardware RAID controller, which needs a binary driver to be recognized by the operating system. The vendor typically releases a Driver Update Disk (DUD) with the fixed or updated version of the required driver.

In both cases you need to access the installed system in the rescue mode and fix the kernel related problem, otherwise the system may fail to boot correctly:

1. Boot from the SUSE Linux Enterprise Desktop installation media.
2. If you are recovering after a faulty kernel update, skip this step. If you need to use a driver update disk (DUD), press F6 to load the driver update after the boot menu appears, and choose the path or URL to the driver update and confirm with *Yes*.
3. Choose *Rescue System* from the boot menu and press Enter . If you chose to use DUD, you will be asked to specify where the driver update is stored.
4. Enter `root` at the `Rescue:` prompt. A password is not required.
5. Manually mount the target system and “change root” into the new environment. For more information, see *Section 42.5.2.3, “Accessing the installed system”*.
6. If using DUD, install/re-install/update the faulty device driver package. Always make sure the installed kernel version exactly matches the version of the driver you are installing.
   If fixing faulty kernel update installation, you can install the original kernel from the installation media with the following procedure.
   a. Identify your DVD device with **`hwinfo --cdrom`** and mount it with **`mount /dev/sr0 /mnt`**.
   b. Navigate to the directory where your kernel files are stored on the DVD, for example **`cd /mnt/suse/x86_64/`**.
   c. Install required `kernel-*`, `kernel-*-base`, and `kernel-*-extra` packages of your flavor with the **`rpm -i`** command.

7. Update configuration files and reinitialize the boot loader if needed. For more information, see *Section 42.5.2.4, “Modifying and re-installing the boot loader”*.

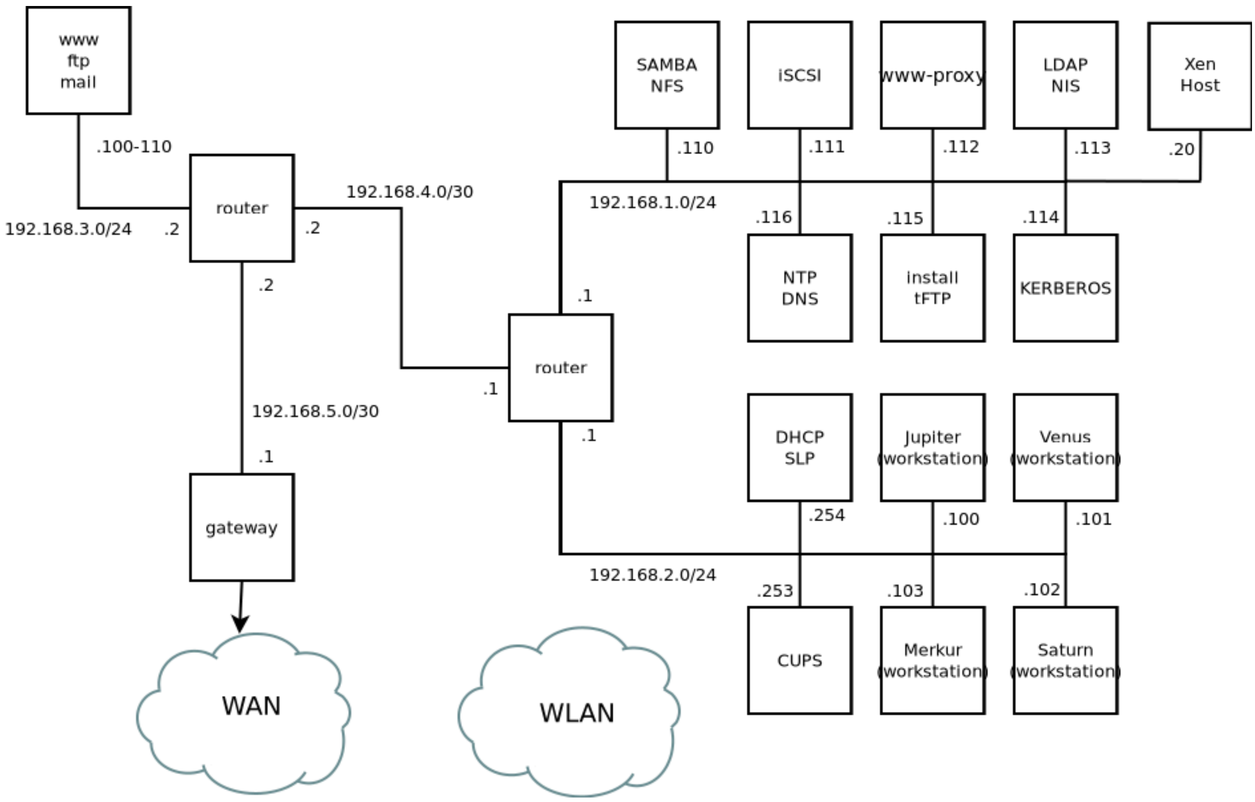
8. Remove any bootable media from the system drive and reboot.

A An example network

Revision History

2022-10-10

This example network is used across all network-related chapters of the SUSE® Linux Enterprise Desktop documentation.



B GNU licenses

Revision History

2023-02-03

This appendix contains the GNU Free Documentation License version 1.2.

GNU Free Documentation License

0. PREAMBLE

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

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

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

1. APPLICABILITY AND DEFINITIONS

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

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

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

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

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

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

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

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

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

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

2. VERBATIM COPYING

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

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

3. COPYING IN QUANTITY

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

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

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

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

4. MODIFICATIONS

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

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

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

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

D. Preserve all the copyright notices of the Document.

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

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

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

H. Include an unaltered copy of this License.

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

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

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

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

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

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

O. Preserve any Warranty Disclaimers.

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

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

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

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

5. COMBINING DOCUMENTS

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

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

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

6. COLLECTIONS OF DOCUMENTS

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

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

7. AGGREGATION WITH INDEPENDENT WORKS

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

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

8. TRANSLATION

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

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

9. TERMINATION

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

10. FUTURE REVISIONS OF THIS LICENSE

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

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

ADDENDUM: How to use this License for your documents

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

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.