



Reference

openSUSE Leap 15.2



Reference

openSUSE Leap 15.2

Publication Date: December 16, 2020

SUSE LLC

1800 South Novell Place

Provo, UT 84606

USA

<https://documentation.suse.com> ↗

Copyright © 2006– 2020 SUSE LLC and contributors. All rights reserved.

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

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

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

Contents

About This Guide xvii

- I ADVANCED ADMINISTRATION 1
- 1 YaST in Text Mode 2**
 - 1.1 Navigation in Modules 3
 - 1.2 Advanced Key Combinations 4
 - 1.3 Restriction of Key Combinations 5
 - 1.4 YaST Command Line Options 5
 - Installing Packages from the Command Line 6 • Working with Individual Modules 6 • Command Line Parameters of YaST Modules 6
- 2 Managing Software with Command Line Tools 31**
 - 2.1 Using Zypper 31
 - General Usage 31 • Using Zypper Subcommands 33 • Installing and Removing Software with Zypper 33 • Updating Software with Zypper 38 • Identifying Processes and Services Using Deleted Files 42 • Managing Repositories with Zypper 43 • Querying Repositories and Packages with Zypper 45 • Configuring Zypper 47 • Troubleshooting 47 • Zypper Rollback Feature on Btrfs File System 48 • For More Information 48
 - 2.2 RPM—the Package Manager 48
 - Verifying Package Authenticity 49 • Managing Packages: Install, Update, and Uninstall 49 • Delta RPM Packages 51 • RPM Queries 51 • Installing and Compiling Source Packages 54 • Compiling RPM Packages with build 56 • Tools for RPM Archives and the RPM Database 56

3 System Recovery and Snapshot Management with Snapper 58

- 3.1 Default Setup 59
 - Default Settings 60 • Types of Snapshots 60 • Directories That Are Excluded from Snapshots 61 • Customizing the Setup 62
- 3.2 Using Snapper to Undo Changes 65
 - Undoing YaST and Zypper Changes 66 • Using Snapper to Restore Files 71
- 3.3 System Rollback by Booting from Snapshots 73
 - Snapshots after Rollback 75 • Accessing and Identifying Snapshot Boot Entries 76 • Limitations 77
- 3.4 Enabling Snapper in User Home Directories 78
 - Installing pam_snapper and Creating Users 79 • Removing Users 79 • Manually Enabling Snapshots in Home Directories 80
- 3.5 Creating and Modifying Snapper Configurations 80
 - Managing Existing Configurations 82
- 3.6 Manually Creating and Managing Snapshots 85
 - Snapshot Metadata 85 • Creating Snapshots 87 • Modifying Snapshot Metadata 88 • Deleting Snapshots 88
- 3.7 Automatic Snapshot Clean-Up 90
 - Cleaning Up Numbered Snapshots 90 • Cleaning Up Timeline Snapshots 92 • Cleaning Up Snapshot Pairs That Do Not Differ 94 • Cleaning Up Manually Created Snapshots 94 • Adding Disk Quota Support 94
- 3.8 Showing Exclusive Disk Space Used by Snapshots 96
- 3.9 Frequently Asked Questions 97

4 Remote Graphical Sessions with VNC 99

- 4.1 The **vncviewer** Client 99
 - Connecting Using the vncviewer CLI 99 • Connecting Using the vncviewer GUI 100 • Notification of Unencrypted Connections 100

- 4.2 **Remmina: the Remote Desktop Client 101**
 - Installation 101 • Main Window 101 • Adding Remote Sessions 101 • Starting Remote Sessions 103 • Editing, Copying, and Deleting Saved Sessions 104 • Running Remote Sessions from the Command Line 104
- 4.3 **Configuring One-time Sessions on the VNC Server 105**
 - Available Configurations 106 • Initiating a One-time VNC Session 107 • Configuring One-time VNC Sessions 107
- 4.4 **Configuring Persistent VNC Server Sessions 108**
 - VNC Session Initiated Using `vncserver` 109 • VNC Session Initiated Using `vncmanager` 110
- 4.5 **Configuring Encryption on the VNC Server 113**
- 5 Expert Partitioner 116**
- 5.1 **Using the Expert Partitioner 116**
 - Partition Tables 118 • Partitions 119 • Editing a Partition 122 • Expert Options 124 • Advanced Options 125 • More Partitioning Tips 125 • Partitioning and LVM 128
- 5.2 **LVM Configuration 128**
 - Create Physical Volume 129 • Creating Volume Groups 129 • Configuring Logical Volumes 130
- 5.3 **Soft RAID 132**
 - Soft RAID Configuration 132 • Troubleshooting 133 • For More Information 134
- 6 Installing Multiple Kernel Versions 135**
- 6.1 **Enabling and Configuring Multiversion Support 135**
 - Automatically Deleting Unused Kernels 136 • Use Case: Deleting an Old Kernel after Reboot Only 137 • Use Case: Keeping Older Kernels as Fallback 137 • Use Case: Keeping a Specific Kernel Version 138
- 6.2 **Installing/Removing Multiple Kernel Versions with YaST 138**
- 6.3 **Installing/Removing Multiple Kernel Versions with Zypper 139**

- 6.4 Installing the Latest Kernel Version from the Repository
Kernel:HEAD 140

7 Graphical User Interface 142

- 7.1 X Window System 142
- 7.2 Installing and Configuring Fonts 142
 - Showing Installed Fonts 144 • Viewing Fonts 144 • Querying Fonts 144 • Installing Fonts 145 • Configuring the Appearance of Fonts 146
- 7.3 GNOME Configuration for Administrators 155
 - The dconf System 155 • System-wide Configuration 155 • More Information 156
- 7.4 Switching Between Intel and NVIDIA Optimus GPUs with SUSE Prime 156
 - Prerequisites 157 • Installing and Using SUSE Prime 157 • Installing NVIDIA Drivers 158

II SYSTEM 159

8 32-Bit and 64-Bit Applications in a 64-Bit System Environment 160

- 8.1 Runtime Support 160
- 8.2 Kernel Specifications 161

9 Introduction to the Boot Process 162

- 9.1 Terminology 162
- 9.2 The Linux Boot Process 163
 - The Initialization and Boot Loader Phase 163 • The Kernel Phase 164 • The init on initramfs Phase 167 • The systemd Phase 169

10 The systemd Daemon 170

- 10.1 The systemd Concept 170
 - What Is systemd 170 • Unit File 171

- 10.2 Basic Usage 172
 - Managing Services in a Running System 172 • Permanently Enabling/Disabling Services 174
- 10.3 System Start and Target Management 176
 - Targets Compared to Runlevels 176 • Debugging System Start-Up 179 • System V Compatibility 182
- 10.4 Managing Services with YaST 183
- 10.5 Customizing `systemd` 184
 - Customizing Unit Files 184 • Creating “Drop-in” Files 186 • Creating Custom Targets 187
- 10.6 Advanced Usage 187
 - Cleaning Temporary Directories 188 • System Log 188 • Snapshots 189 • Loading Kernel Modules 189 • Performing Actions before Loading a Service 190 • Kernel Control Groups (cgroups) 191 • Terminating Services (Sending Signals) 192 • Important Notes on the D-Bus Service 192 • Debugging Services 193
- 10.7 More Information 194
- 11 `journalctl`: Query the `systemd` Journal 195**
 - 11.1 Making the Journal Persistent 195
 - 11.2 **`journalctl`** Useful Switches 196
 - 11.3 Filtering the Journal Output 197
 - Filtering Based on a Boot Number 197 • Filtering Based on Time Interval 197 • Filtering Based on Fields 198
 - 11.4 Investigating `systemd` Errors 199
 - 11.5 Journald Configuration 200
 - Changing the Journal Size Limit 200 • Forwarding the Journal to `/dev/ttyX` 200 • Forwarding the Journal to Syslog Facility 201
 - 11.6 Using YaST to Filter the `systemd` Journal 201
 - 11.7 Viewing Logs in GNOME 202

12 The Boot Loader GRUB 2 203

- 12.1 Main Differences between GRUB Legacy and GRUB 2 203
- 12.2 Configuration File Structure 203
 - The File `/boot/grub2/grub.cfg` 204
 - The File `/etc/default/grub` 205
 - Scripts in `/etc/grub.d` 208
 - Mapping between BIOS Drives and Linux Devices 209
 - Editing Menu Entries during the Boot Procedure 210
 - Setting a Boot Password 211
- 12.3 Configuring the Boot Loader with YaST 212
 - Boot Loader Location and Boot Code Options 213
 - Adjusting the Disk Order 215
 - Configuring Advanced Options 216
- 12.4 Helpful GRUB 2 Commands 218
- 12.5 More Information 220

13 Basic Networking 221

- 13.1 IP Addresses and Routing 224
 - IP Addresses 224
 - Netmasks and Routing 224
- 13.2 IPv6—The Next Generation Internet 226
 - Advantages 227
 - Address Types and Structure 228
 - Coexistence of IPv4 and IPv6 232
 - Configuring IPv6 233
 - For More Information 234
- 13.3 Name Resolution 234
- 13.4 Configuring a Network Connection with YaST 236
 - Configuring the Network Card with YaST 236
- 13.5 NetworkManager 247
 - NetworkManager and **wicked** 247
 - NetworkManager Functionality and Configuration Files 248
 - Controlling and Locking Down NetworkManager Features 249
- 13.6 Configuring a Network Connection Manually 249
 - The **wicked** Network Configuration 249
 - Configuration Files 256
 - Testing the Configuration 267
 - Unit Files and Start-Up Scripts 271
- 13.7 Basic Router Setup 272

- 13.8 Setting Up Bonding Devices 274
 - Hotplugging of Bonding Slaves 277
- 13.9 Setting Up Team Devices for Network Teaming 278
 - Use Case: Load Balancing with Network Teaming 281 • Use Case: Failover with Network Teaming 282 • Use Case: VLAN over Team Device 283
- 13.10 Software-Defined Networking with Open vSwitch 285
 - Advantages of Open vSwitch 286 • Installing Open vSwitch 286 • Overview of Open vSwitch Daemons and Utilities 287 • Creating a Bridge with Open vSwitch 288 • Using Open vSwitch Directly with KVM 289 • Using Open vSwitch with libvirt 290 • For More Information 291
- 14 UEFI (Unified Extensible Firmware Interface) 292**
- 14.1 Secure Boot 292
 - Implementation on openSUSE Leap 293 • MOK (Machine Owner Key) 295 • Booting a Custom Kernel 296 • Using Non-Inbox Drivers 298 • Features and Limitations 299
- 14.2 For More Information 300
- 15 Special System Features 301**
- 15.1 Information about Special Software Packages 301
 - The bash Package and /etc/profile 301 • The cron Package 302 • Stopping Cron Status Messages 303 • Log Files: Package logrotate 303 • The **locate** Command 303 • The **ulimit** Command 304 • The **free** Command 305 • Man Pages and Info Pages 305 • Selecting Man Pages Using the **man** Command 305 • Settings for GNU Emacs 306
- 15.2 Virtual Consoles 307
- 15.3 Keyboard Mapping 307
- 15.4 Language and Country-Specific Settings 308
 - System Wide Locale Settings 309 • Some Examples 310 • Locale Settings in ~/.i18n 311 • Settings for Language Support 311 • For More Information 312

16 Dynamic Kernel Device Management with udev 313

- 16.1 The /dev Directory 313
- 16.2 Kernel uevents and udev 313
- 16.3 Drivers, Kernel Modules and Devices 314
- 16.4 Booting and Initial Device Setup 314
- 16.5 Monitoring the Running udev Daemon 315
- 16.6 Influencing Kernel Device Event Handling with udev Rules 316
 - Using Operators in udev Rules 318
 - Using Substitutions in udev Rules 319
 - Using udev Match Keys 320
 - Using udev Assign Keys 321
- 16.7 Persistent Device Naming 322
- 16.8 Files used by udev 323
- 16.9 For More Information 324

III SERVICES 325

17 SLP 326

- 17.1 The SLP Front-End **slptool** 326
- 17.2 Providing Services via SLP 327
 - Setting up an SLP Installation Server 329
- 17.3 For More Information 329

18 Time Synchronization with NTP 330

- 18.1 Configuring an NTP Client with YaST 331
 - NTP Daemon Start 331
 - Type of the Configuration Source 332
 - Configure Time Servers 332
- 18.2 Manually Configuring NTP in the Network 333
- 18.3 Configure chronyd at Runtime Using **chronyc** 334
- 18.4 Dynamic Time Synchronization at Runtime 334
- 18.5 Setting Up a Local Reference Clock 335

19 The Domain Name System 337

- 19.1 DNS Terminology 337
- 19.2 Installation 338
- 19.3 Configuration with YaST 338
 - Wizard Configuration 338 • Expert Configuration 341
- 19.4 Starting the BIND Name Server 349
- 19.5 The /etc/named.conf Configuration File 351
 - Important Configuration Options 352 • Logging 353 • Zone Entries 354
- 19.6 Zone Files 355
- 19.7 Dynamic Update of Zone Data 358
- 19.8 Secure Transactions 359
- 19.9 DNS Security 360
- 19.10 For More Information 361

20 DHCP 362

- 20.1 Configuring a DHCP Server with YaST 363
 - Initial Configuration (Wizard) 363 • DHCP Server Configuration (Expert) 367
- 20.2 DHCP Software Packages 372
- 20.3 The DHCP Server dhcpd 373
 - Clients with Fixed IP Addresses 374 • The openSUSE Leap Version 375
- 20.4 For More Information 376

21 Samba 377

- 21.1 Terminology 377
- 21.2 Installing a Samba Server 378
- 21.3 Starting and Stopping Samba 379

- 21.4 **Configuring a Samba Server 379**
 - Configuring a Samba Server with YaST 379 • Configuring the Server Manually 382
- 21.5 **Configuring Clients 386**
 - Configuring a Samba Client with YaST 386 • Mounting SMB1/CIFS Shares on Clients 386
- 21.6 **Samba as Login Server 388**
- 21.7 **Samba Server in the Network with Active Directory 389**
- 21.8 **Advanced Topics 390**
 - Transparent File Compression on Btrfs 391 • Snapshots 392
- 21.9 **For More Information 400**
- 22 Sharing File Systems with NFS 401**
- 22.1 **Overview 401**
- 22.2 **Installing NFS Server 402**
- 22.3 **Configuring NFS Server 403**
 - Exporting File Systems with YaST 403 • Exporting File Systems Manually 404 • NFS with Kerberos 407
- 22.4 **Configuring Clients 407**
 - Importing File Systems with YaST 408 • Importing File Systems Manually 409 • Parallel NFS (pNFS) 410
- 22.5 **For More Information 411**
- 23 On-Demand Mounting with Autofs 412**
- 23.1 **Installation 412**
- 23.2 **Configuration 412**
 - The Master Map File 412 • Map Files 414
- 23.3 **Operation and Debugging 415**
 - Controlling the autofs Service 415 • Debugging the Automounter Problems 416

23.4 Auto-Mounting an NFS Share 417

23.5 Advanced Topics 418

/net Mount Point 418 • Using Wild Cards to Auto-Mount
Subdirectories 418 • Auto-Mounting CIFS File System 419

24 The Apache HTTP Server 420

24.1 Quick Start 420

Requirements 420 • Installation 421 • Start 421

24.2 Configuring Apache 422

Apache Configuration Files 422 • Configuring Apache
Manually 425 • Configuring Apache with YaST 430

24.3 Starting and Stopping Apache 436

24.4 Installing, Activating, and Configuring Modules 438

Module Installation 439 • Activation and Deactivation 439 • Base and
Extension Modules 439 • Multiprocessing Modules 442 • External
Modules 444 • Compilation 445

24.5 Enabling CGI Scripts 445

Apache Configuration 446 • Running an Example Script 446 • CGI
Troubleshooting 447

24.6 Setting Up a Secure Web Server with SSL 448

Creating an SSL Certificate 448 • Configuring Apache with SSL 452

24.7 Running Multiple Apache Instances on the Same Server 454

24.8 Avoiding Security Problems 457

Up-to-Date Software 457 • DocumentRoot Permissions 457 • File System
Access 458 • CGI Scripts 458 • User Directories 458

24.9 Troubleshooting 459

24.10 For More Information 460

Apache 2.4 460 • Apache Modules 460 • Development 460

25 Setting Up an FTP Server with YaST 462

25.1 Starting the FTP Server 463

- 25.2 FTP General Settings 463
- 25.3 FTP Performance Settings 464
- 25.4 Authentication 464
- 25.5 Expert Settings 465
- 25.6 For More Information 465
- 26 Squid Caching Proxy Server 466**
- 26.1 Some Facts about Proxy Servers 466
 - Squid and Security 467 • Multiple Caches 467 • Caching Internet Objects 468
- 26.2 System Requirements 468
 - RAM 469 • CPU 469 • Size of the Disk Cache 469 • Hard Disk/SSD Architecture 470
- 26.3 Basic Usage of Squid 470
 - Starting Squid 470 • Checking Whether Squid Is Working 471 • Stopping, Reloading, and Restarting Squid 473 • Removing Squid 473 • Local DNS Server 474
- 26.4 The YaST Squid Module 475
- 26.5 The Squid Configuration File 475
 - General Configuration Options 476 • Options for Access Controls 479
- 26.6 Configuring a Transparent Proxy 481
- 26.7 Using the Squid Cache Manager CGI Interface (`cachemgr.cgi`) 482
- 26.8 Cache Report Generation with Calamaris 485
- 26.9 For More Information 485

IV MOBILE COMPUTERS 486

27 Mobile Computing with Linux 487

27.1 Laptops 487

Power Conservation 487 • Integration in Changing Operating Environments 488 • Software Options 490 • Data Security 495

27.2 Mobile Hardware 496

27.3 Mobile Devices (Smartphones and Tablets) 497

28 Using NetworkManager 498

28.1 Use Cases for NetworkManager 498

28.2 Enabling or Disabling NetworkManager 498

28.3 Configuring Network Connections 499

Managing Wired Network Connections 501 • Managing Wireless Network Connections 501 • Configuring Your Wi-Fi/Bluetooth Card as an Access Point 502 • NetworkManager and VPN 502

28.4 NetworkManager and Security 504

User and System Connections 504 • Storing Passwords and Credentials 505

28.5 Frequently Asked Questions 505

28.6 Troubleshooting 506

28.7 For More Information 507

29 Power Management 508

29.1 Power Saving Functions 508

29.2 Advanced Configuration and Power Interface (ACPI) 509

Controlling the CPU Performance 510 • Troubleshooting 510

29.3 Rest for the Hard Disk 512

29.4 Troubleshooting 513

CPU Frequency Does Not Work 513

A An Example Network 514

B GNU Licenses 515

B.1 GNU Free Documentation License 515

About This Guide

This manual gives you a general understanding of openSUSE® Leap. It is intended mainly for system administrators and home users with basic system administration knowledge. Check out the various parts of this manual for a selection of applications needed in everyday life and in-depth descriptions of advanced installation and configuration scenarios.

Advanced Administration

Learn about advanced administration tasks such as using YaST in text mode and managing software from the command line. Find out how to do system rollbacks with Snapper and how to use advanced storage techniques on openSUSE Leap.

System

Get an introduction to the components of your Linux system and a deeper understanding of their interaction.

Services

Learn how to configure the various network and file services that come with openSUSE Leap.

Mobile Computers

Get an introduction to mobile computing with openSUSE Leap, get to know the various options for wireless computing and power management.

1 Available Documentation



Note: Online Documentation and Latest Updates

Documentation for our products is available at <http://doc.opensuse.org/>, where you can also find the latest updates, and browse or download the documentation in various formats. The latest documentation updates are usually available in the English version of the documentation.

The following documentation is available for this product:

Book “Start-Up”

This manual will see you through your initial contact with openSUSE® Leap. Check out the various parts of this manual to learn how to install, use and enjoy your system.

Reference

Covers system administration tasks like maintaining, monitoring and customizing an initially installed system.

Book “Virtualization Guide”

Describes virtualization technology in general, and introduces libvirt—the unified interface to virtualization—and detailed information on specific hypervisors.

Book “AutoYaST Guide”

AutoYaST is a system for unattended mass deployment of openSUSE Leap systems using an AutoYaST profile containing installation and configuration data. The manual guides you through the basic steps of auto-installation: preparation, installation, and configuration.

Book “Security and Hardening Guide”

Introduces basic concepts of system security, covering both local and network security aspects. Shows how to use the product inherent security software like AppArmor, SELinux, or the auditing system that reliably collects information about any security-relevant events. Supports the administrator with security-related choices and decisions in installing and setting up a secure SUSE Linux Enterprise Server and additional processes to further secure and harden that installation.

Book “System Analysis and Tuning Guide”

An administrator's guide for problem detection, resolution and optimization. Find how to inspect and optimize your system by means of monitoring tools and how to efficiently manage resources. Also contains an overview of common problems and solutions and of additional help and documentation resources.

Book “GNOME User Guide”

Introduces the GNOME desktop of openSUSE Leap. It guides you through using and configuring the desktop and helps you perform key tasks. It is intended mainly for end users who want to make efficient use of GNOME as their default desktop.

The release notes for this product are available at <https://www.suse.com/releasesnotes/>.

2 Giving Feedback

Your feedback and contributions to this documentation are welcome! Several channels are available:

Bug Reports

Report issues with the documentation at <https://bugzilla.opensuse.org/>. To simplify this process, you can use the *Report Documentation Bug* links next to headlines in the HTML version of this document. These preselect the right product and category in Bugzilla and add 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, use the *Edit Source* links next to headlines in the HTML version of this document. They take you to the source code on GitHub, where you can open a pull request. A GitHub account is required.

For more information about the documentation environment used for this documentation, see the repository's README (<https://github.com/SUSE/doc-sle/blob/master/README.adoc>).

Mail

Alternatively, you can report errors and send feedback concerning the documentation to doc-team@suse.com. Make sure to include the document title, the product version and the publication date of the documentation. Refer to the relevant section number and title (or include the URL) and provide a concise description of the problem.

Help

If you need further help on openSUSE Leap, see <https://en.opensuse.org/Portal:Support>.

3 Documentation Conventions

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

- /etc/passwd: directory names and file names
- PLACEHOLDER: replace PLACEHOLDER with the actual value
- PATH: the environment variable PATH
- ls, --help: commands, options, and parameters

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

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

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

```
tux > command
```

- Notices



Warning: Warning Notice

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



Important: Important Notice

Important information you should be aware of before proceeding.



Note: Note Notice

Additional information, for example about differences in software versions.



Tip: Tip Notice

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

4 Source Code

The source code of openSUSE Leap is publicly available. Refer to http://en.opensuse.org/Source_code for download links and more information.

5 Acknowledgments

With a lot of voluntary commitment, the developers of Linux cooperate on a global scale to promote the development of Linux. We thank them for their efforts—this distribution would not exist without them. Special thanks, of course, goes to Linus Torvalds.

I Advanced Administration

- 1 YaST in Text Mode [2](#)
- 2 Managing Software with Command Line Tools [31](#)
- 3 System Recovery and Snapshot Management with Snapper [58](#)
- 4 Remote Graphical Sessions with VNC [99](#)
- 5 Expert Partitioner [116](#)
- 6 Installing Multiple Kernel Versions [135](#)
- 7 Graphical User Interface [142](#)

1 YaST in Text Mode

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.

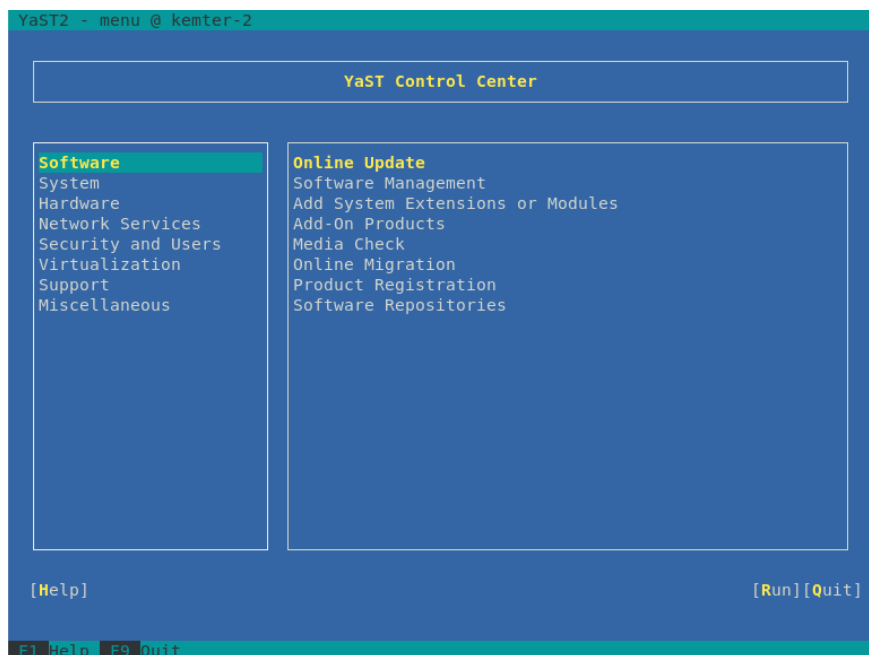


FIGURE 1.1: MAIN WINDOW OF YAST IN TEXT MODE

To launch the ncurses-based version of YaST, open the terminal and run the `sudo yast2` command. Use the `Tab` 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+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.

1.1 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 1.3, "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 various buttons. Available function key combinations (**FX**) are shown in the bottom line of the YaST screen. Which function keys are actually 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

Some 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 to open or close tree items. In the ncurses mode, 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 or . Alternatively, use the *Actions* menu to select the needed status change (install, delete, update, taboo, or lock).

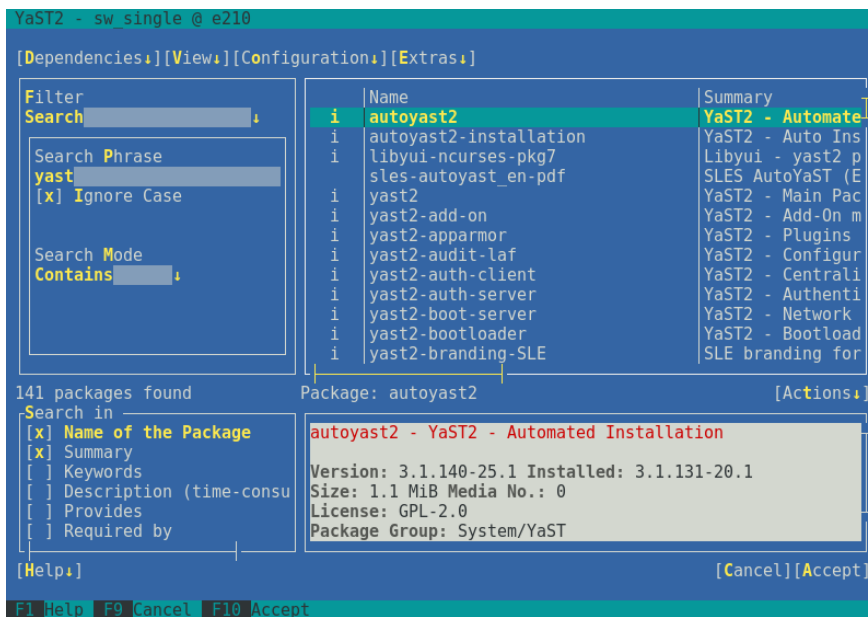


FIGURE 1.2: THE SOFTWARE INSTALLATION MODULE

1.2 Advanced Key Combinations

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

-

List advanced hotkeys.

-

Change color schema.

Ctrl-****

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.

1.3 Restriction of Key Combinations

If your window manager uses global **Alt** combinations, the **Alt** combinations in YaST might 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 might 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.

1.4 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:

```
tux > sudo yast -h
```

1.4.1 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:

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

or

```
tux > 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 2.1, "Using Zypper"](#).

1.4.2 Working with Individual Modules

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

```
tux > sudo yast module_name
```

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

1.4.3 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:

```
tux > 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.

1.4.3.1 Common YaST Module Commands

All YaST modules support the following commands:

help

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

```
tux > sudo yast lan help
```

longhelp

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

```
tux > sudo yast lan longhelp
```

xmlhelp

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

```
tux > 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.

1.4.3.2 yast add-on

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

```
tux > 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://`.

1.4.3.3 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:

```
tux > 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:

```
tux > 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**.

1.4.3.4 **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:

```
tux > 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**.

1.4.3.5 **yast dns-server**

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

acls

Displays access control list settings:

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

dnsrecord

Configures zone resource records:

```
tux > 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:

```
tux > sudo yast dns-server forwarders add ip=10.0.0.100
tux > 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:

```
tux > 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:

```
tux > 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:

```
tux > 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:

```
tux > 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:

```
tux > 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:

```
tux > 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:

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

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

1.4.3.6 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:

```
tux > 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:

```
tux > 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    | /           |       |
[...]
```

1.4.3.7 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.

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

access

Configures access permissions:

```
tux > 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:

```
tux > 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:

```
tux > 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):

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

idle-time

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

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

logging

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

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

max_clients

Specifies the maximum number of concurrently connected clients:

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

max_clients_ip

Specifies the maximum number of concurrently connected clients via IP:

```
tux > 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):

```
tux > 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):

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

port_range

Specifies the port range for passive connection replies:

```
tux > 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:

```
tux > 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:

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

welcome_message

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

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

1.4.3.8 **yast http-server**

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

configure

Configures the HTTP server host settings:

```
tux > 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:

```
tux > 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:

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

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

modules

Controls the Apache2 server modules:

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

1.4.3.9 `yast kdump`

Configures `kdump` settings. For more information on `kdump`, refer to the *Book "System Analysis and Tuning Guide", Chapter 17 "Kexec and Kdump", Section 17.7 "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]`.

```
tux > 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':

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

`dumplevel`

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

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

`dumptarget`

Specifies the destination for saving dump images:

```
tux > sudo kdump dumptarget taget=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:

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

keepolddumps

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

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

kernelcommandline

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

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

kernelcommandlineappend

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

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

notificationcc

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

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

notificationto

Specifies an e-mail address for sending notification messages:

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

show

Displays kdump settings:

```
tux > 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:

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

smtpserver

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

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

smtpuser

Specifies the SMTP user name used for sending notification messages:

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

startup

Enables or disables start-up options:

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

1.4.3.10 **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:

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

summary

Displays the current keyboard configuration.

1.4.3.11 `yast lan`

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

add

Configures a new network card:

```
tux > 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:

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

edit

Changes the configuration of an existing network card:

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

list

Displays a summary of network card configuration:

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

1.4.3.12 `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:

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

1.4.3.13 **yast mail**

Displays the configuration of the mail system:

```
tux > sudo yast mail summary
```

1.4.3.14 **yast nfs**

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

add

Adds a new NFS mount:

```
tux > 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:

```
tux > 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:

```
tux > 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:

```
tux > 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
```

1.4.3.15 `yast nfs-server`

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

add

Adds a directory to export:

```
tux > 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:

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

set

Specifies additional parameters for the NFS server:

```
tux > 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:

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

stop

Stops the NFS server service:

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

summary

Displays a summary of the NFS server configuration:

```
tux > 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.
```

1.4.3.16 `yast nis`

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

configure

Changes global settings of a NIS client:

```
tux > 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:

```
tux > sudo yast nis disable
```

enable

Enables your machine as NIS client:

```
tux > 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:

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

summary

Displays a configuration summary of a NIS client.

1.4.3.17 `yast nis-server`

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

master

Configures a NIS master server:

```
tux > 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 slave server:

```
tux > 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:

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

summary

Displays a configuration summary of a NIS server:

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

1.4.3.18 **yast proxy**

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

authentication

Specifies the authentication options for proxy:

```
tux > 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:

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

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

summary

Displays proxy settings.

1.4.3.19 `yast rdp`

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

allow

Allows remote access to the server's desktop:

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

list

Displays the remote desktop configuration summary.

1.4.3.20 `yast samba-client`

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

configure

Changes global settings of Samba:

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

isdomainmember

Checks whether the machine is a member of a domain:

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

joindomain

Makes the machine a member of a domain:

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

winbind

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

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

1.4.3.21 `yast samba-server`

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

backend

Specifies the back-end for storing user information:

```
tux > 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:

```
tux > 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:

```
tux > 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:

```
tux > 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`):

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

share

Manipulates a single Samba share:

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

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

1.4.3.22 **yast security**

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

level

Specifies the security level of the host:

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

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

set

Sets the value of a specific option:

```
tux > 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:

```
sudo yast security summary
```

1.4.3.23 **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.

```
tux > 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:

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

modules

Lists all available sound kernel modules:

```
tux > 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:

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

remove

Removes a configured sound card:

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

set

Specifies new values for a sound card:

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

show

Displays detailed information about a sound card:

```
tux > 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:

```
tux > sudo yast sound summary
```

volume

Specifies the volume level of a sound card:

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

1.4.3.24 `yast sysconfig`

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

clear

Sets empty value to a variable:

```
tux > 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:

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

details

Displays detailed information about a variable:

```
tux > 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:

```
tux > 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:

```
tux > sudo yast sysconfig set DISPLAYMANAGER=gnome
```



Tip: Variable in Multiple Files

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

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

1.4.3.25 yast tftp-server

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

directory

Specifies the directory of the TFTP server:

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

status

Controls the status of the TFTP server service:

```
tux > sudo yast tftp-server status disable
```

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

1.4.3.26 `yast timezone`

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

list

Lists all available time zones grouped by region:

```
tux > 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:

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

summary

Displays the time zone configuration summary:

```
tux > 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
```

1.4.3.27 `yast users`

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

add

Adds a new user:

```
tux > 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:

```
tux > 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:

```
tux > 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:

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

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

show

Displays details about a user:

```
tux > 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**.

2 Managing Software with Command Line Tools

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 *Book "Start-Up", Chapter 10 "Installing or Removing Software", Section 10.1 "Definition of Terms"*.

2.1 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.

2.1.1 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:

```
tux > sudo zypper patch
```

Global Options

Additionally, you can choose from one or more global options by typing them immediately before the command:

```
tux > 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:

```
tux > 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, license 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*:

```
tux > sudo zypper install mplayer
```

Some options also require a single argument. The following command will list all known patterns:

```
tux > 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:

```
tux > 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.

```
tux > 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.

```
tux > sudo zypper --userdata STRING patch
```

2.1.2 Using Zypper Subcommands

Zypper subcommands are executables that are stored in the `zypper_execdir`, `/usr/lib/zypper/commands`. If a subcommand is not found in the `zypper_execdir`, Zypper automatically searches the rest of your `$PATH` for it. This enables writing your own local extensions and storing them in userspace.

Executing subcommands in the Zypper shell, and using global Zypper options are not supported.

List your available subcommands:

```
tux > 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

<none>
```

View the help screen for a subcommand:

```
tux > zypper help appstream-cache
```

2.1.3 Installing and Removing Software with Zypper

To install or remove packages, use the following commands:

```
tux > sudo zypper install PACKAGE_NAME
tux > 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.

2.1.3.1 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

```
tux > sudo zypper install MozillaFirefox
```

By Exact Package Name and Version Number

```
tux > sudo zypper install MozillaFirefox-52.2
```

By Repository Alias and Package Name

```
tux > 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”:

```
tux > 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:

```
tux > 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`:

```
tux > 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:

```
tux > 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).

```
tux > sudo zypper install 'firefox>=74.2'
```

- You can also combine a hardware architecture and version requirement:

```
tux > 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:

```
tux > sudo zypper install /tmp/install/MozillaFirefox.rpm
tux > sudo zypper install http://download.example.com/MozillaFirefox.rpm
```

2.1.3.2 Combining Installation and Removal of Packages

To install and remove packages simultaneously, use the `+/-` modifiers. To install `emacs` and simultaneously remove `vim`, use:

```
tux > sudo zypper install emacs -vim
```

To remove `emacs` and simultaneously install `vim`, use:

```
tux > 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 `--`:

```
tux > sudo zypper install -emacs +vim      # Wrong
tux > sudo zypper install vim -emacs      # Correct
```



```
tux > sudo zypper install -- -emacs +vim    # Correct
tux > sudo zypper remove emacs +vim        # Correct
```

2.1.3.3 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:

```
tux > sudo zypper rm --clean-deps PACKAGE_NAME
```

2.1.3.4 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:

```
tux > sudo zypper --non-interactive install PACKAGE_NAME
```

This option allows the use of Zypper in scripts and cron jobs.

2.1.3.5 Installing or Downloading Source Packages

To install the corresponding source package of a package, use:

```
tux > 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`:

```
tux > sudo zypper source-install -D PACKAGE_NAME
```

To install only the build dependencies use `-d`.

```
tux > 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 2.1.6, “Managing Repositories with Zypper”](#) for details on repository management.

A list of all source packages available in your repositories can be obtained with:

```
tux > zypper search -t srcpackage
```

You can also download source packages for all installed packages to a local directory. To download source packages, use:

```
tux > 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.

2.1.3.6 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:

```
tux > 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 openSUSE Leap 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.

2.1.3.7 Utilities

To verify whether all dependencies are still fulfilled and to repair missing dependencies, use:

```
tux > 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:

```
tux > 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.

2.1.4 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 openSUSE Leap is discussed in *Book “Start-Up”, Chapter 13 “Upgrading the System and System Changes”*.

2.1.4.1 Installing All Needed Patches

To install all officially released patches that apply to your system, run:

```
tux > 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 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:

```
tux > sudo zypper patch --with-update
```

To install also optional patches, use:

```
tux > sudo zypper patch --with-optional
```

To install all patches relating to a specific Bugzilla issue, use:

```
tux > sudo zypper patch --bugzilla=NUMBER
```

To install all patches relating to a specific CVE database entry, use:

```
tux > sudo zypper patch --cve=NUMBER
```

For example, to install a security patch with the CVE number CVE-2010-2713, execute:

```
tux > sudo zypper patch --cve=CVE-2010-2713
```

To install only patches which affect Zypper and the package management itself, use:

```
tux > 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.

2.1.4.2 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:

```
tux > 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 list-patches:

```
tux > zypper list-patches
```

```

Repository | Name                | Category | Severity | Interactive | Status | S>
-----+-----+-----+-----+-----+-----+----->
Update     | openSUSE-2019-828 | security | moderate | ---         | needed | S>

Found 1 applicable patch:
1 patch needed (1 security patch)

```

Note the new Since column. From Zypper 1.14.36, this shows when a patch was installed.

List of All Patches

To list all patches available for openSUSE Leap, 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:

```
tux > 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:

```
tux > zypper list-patches --bugzilla=CVE-2016-2315,CVE-2016-2324
```

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:

```

tux > 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
[...]

```

2.1.4.3 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 (while maintaining system integrity), use:

```
tux > sudo zypper update
```

To update individual packages, specify the package with either the update or install command:

```
tux > sudo zypper update PACKAGE_NAME  
tux > sudo zypper install PACKAGE_NAME
```

A list of all new installable packages can be obtained with the command:

```
tux > 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:

```
tux > 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.

2.1.4.4 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:

```
tux > sudo zypper packages --orphaned
```

With this list, you can decide if a package is still needed or can be removed safely.

2.1.5 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:

```
tux > 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.

```
tux > 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 10, The systemd Daemon](#).

2.1.6 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:

```
tux > zypper repos
```

The result will look similar to the following output:

EXAMPLE 2.1: ZYPPER—LIST OF KNOWN REPOSITORIES

```

tux > zypper repos
# | Alias | Name | Enabled | GPG Check | Refresh
-----+-----+-----+-----+-----+-----
1 | Leap-15.1-Main | Main (OSS) | Yes | ( r ) Yes | Yes
2 | Leap-15.1-Update | Update (OSS) | Yes | ( r ) Yes | Yes
3 | Leap-15.1-NOSS | Main (NON-OSS) | Yes | ( r ) Yes | Yes
4 | Leap-15.1-Update-NOSS | Update (NON-OSS) | Yes | ( r ) Yes | Yes
[...]
```


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:

```
tux > zypper repos -d
```

2.1.6.1 Adding Repositories

To add a repository, run

```
tux > sudo zypper addrepo URI ALIAS
```

URI can either be an Internet repository, a network resource, a directory or a CD or DVD (see http://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.

2.1.6.2 Refreshing Repositories

zypper enables you to fetch changes in packages from configured repositories. To fetch the changes, run:

```
tux > 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:

```
tux > sudo zypper --plus-content refresh
```

This option fetches changes in repositories, but keeps the disabled repositories in the same state—disabled.

2.1.6.3 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 `Leap-42.3-NOSS` from *Example 2.1, “Zypper—List of Known Repositories”*, use one of the following commands:

```
tux > sudo zypper removerepo 4
tux > sudo zypper removerepo "Leap-42.3-NOSS"
```

2.1.6.4 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:

```
tux > 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`:

```
tux > sudo zypper renamerepo 'Mozilla Firefox' firefox
```

2.1.7 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:

```
tux > zypper products
tux > zypper patterns
tux > zypper packages
```

```
tux > zypper patches
```

To query all repositories for certain packages, use `search`. To get information regarding particular packages, use the `info` command.

2.1.7.1 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`

```
tux > zypper search "fire"
```

Simple search for the exact package `MozillaFirefox`

```
tux > zypper search --match-exact "MozillaFirefox"
```

Also search in package descriptions and summaries

```
tux > zypper search -d fire
```

Only display packages not already installed

```
tux > zypper search -u fire
```

Display packages containing the string `fir` not followed by `e`

```
tux > zypper se "/fir[^e]/"
```

2.1.7.2 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:

```
tux > 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.

2.1.7.3 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**:

```
tux > zypper info --requires MozillaFirefox
```

2.1.8 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.

2.1.9 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:

```
tux > sudo zypper refresh
```

If that does not help, try

```
tux > sudo zypper refresh -fdb
```

This forces a complete refresh and rebuild of the database, including a forced download of raw metadata.

2.1.10 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 3, System Recovery and Snapshot Management with Snapper* for more information.

2.1.11 For 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 http://en.opensuse.org/SDB:Zypper_usage. A list of software changes for the latest openSUSE Leap version can be found at http://en.opensuse.org/openSUSE:Zypper_versions.

2.2 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`.

2.2.1 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.

2.2.2 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`.

2.2.3 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:

```
tux > 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:

```
tux > sudo applydeltarpm new.delta.rpm new.rpm
```

To derive it from the old RPM without accessing the file system, use the `-r` option:

```
tux > sudo applydeltarpm -r old.rpm new.delta.rpm new.rpm
```

See </usr/share/doc/packages/deltarpm/README> for technical details.

2.2.4 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 2.1, “The Most Important RPM Query Options”](#).

TABLE 2.1: THE MOST IMPORTANT RPM QUERY OPTIONS

| | |
|-----------------|---------------------|
| <code>-i</code> | Package information |
| <code>-l</code> | File list |

| | |
|-------------------------------|--|
| <u>-f FILE</u> | Query the package that contains the file <u>FILE</u> (the full path must be specified with <u>FILE</u>) |
| <u>-s</u> | File list with status information (implies <u>-l</u>) |
| <u>-d</u> | List only documentation files (implies <u>-l</u>) |
| <u>-c</u> | List only configuration files (implies <u>-l</u>) |
| <u>--dump</u> | File list with complete details (to be used with <u>-l</u> , <u>-c</u> , or <u>-d</u>) |
| <u>--provides</u> | List features of the package that another package can request with <u>--requires</u> |
| <u>--requires</u> , <u>-R</u> | Capabilities the package requires |
| <u>--scripts</u> | Installation scripts (preinstall, postinstall, uninstall) |

For example, the command `rpm -q -i wget` displays the information shown in *Example 2.2*, “`rpm -q -i wget`”.

EXAMPLE 2.2: `rpm -q -i wget`

```
Name       : wget
Name       : wget
Version    : 1.19.5
Release    : lp151.4.1
Architecture: x86_64
Install Date: Tue 30 Jul 2019 02:26:21 PM PDT
Group      : Productivity/Networking/Web/Utilities
Size       : 2881903
License    : GPL-3.0+
Signature  : RSA/SHA256, Thu 11 Apr 2019 02:23:42 AM PDT, Key ID b88b2fd43dbdc284
Source RPM : wget-1.19.5-lp151.4.1.src.rpm
Build Date : Thu 11 Apr 2019 02:23:27 AM PDT
Build Host : cloud114
Relocations: (not relocatable)
Packager   : https://bugs.opensuse.org
Vendor     : openSUSE
URL        : https://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: openSUSE Leap 15.1
```

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:

```
tux > 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 2.3, “Script to Search for Packages”*. Pass the partial file name to the script shown as a parameter when running it.

EXAMPLE 2.3: 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 2.2: RPM VERIFY OPTIONS

| | |
|----------|----------------------------------|
| <u>S</u> | MD5 check sum |
| <u>S</u> | File size |
| <u>L</u> | Symbolic link |
| <u>T</u> | Modification time |
| <u>D</u> | Major and minor device numbers |
| <u>U</u> | Owner |
| <u>G</u> | Group |
| <u>M</u> | 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):

```
tux > 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 --rebuilddb. 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_RPMDDB_BACKUPS (default: 5) in /etc/sysconfig/backup. The size of a single backup is approximately 1 MB for 1 GB in /usr.

2.2.5 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 -bX /usr/src/packages/SPECS/wget.spec starts the compilation. X is a wildcard 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, the `BuildRoot` directive in the spec file is deprecated since openSUSE Leap 42.1. If you still need this feature, use the `--buildroot` option as a workaround.

2.2.6 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`:

```
root # cd /usr/src/packages/SOURCES/  
root # mv ../SPECS/wget.spec .  
root # 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.

2.2.7 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 *Book "Start-Up", Chapter 10 "Installing or Removing Software"*.

3 System Recovery and Snapshot Management with Snapper

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 3.3, “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 3.2, “Using Snapper to Undo Changes”* for details.
- Restore files from previous snapshots. See *Section 3.2.2, “Using Snapper to Restore Files”* for details.
- Do a system rollback by booting from a snapshot. See *Section 3.3, “System Rollback by Booting from Snapshots”* for details.
- Manually create and manage snapshots, within the running system. See *Section 3.6, “Manually Creating and Managing Snapshots”* for details.

3.1 Default Setup

Snapper on openSUSE Leap is set up as an undo and recovery tool for system changes. By default, the root partition (`/`) of openSUSE Leap 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:

```
tux > sudo snapper -c root create-config /
```

Afterward enable the different snapshot types as described in [Section 3.1.4.1, “Disabling/Enabling Snapshots”](#).

Note that on a `Btrfs` root file system, snapshots require a file system with subvolumes set up 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 3.1.4.4, “Controlling Snapshot Archiving”](#) for details.

3.1.1 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

3.1.2 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. Old snapshots are automatically deleted. By default, the first snapshot of the last ten days, months, and years are kept. Timeline snapshots are disabled by default.

Installation Snapshots

Whenever one or more packages are installed with YaST or Zypper, a pair of snapshots is created: one before the installation starts (“Pre”) and another one after the installation has finished (“Post”). In case an important system component such as the kernel has been installed, the snapshot pair is marked as important (important=yes). Old snapshots are automatically deleted. By default the last ten important snapshots and the last ten “regular” (including administration snapshots) snapshots are kept. Installation snapshots are enabled by default.

Administration Snapshots

Whenever you administrate the system with YaST, a pair of snapshots is created: one when a YaST module is started (“Pre”) and another when the module is closed (“Post”). Old snapshots are automatically deleted. By default the last ten important snapshots and the last ten “regular” snapshots (including installation snapshots) are kept. Administration snapshots are enabled by default.

3.1.3 Directories That Are Excluded from Snapshots

Some 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.

3.1.4 Customizing the Setup

openSUSE Leap 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.

3.1.4.1 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"`

Timeline snapshots are enabled by default, except for the root partition.

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.

3.1.4.2 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:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <snapper-zypp-plugin-conf>
3 <solvables>
4 <solvable match="w" ① important="true" ②>kernel-* ③</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> ④
10 </solvables>
11 </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 will also be 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 will also be 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>

```

3.1.4.3 Creating and Mounting New Subvolumes

Creating a new subvolume underneath the `/` hierarchy and permanently mounting it is supported. Such a subvolume will be 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.

openSUSE Leap 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`.

```
tux > sudo mount /dev/sda2 -o subvol=@ /mnt
tux > sudo btrfs subvolume create /mnt/usr/important
tux > 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`.

3.1.4.4 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 will be deleted. A minimum of four important and two regular snapshots are always kept.

Refer to [Section 3.5.1, “Managing Existing Configurations”](#) for instructions on how to change these values.

3.1.4.5 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 [Section 5.2, “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, ext4 or xfs are valid values for FILESYSTEM. Example:

```
tux > sudo snapper -c lvm create-config --fstype="lvm(xfs)" /thin_lvm
```

You can adjust this configuration according to your needs as described in [Section 3.5.1, “Managing Existing Configurations”](#).

3.2 Using Snapper to Undo Changes

Snapper on openSUSE Leap 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 3.1.4.1, “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 3.3, “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 3.3, “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, some 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.

3.2.1 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 3.1: 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.

- 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).

Snapshots

Current Configuration

| ID | Type | Start Date | End Date | Description | User Data |
|---------|------------|---------------------|---------------------|-----------------------|---------------|
| 1 | Single | 2016-06-17 17:20:05 | | first root filesystem | |
| 2 | Single | 2016-06-17 17:27:05 | | after installation | important=yes |
| 3 & 4 | Pre & Post | 2016-07-25 11:34:14 | 2016-07-25 11:46:36 | yast online_update | |
| 5 & 6 | Pre & Post | 2016-07-25 11:51:27 | 2016-07-25 11:53:35 | yast sw_single | |
| 7 & 8 | Pre & Post | 2016-07-25 11:53:37 | 2016-07-25 11:56:53 | yast sw_single | |
| 9 & 10 | Pre & Post | 2016-07-25 11:57:23 | 2016-07-25 11:58:37 | yast snapper | |
| 12 & 13 | Pre & Post | 2016-07-25 11:58:54 | 2016-07-25 11:58:57 | zypp(y2base) | important=no |
| 11 & 14 | Pre & Post | 2016-07-25 11:58:47 | 2016-07-25 11:58:59 | yast online_update | |
| 15 | Pre | 2016-07-25 11:59:48 | | yast snapper | |

- Click *Show Changes* to open the list of files that differ between the two snapshots.

Selected Snapshot Overview

/ yast online_update

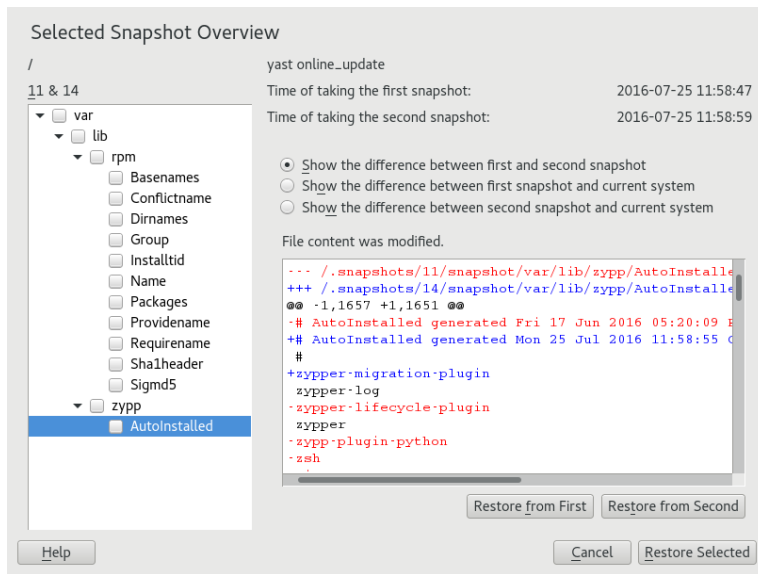
11 & 14

Time of taking the first snapshot: 2016-07-25 11:58:47

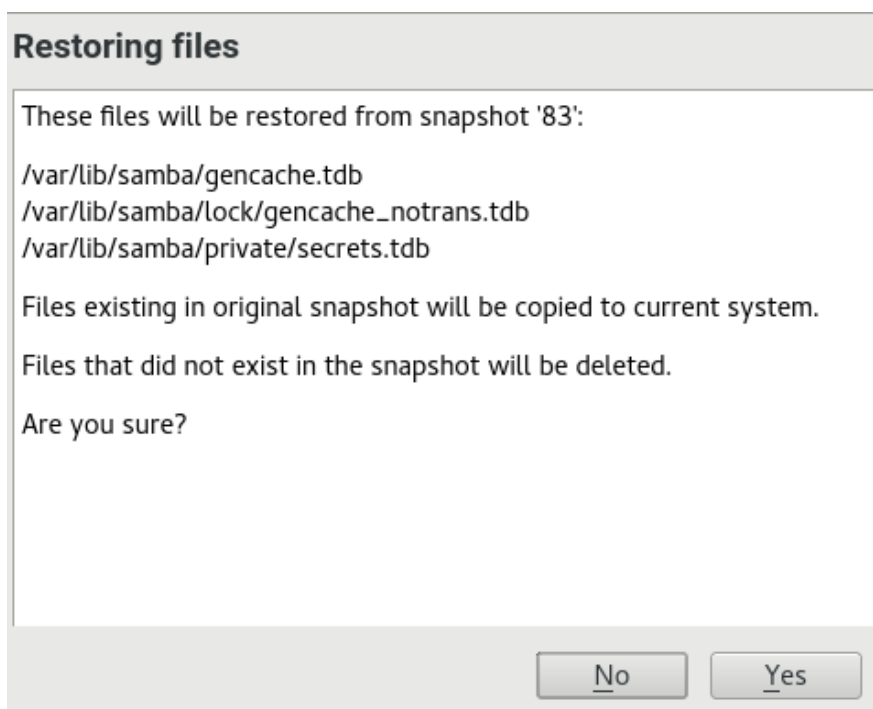
Time of taking the second snapshot: 2016-07-25 11:58:59

- var
 - lib
 - rpm
 - Basenames
 - Conflictname
 - Dirnames
 - Group
 - Installtid
 - Name
 - Packages
 - Providename
 - Requirename
 - Sha1header
 - Sigmd5
 - zypp
 - AutoInstalled

- Review the list of files. To display a “diff” between the pre- and post-version of a file, select it from the list.



- 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*.

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)`.

```
tux > 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 `-`.

```
tux > 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/Shalheader
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.

```
tux > 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-c-0-iso10646-1
 ds=y:ai=0.2:luximr.ttf -b&h-luxi mono-bold-i-normal--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.

```

tux > 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/Shalheader
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.

3.2.2 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 3.5, "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 3.3, "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 3.3: 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 3.4: RESTORING FILES USING THE `snapper` COMMAND

1. Get a list of timeline snapshots for a specific configuration by running the following command:

```
tux > 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:

```
tux > sudo snapper -c CONFIG status SNAPSHOT_ID..0
```

Replace `SNAPSHOT_ID` by the ID for the snapshot from which you want to restore the file(s).

3. Optionally list the differences between the current file version and the one from the snapshot by running

```
tux > 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

```
tux > sudo snapper -c CONFIG -v undochange SNAPSHOT_ID..0 FILENAME1 FILENAME2
```

If you do not specify file names, all changed files will be restored.

3.3 System Rollback by Booting from Snapshots

The GRUB 2 version included on openSUSE Leap 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 openSUSE Leap 15.2 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 3.2, "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, a single partition and a single subvolume. Directories that are excluded from snapshots such as `/srv` (see [Section 3.1.3, “Directories That Are Excluded from Snapshots”](#) for a full list) may reside on separate partitions.
- The system needs to be bootable via the installed boot loader.

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

```
tux > 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 3.3.1, “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).

3.3.1 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 3.7, "Automatic Snapshot Clean-Up"](#) for details. If the snapshot contains important data, extract the data from the snapshot before it is removed.

3.3.1.1 Example of Rollback Snapshot

For example, after a fresh installation the following snapshots are available on the system:

```
root # 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.

```
root # 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 |   |         |                       |
```


3.3.2 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 3.1: BOOT LOADER: SNAPSHOTS

Each snapshot entry in the boot loader follows a naming scheme which makes it possible to identify it easily:

```
[*] ① OS ② (KERNEL ③ ,DATE ④ TIME ⑤ ,DESCRIPTION ⑥ )
```

- ① If the snapshot was marked `important`, the entry is marked with a `*`.
- ② Operating system label.
- ③ Kernel 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:

```
tux > 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.

3.3.3 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.

3.3.3.1 Directories Excluded from Snapshots

Root file system snapshots do not contain all directories. See [Section 3.1.3, “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 others 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.

3.3.3.2 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`.

3.4 Enabling Snapper in User Home Directories

You may enable snapshots for users' `/home` directories, which supports a number of 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 3.4.3, “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 some users remain logged in for extended periods of time. You may change the defaults using the normal Snapper commands and configuration files.

3.4.1 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`:

```
root # 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:

```
root # /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:

```
root # 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.

3.4.2 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.

```
root # /usr/lib/pam_snapper/pam_snapper_userdel.sh username
```

3.4.3 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.

```
root # btrfs subvol create /home/username
root # snapper -c home_username create-config /home/username
root # sed -i -e "s/ALLOW_USERS=\"\"/ALLOW_USERS=\"username\"/g" \
/etc/snapper/configs/home_username
root # yast users add username=username home=/home/username password=password
root # chown username.group /home/username
root # chmod 755 /home/username/.snapshots
```

3.5 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 3.5.1.1, "Configuration Data"](#) for a list of the default values.



Note: Minimum Root File System Size for Enabling Snapshots

As explained in [Section 3.1, "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: $\underline{ROOT_BASE_SIZE} * (1 + \underline{BTRFS_INCREASE_PERCENTAGE}/100)$

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:

```
tux > sudo snapper -c www-data ❶ create-config /srv/www ❷
```

❶ Name of configuration file.

❷ 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 3.5.1, “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:

```
tux > sudo snapper -c www-data create-config -t MY_DEFAULTS /srv/www
```

3.5.1 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:

```
tux > 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 3.5.1.1, "Configuration Data"](#).

To display the default configuration, run:

```
tux > 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 3.5.1.1, "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**.

3.5.1.1 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 3.5.1.2, "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 3.7.3, "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 admin snapshots. See [Section 3.7.1, "Cleaning Up Numbered Snapshots"](#) for details.

QGROUP / SPACE_LIMIT

Adds quota support to the clean-up algorithms. See [Section 3.7.5, "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 3.5.1.2, "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 3.7.2, “Cleaning Up Timeline Snapshots”](#) for details.

3.5.1.2 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 3.5: ENABLING REGULAR USERS TO USE SNAPPER

Note that 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 3.5, “Creating and Modifying Snapper Configurations”](#) for instructions. Example:

```
tux > 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 3.5.1, “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:

```
tux > 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 user(s) and/or group(s). You can test it with the `list` command, for example:

```
www_admin:~ > snapper -c web_data list
```

3.6 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 3.5, “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`).

3.6.1 Snapshot Metadata

Each snapshot consists of the snapshot itself and some 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 3.6.1.1, “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 3.7, “Automatic Snapshot Clean-Up”](#) for details.

3.6.1.1 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.

3.6.1.2 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 3.5.1, “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.

3.6.2 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.

3.6.3 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.

3.6.4 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 3.6.1.2, “Cleanup Algorithms”](#) for details.

3.7 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.

In addition to that, 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.

3.7.1 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:

```
tux > 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 filesystem space into account. The algorithms also delete snapshots above the minimum value until the limits for the snapshot and filesystem are reached.

Example command to change or set:

```
tux > sudo snapper -c CONFIG set-config "NUMBER_LIMIT=10"
```

Important: Ranged Compared to Constant Values

In case quota support is enabled (see [Section 3.7.5, "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 will fail 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:

```
tux > 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
```

On the other hand, 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
```

3.7.2 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:

```
tux > 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 3.1: 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.

3.7.3 Cleaning Up Snapshot Pairs That Do Not Differ

As explained in *Section 3.1.2, "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".

3.7.4 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.

3.7.5 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 3.5, "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.

3.8 Showing Exclusive Disk Space Used by Snapshots

Snapshots share data, for efficient use of storage space, so using ordinary commands like **du** and **df** won't 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:

```
root # 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:

```
root # 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:

```
root # 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 3.6.4, “Deleting Snapshots”](#) for details.

3.9 Frequently Asked Questions

Q: Why does Snapper never show changes in `/var/log`, `/tmp` and other directories?

A: For some directories we decided to exclude them from snapshots. See [Section 3.1.3, “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 3.3, “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 3.6.2, “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:

```
tux > sudo snapper list -a
```

2. Memorize the number of the snapshot(s) you want to prevent from being deleted.
3. Run the following command and replace the number placeholders with the number(s) you memorized:

```
tux > 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/>.

4 Remote Graphical Sessions with VNC

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.

openSUSE Leap supports two different kinds of VNC sessions: One-time sessions that “live” as long as 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.

4.1 The `vncviewer` Client



Important: Supported Display Managers

A machine can reliably accept VNC connections only if it uses a display manager that supports the XDMCP protocol. While `gdm`, `lxdm`, or `lightdm` support XDMCP, the KDE 5 default display manager `sddm` does not support it. When changing the default display manager, remember to log out of the current X session and restart the display manager with

```
tux > sudo systemctl restart xdm.service
```

To connect to a VNC service provided by a server, a client is needed. The default in openSUSE Leap is `vncviewer`, provided by the `tigervnc` package.

4.1.1 Connecting Using the `vncviewer` CLI

To start your VNC viewer and initiate a session with the server, use the command:

```
tux > vncviewer jupiter.example.com:1
```


Instead of the VNC display number you can also specify the port number with two colons:

```
tux > 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 picked by the `vncserver` command on the target machine. See [Section 4.4, “Configuring Persistent VNC Server Sessions”](#) for further info.

4.1.2 Connecting Using the vncviewer GUI

By running `vncviewer` without specifying `--listen` or a host to connect to, it will show a window to ask for connection details. Enter the host into the *VNC server* field like in [Section 4.1.1, “Connecting Using the vncviewer CLI”](#) and click *Connect*.

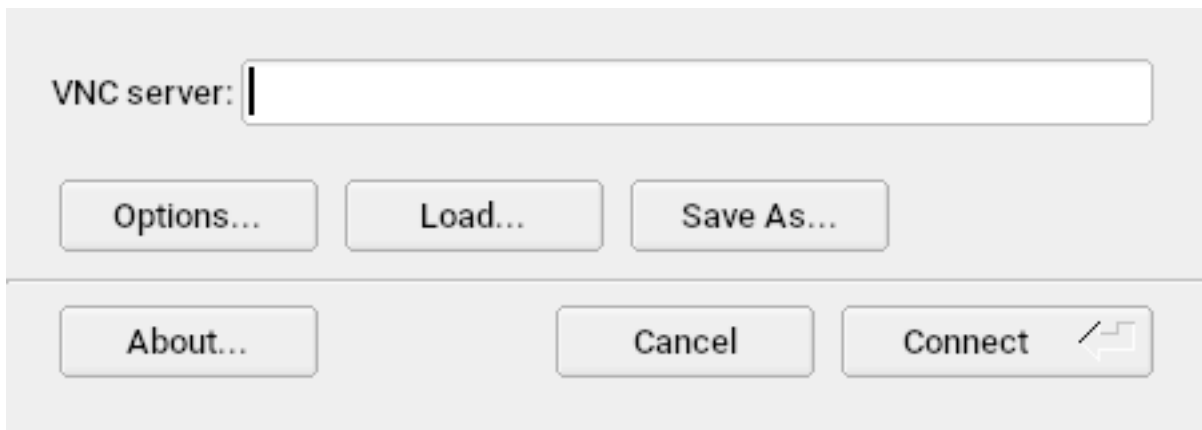


FIGURE 4.1: VNCVIEWER

4.1.3 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.

4.2 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.

4.2.1 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:

```
root # zypper in remmina remmina-plugin-vnc
```

4.2.2 Main Window

Run Remmina by entering the `remmina` command.

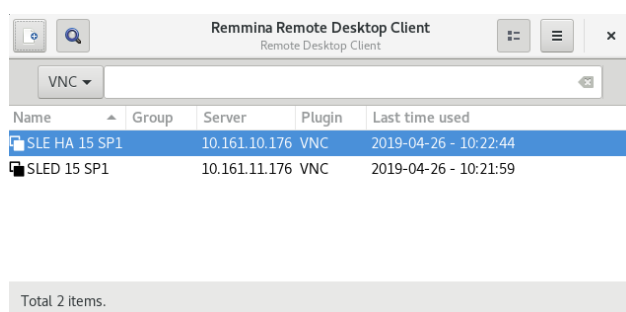


FIGURE 4.2: REMMINA'S MAIN WINDOW

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.

4.2.3 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 4.3: 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 will be listed in the main window.

4.2.4 Starting Remote Sessions

You can either start a previously saved session, or quick-start a remote session without saving the connection details.

4.2.4.1 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 4.4: QUICK-STARTING

Select the communication protocol from the drop-down box, for example 'VNC', then enter the VNC server DNS or IP address followed by a colon and a display number, and confirm with .

4.2.4.2 Opening Saved Remote Sessions

To open a specific remote session, double-click it from the list of sessions.

4.2.4.3 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, such as toggle fullscreen 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.

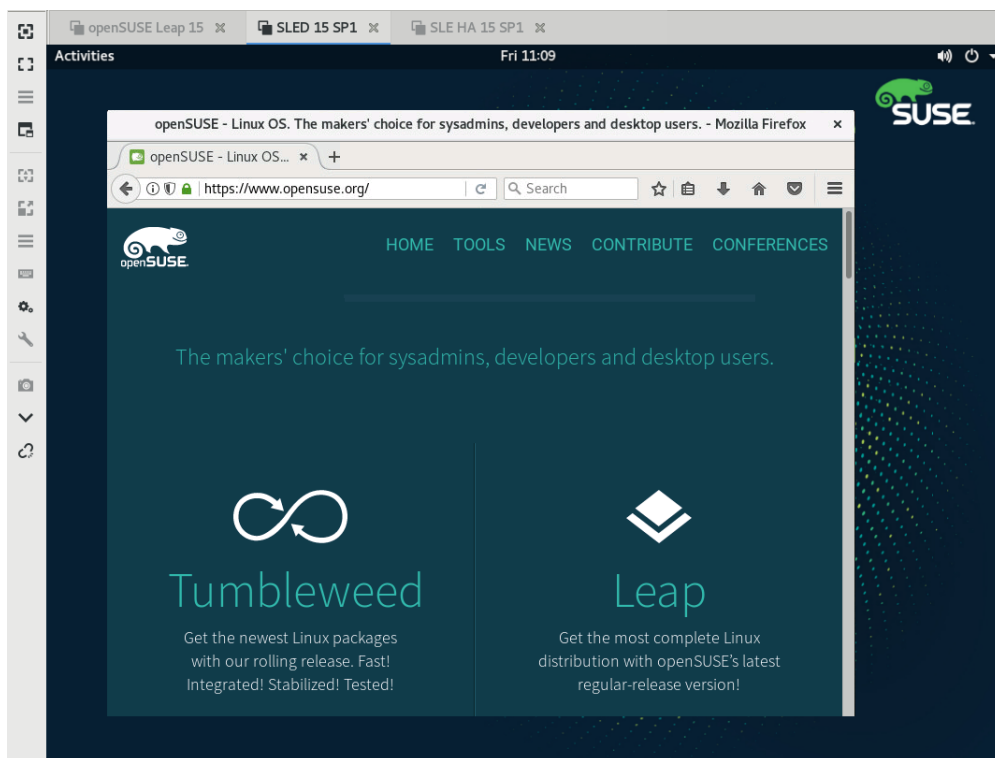


FIGURE 4.5: REMMINA VIEWING REMOTE SESSION

4.2.5 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 4.2.3, "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.

4.2.6 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:

```
tux > 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.

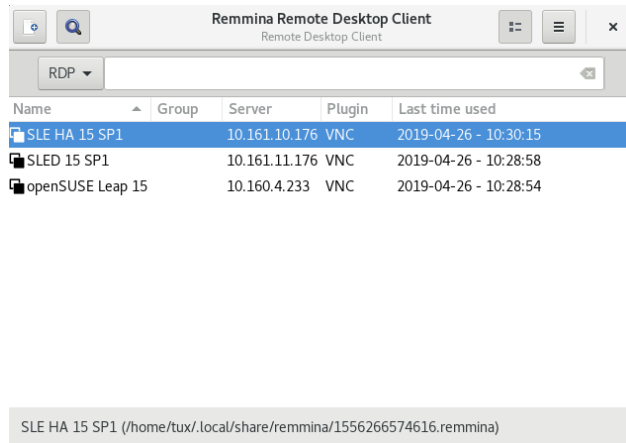


FIGURE 4.6: READING PATH TO THE PROFILE FILE

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.

4.3 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 terminate the client connection to such a VNC session, all applications started within that session will be 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 4.1: 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.

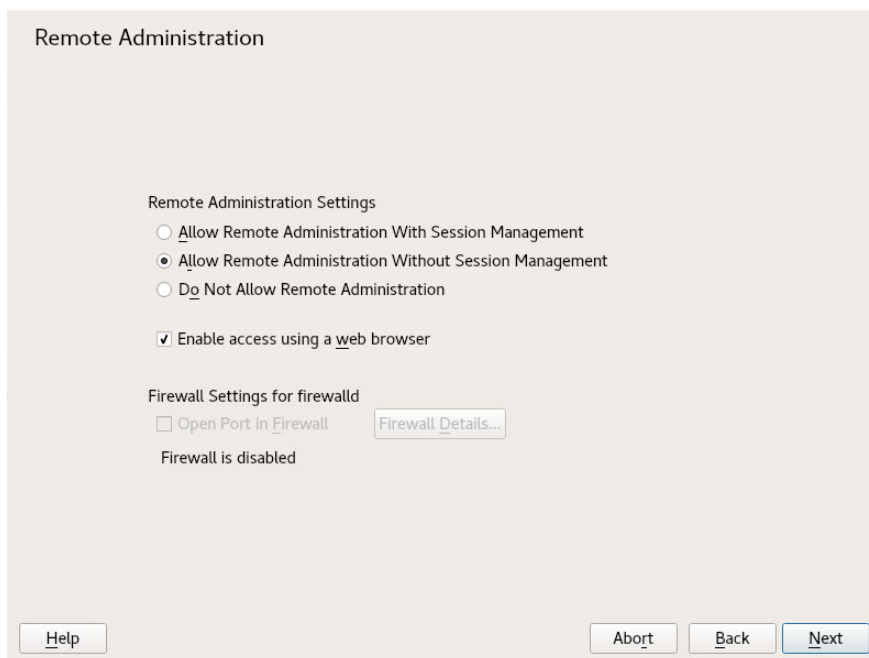


FIGURE 4.7: REMOTE ADMINISTRATION

4.3.1 Available Configurations

The default configuration on openSUSE Leap 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, see [Section 4.3.3, “Configuring One-time VNC Sessions”](#).

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.

4.3.2 Initiating a One-time VNC Session

To connect to a one-time VNC session, a VNC viewer must be installed, see also [Section 4.1, “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>

4.3.3 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:

```
tux > sudo systemctl enable xvnc.socket
```

To start the socket immediately, run:

```
tux > 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:

```
tux > sudo systemctl reload xvnc.socket
```

Important: Firewall and VNC Ports

When activating Remote Administration as described in *Procedure 4.1, “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 25 “Masquerading and Firewalls”* for instructions.

4.4 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 might 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 4.1, “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>

There are two types of persistent VNC sessions:

- *VNC Session Initiated Using vncserver*
- *VNC Session Initiated Using vncmanager*

4.4.1 VNC Session Initiated Using `vncserver`

This type of persistent VNC session is initiated on the server. The session and all applications started in this session run regardless of client connections until the session is terminated. Access to persistent sessions is protected by two possible types of passwords:

- a regular password that grants full access or
- an optional view-only password that grants a non-interactive (view-only) access.

A session can have multiple client connections of both kinds at once.

PROCEDURE 4.2: STARTING A PERSISTENT VNC SESSION USING `vncserver`

1. Open a shell and make sure you are logged in as the user that should own the VNC session.
2. If the network interface serving the VNC sessions is protected by a firewall, you need to manually open the port used by your session in the firewall. If starting multiple sessions you may alternatively open a range of ports. See *Book "Security and Hardening Guide", Chapter 25 "Masquerading and Firewalls"* for details on how to configure the firewall.
`vncserver` uses the ports `5901` for display `:1`, `5902` for display `:2`, and so on. For persistent sessions, the VNC display and the X display usually have the same number.
3. To start a session with a resolution of 1024x768 pixel and with a color depth of 16-bit, enter the following command:

```
vncserver -alwaysshared -geometry 1024x768 -depth 16
```

The `vncserver` command picks an unused display number when none is given and prints its choice. See `man 1 vncserver` for more options.

When running `vncserver` for the first time, it asks for a password for full access to the session. If needed, you can also provide a password for view-only access to the session.

The password(s) you are providing here are also used for future sessions started by the same user. They can be changed with the `vncpasswd` command.

Important: Security Considerations

Make sure to use strong passwords of significant length (eight or more characters). Do not share these passwords.

To terminate the session shut down the desktop environment that runs inside the VNC session from the VNC viewer as you would shut it down if it was a regular local X session.

If you prefer to manually terminate a session, open a shell on the VNC server and make sure you are logged in as the user that owns the VNC session you want to terminate. Run the following command to terminate the session that runs on display `:1`: `vncserver -kill :1`

4.4.1.1 Configuring Persistent VNC Sessions

Persistent VNC sessions can be configured by editing `$HOME/.vnc/xstartup`. By default this shell script starts the same GUI/window manager it was started from. In openSUSE Leap this will either be GNOME or IceWM. If you want to start your session with a window manager of your choice, set the variable `WINDOWMANAGER`:

```
WINDOWMANAGER=gnome vncserver -geometry 1024x768
WINDOWMANAGER=icewm vncserver -geometry 1024x768
```



Note: One Configuration for Each User

Persistent VNC sessions are configured in a single per-user configuration. Multiple sessions started by the same user will all use the same start-up and password files.

4.4.2 VNC Session Initiated Using vncmanager

PROCEDURE 4.3: 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.

4.4.2.1 Configuring Persistent VNC Sessions

After you enable the VNC session management as described in *Procedure 4.3, "Enabling Persistent VNC Sessions"*, you can normally connect to the remote session with your favorite VNC viewer, such as **vncviewer** or Remmina. You will be presented with the login screen. After you log in, the 'VNC' icon will appear in the system tray of your desktop environment. Click the icon to open the *VNC Session* window. If it does not appear or if your desktop environment does not support icons in the system tray, run **vncmanager-controller** manually.

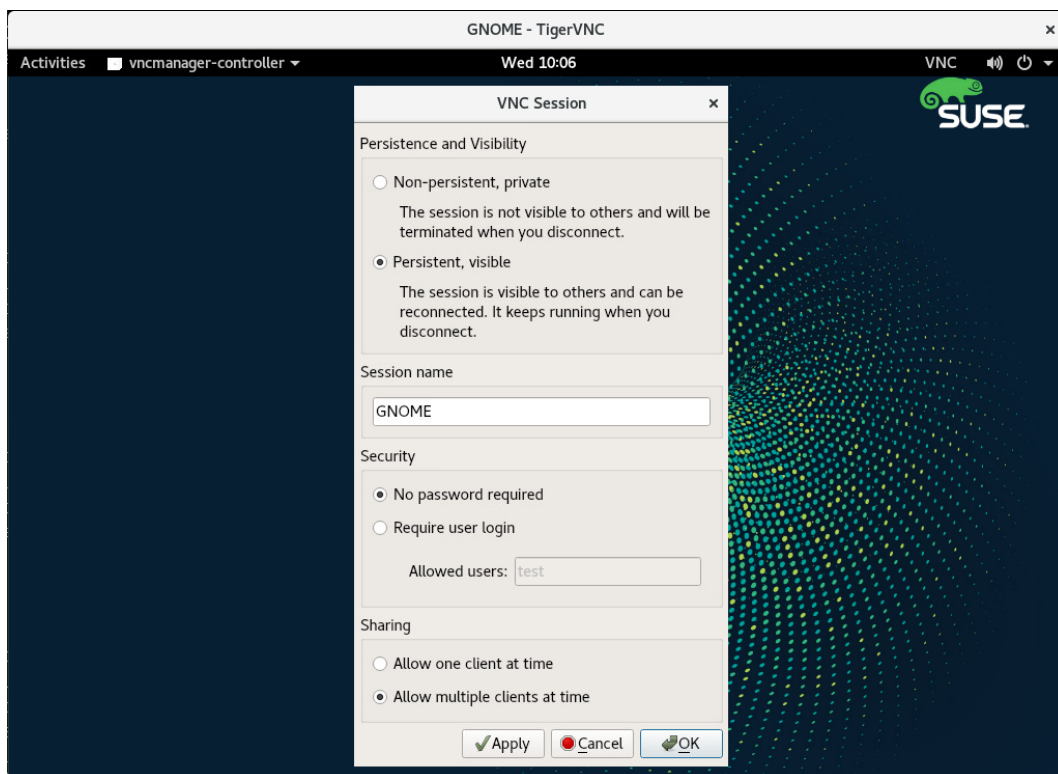


FIGURE 4.8: VNC SESSION SETTINGS

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 will be terminated after you disconnect from it. Refer to [Section 4.3, "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

Here you can specify the name of the persistent session so that it is easily identified when reconnecting.

No password required

The session will be 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*.

4.4.2.2 Joining Persistent VNC Sessions

After you set up a persistent VNC session as described in *Section 4.4.2.1, “Configuring Persistent VNC Sessions”*, you can join it with your VNC viewer. After your VNC client connects to the server, you will be prompted to choose whether you want to create a new session, or join the existing one:

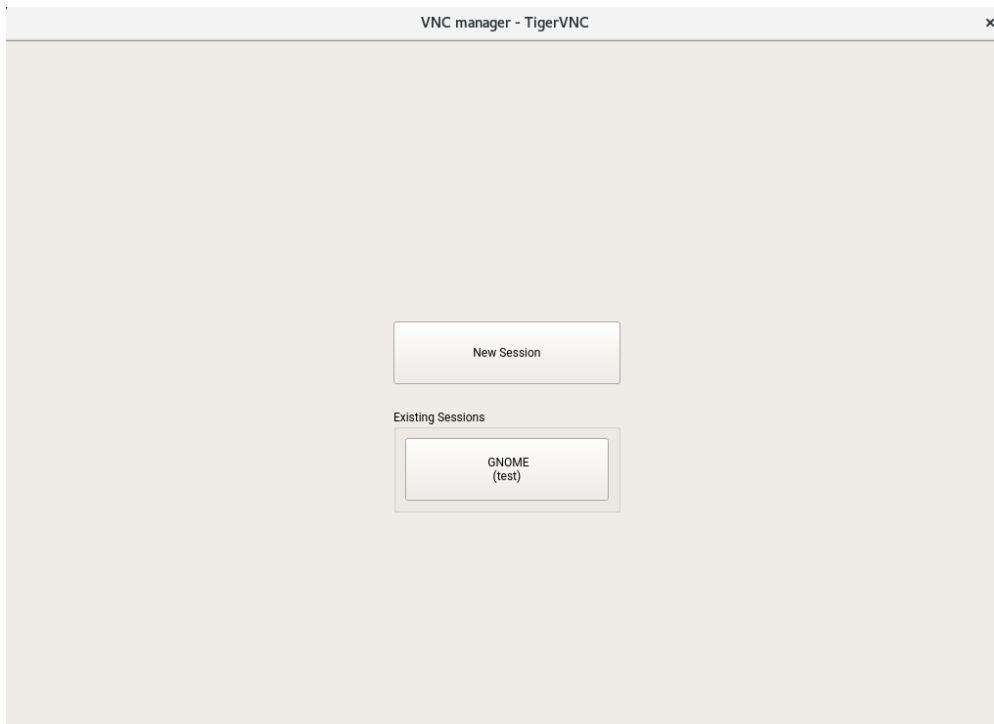


FIGURE 4.9: JOINING A PERSISTENT VNC SESSION

After you click the name of the existing session, you may be asked for login credentials, depending on the persistent session settings.

4.5 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: 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 will be 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 will warn you with the "Connection not encrypted!" message in that case).

5 Expert Partitioner

Sophisticated system configurations require specific disk setups. All common partitioning tasks can be done during the installation. To get persistent device naming with block devices, use the block devices below [/dev/disk/by-id](#) or [/dev/disk/by-uuid](#). Logical Volume Management (LVM) is a disk partitioning scheme that is designed to be much more flexible than the physical partitioning used in standard setups. Its snapshot functionality enables easy creation of data backups. Redundant Array of Independent Disks (RAID) offers increased data integrity, performance, and fault tolerance. openSUSE Leap also supports multipath I/O . There is also the option to use iSCSI as a networked disk.



Warning: Disk Space Units

Note that for partitioning purposes, disk space is measured in binary units, rather than in decimal units. For example, if you enter sizes of 1GB, 1GiB or 1G, they all signify 1 GiB (Gibibyte), as opposed to 1 GB (Gigabyte).

Binary

1 GiB = 1 073 741 824 bytes.

Decimal

1 GB = 1 000 000 000 bytes.

Difference

1 GiB \approx 1.07 GB.

5.1 Using the Expert Partitioner

With the Expert Partitioner, shown in *Figure 5.1, "The YaST Partitioner"*, manually modify the partitioning of one or several hard disks. You can add, delete, resize, and edit partitions, or access the soft RAID, and LVM configuration.



Warning: Repartitioning the Running System

Although it is possible to repartition your system while it is running, the risk of making a mistake that causes data loss is very high. Try to avoid repartitioning your installed system and always create a complete backup of your data before attempting to do so.

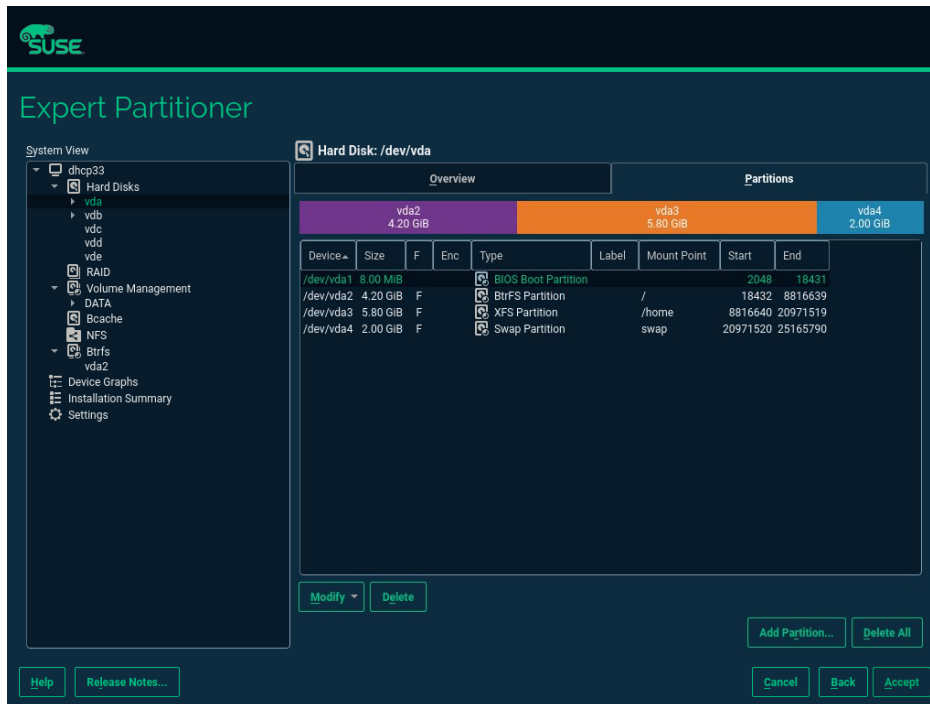


FIGURE 5.1: THE YAST PARTITIONER

All existing or suggested partitions on all connected hard disks are displayed in the list of *Available Storage* in the YaST *Expert Partitioner* dialog. Entire hard disks are listed as devices without numbers, such as `/dev/sda`. Partitions are listed as parts of these devices, such as `/dev/sda1`. The size, type, encryption status, file system, and mount point of the hard disks and their partitions are also displayed. The mount point describes where the partition appears in the Linux file system tree.

Several functional views are available on the left hand *System View*. These views can be used to collect information about existing storage configurations, configure functions (like RAID, Volume Management, Crypt Files), and view file systems with additional features, such as Btrfs, NFS, or TMPFS.

If you run the expert dialog during installation, any free hard disk space is also listed and automatically selected. To provide more disk space to openSUSE Leap, free the needed space by going from the bottom toward the top in the list of partitions.

5.1.1 Partition Tables

openSUSE Leap allows to use and create different *partition tables*. In some cases the partition table is called *disk label*. The partition table is important to the boot process of your computer. To boot your machine from a partition in a newly created partition table, make sure that the table format is supported by the firmware.

To change the partition table, click the relevant disk name in the *System View* and choose *Expert > Create New Partition Table*.

5.1.1.1 Master Boot Record

The *master boot record (MBR)* is the legacy partition table used on IBM PCs. It is sometimes also called an *MS-DOS* partition table. The MBR only supports four primary partitions. If the disk already has an MBR, openSUSE Leap allows you to create additional partitions in it which can be used as the installation target.

The limit of four partitions can be overcome by creating an *extended partition*. The extended partition itself is a primary partition and can contain more *logical partitions*.

UEFI firmware usually supports booting from MBR in the legacy mode.

5.1.1.2 GPT Partition Table

UEFI computers use a *GUID Partition Table (GPT)* by default. openSUSE Leap will create a GPT on a disk if no other partition table exists.

Old BIOS firmware does not support booting from GPT partitions.

You need a GPT partition table to use one of the following features:

- More than four primary partitions
- UEFI Secure Boot
- Use disks larger than 2 TB



Note: Partitions Created with Parted 3.1 or Earlier Mislabeled

GPT partitions created with Parted 3.1 or earlier used the Microsoft Basic Data partition type instead of the newer Linux-specific GPT GUID. Newer versions of Parted will set the misleading flag `msftdata` on such partitions. This will also lead to various disk tools labeling the partition as a *Windows Data Partition* or similar.

To remove the flag, run:

```
root # parted DEVICE set PARTITION_NUMBER msftdata off
```

5.1.2 Partitions

The YaST Partitioner can create and format partitions with several file systems. The default file system used by openSUSE Leap is Btrfs. For details, see [Section 5.1.2.2, “Btrfs Partitioning”](#).

Other commonly used file systems are available: Ext2, Ext3, Ext4, FAT, XFS, Swap, and UDF.

5.1.2.1 Creating a Partition

To create a partition select *Hard Disks* and then a hard disk with free space. The actual modification can be done in the *Partitions* tab:

1. Click *Add* to create a new partition. When using *MBR*, specify to create a primary or extended partition. Within the extended partition, you can create several logical partitions. For details, see [Section 5.1.1, “Partition Tables”](#).
2. Specify the size of the new partition. You can either choose to occupy all the free unpartitioned space, or enter a custom size.
3. Select the file system to use and a mount point. YaST suggests a mount point for each partition created. To use a different mount method, like mount by label, select *Fstab Options*.
4. Specify additional file system options if your setup requires them. This is necessary, for example, if you need persistent device names. For details on the available options, refer to [Section 5.1.3, “Editing a Partition”](#).
5. Click *Finish* to apply your partitioning setup and leave the partitioning module. If you created the partition during installation, you are returned to the installation overview screen.

5.1.2.2 Btrfs Partitioning

The default file system for the root partition is Btrfs. For details, see *Chapter 3, System Recovery and Snapshot Management with Snapper*. The root file system is the default subvolume and it is not listed in the list of created subvolumes. As a default Btrfs subvolume, it can be mounted as a normal file system.

Important: Btrfs on an Encrypted Root Partition

The default partitioning setup suggests the root partition as Btrfs with `/boot` being a directory. To encrypt the root partition, make sure to use the GPT partition table type instead of the default MSDOS type. Otherwise the GRUB2 boot loader may not have enough space for the second stage loader.

It is possible to create snapshots of Btrfs subvolumes—either manually, or automatically based on system events. For example when making changes to the file system, `zypper` invokes the `snapper` command to create snapshots before and after the change. This is useful if you are not satisfied with the change `zypper` made and want to restore the previous state. As `snapper` invoked by `zypper` creates snapshots of the *root* file system by default, it makes sense to exclude specific directories from snapshots. This is the reason YaST suggests creating the following separate subvolumes:

`/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.

Tip: Size of Btrfs Partition

Since saved snapshots require more disk space, it is recommended to reserve enough space for Btrfs. While the minimum size for a root Btrfs partition with snapshots and default subvolumes is 16 GB, SUSE recommends at least 32 GB, or more if /home does not reside on a separate partition.

5.1.2.3 Managing Btrfs Subvolumes using YaST

Subvolumes of a Btrfs partition can be now managed with the YaST *Expert partitioner* module. You can add new or remove existing subvolumes.

PROCEDURE 5.1: BTRFS SUBVOLUMES WITH YAST

1. Start the YaST *Expert Partitioner* with *System > Partitioner*.
2. Choose *Btrfs* in the left *System View* pane.
3. Select the Btrfs partition whose subvolumes you need to manage and click *Edit*.
4. Click *Subvolume Handling*. You can see a list of all existing subvolumes of the selected Btrfs partition. There are several @/.snapshots/xyz/snapshot entries—each of these subvolumes belongs to one existing snapshot.
5. Depending on whether you want to add or remove subvolumes, do the following:
 - a. To remove a subvolume, select it from the list of *Existing Subvolumes* and click *Remove*.

- b. To add a new subvolume, enter its name to the *New Subvolume* text box and click *Add new*.

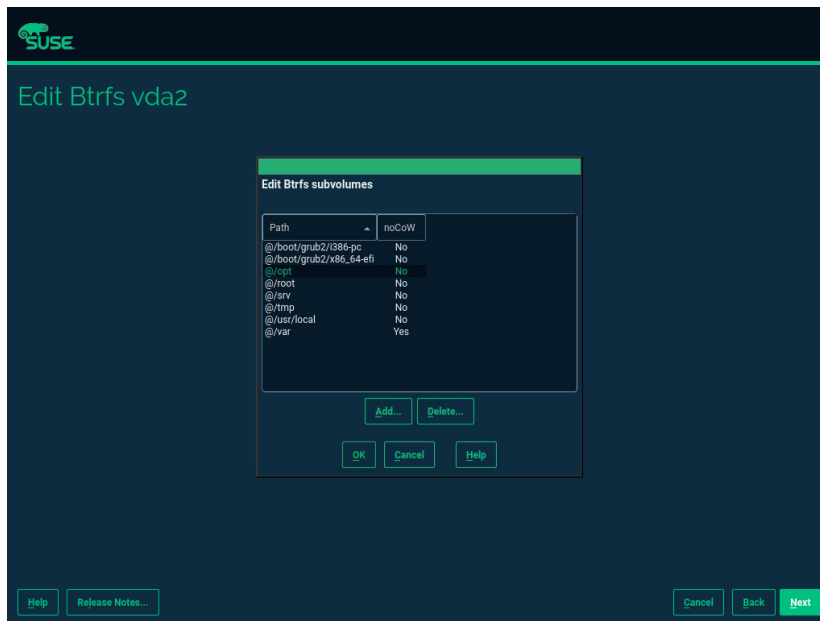


FIGURE 5.2: BTRFS SUBVOLUMES IN YAST PARTITIONER

6. Confirm with *OK* and *Finish*.
7. Leave the partitioner with *Finish*.

5.1.3 Editing a Partition

When you create a new partition or modify an existing partition, you can set various parameters. For new partitions, the default parameters set by YaST are usually sufficient and do not require any modification. To edit your partition setup manually, proceed as follows:

1. Select the partition.
2. Click *Edit* to edit the partition and set the parameters:

File System ID

Even if you do not want to format the partition at this stage, assign it a file system ID to ensure that the partition is registered correctly. Typical values are *Linux*, *Linux swap*, *Linux LVM*, and *Linux RAID*.

File System

To change the partition file system, click *Format Partition* and select file system type in the *File System* list.

openSUSE Leap supports several types of file systems. Btrfs is the Linux file system of choice for the root partition because of its advanced features. It supports copy-on-write functionality, creating snapshots, multi-device spanning, subvolumes, and other useful techniques. XFS, Ext3, and Ext4 are journaling file systems. These file systems can restore the system very quickly after a system crash, using write processes logged during the operation. Ext2 is not a journaling file system, but it is adequate for smaller partitions because it does not require much disk space for management.

The default file system for the root partition is Btrfs. The default file system for additional partitions is XFS.

The UDF file system can be used on optical rewritable and non-rewritable media, USB flash drives and hard disks. It is supported by multiple operating systems.

Swap is a special format that allows the partition to be used as a virtual memory. Create a swap partition of at least 256 MB. However, if you use up your swap space, consider adding memory to your system instead of adding swap space.



Warning: Changing the File System

Changing the file system and reformatting partitions irreversibly deletes all data from the partition.

For details on the various file systems, refer to *Storage Administration Guide*.

Encrypt Device

If you activate the encryption, all data is written to the hard disk in encrypted form. This increases the security of sensitive data, but reduces the system speed, as the encryption takes some time to process. More information about the encryption of file systems is provided in *Book "Security and Hardening Guide", Chapter 14 "Encrypting Partitions and Files"*.

Mount Point

Specify the directory where the partition should be mounted in the file system tree. Select from YaST suggestions or enter any other name.

Fstab Options

Specify various parameters contained in the global file system administration file (`/etc/fstab`). The default settings should suffice for most setups. You can, for example, change the file system identification from the device name to a volume label. In the volume label, use all characters except `/` and space.

To get persistent device names, use the mount option *Device ID*, *UUID* or *LABEL*. In openSUSE Leap, persistent device names are enabled by default.

If you prefer to mount the partition by its label, you need to define one in the *Volume label* text entry. For example, you could use the partition label `HOME` for a partition intended to mount to `/home`.

If you intend to use quotas on the file system, use the mount option *Enable Quota Support*. This must be done before you can define quotas for users in the YaST *User Management* module. For further information on how to configure user quota, refer to *Book "Start-Up", Chapter 5 "Managing Users with YaST", Section 5.3.3 "Managing Quotas"*.

3. Select *Finish* to save the changes.



Note: Resize File Systems

To resize an existing file system, select the partition and use *Resize*. Note, that it is not possible to resize partitions while mounted. To resize partitions, unmount the relevant partition before running the partitioner.

5.1.4 Expert Options

After you select a hard disk device (like *sda*) in the *System View* pane, you can access the *Expert* menu in the lower right part of the *Expert Partitioner* window. The menu contains the following commands:

Create New Partition Table

This option helps you create a new partition table on the selected device.



Warning: Creating a New Partition Table

Creating a new partition table on a device irreversibly removes all the partitions and their data from that device.

Clone This Disk

This option helps you clone the device partition layout (but not the data) to other available disk devices.

5.1.5 Advanced Options

After you select the host name of the computer (the top-level of the tree in the *System View* pane), you can access the *Configure* menu in the lower right part of the *Expert Partitioner* window. The menu contains the following commands:

Configure iSCSI

To access SCSI over IP block devices, you first need to configure iSCSI. This results in additionally available devices in the main partition list.

Configure Multipath

Selecting this option helps you configure the multipath enhancement to the supported mass storage devices.

5.1.6 More Partitioning Tips

The following section includes a few hints and tips on partitioning that should help you make the right decisions when setting up your system.

5.1.6.1 Cylinder Numbers

Note, that different partitioning tools may start counting the cylinders of a partition with 0 or with 1. When calculating the number of cylinders, you should always use the difference between the last and the first cylinder number and add one.

5.1.6.2 Using swap

Swap is used to extend the available physical memory. It is then possible to use more memory than physical RAM available. The memory management system of kernels before 2.4.10 needed swap as a safety measure. Then, if you did not have twice the size of your RAM in swap, the performance of the system suffered. These limitations no longer exist.

Linux uses a page called “Least Recently Used” (LRU) to select pages that might be moved from memory to disk. Therefore, running applications have more memory available and caching works more smoothly.

If an application tries to allocate the maximum allowed memory, problems with swap can arise. There are three major scenarios to look at:

System with no swap

The application gets the maximum allowed memory. All caches are freed, and thus all other running applications are slowed. After a few minutes, the kernel's out-of-memory kill mechanism activates and kills the process.

System with medium sized swap (128 MB–512 MB)

At first, the system slows like a system without swap. After all physical RAM has been allocated, swap space is used as well. At this point, the system becomes very slow and it becomes impossible to run commands from remote. Depending on the speed of the hard disks that run the swap space, the system stays in this condition for about 10 to 15 minutes until the out-of-memory kill mechanism resolves the issue. Note that you will need a certain amount of swap if the computer needs to perform a “suspend to disk”. In that case, the swap size should be large enough to contain the necessary data from memory (512 MB–1GB).

System with lots of swap (several GB)

It is better to not have an application that is out of control and swapping excessively in this case. If you use such application, the system will need many hours to recover. In the process, it is likely that other processes get timeouts and faults, leaving the system in an undefined state, even after terminating the faulty process. In this case, do a hard machine reboot and try to get it running again. Lots of swap is only useful if you have an application that relies on this feature. Such applications (like databases or graphics manipulation programs) often have an option to directly use hard disk space for their needs. It is advisable to use this option instead of using lots of swap space.

If your system is not out of control, but needs more swap after some time, it is possible to extend the swap space online. If you prepared a partition for swap space, add this partition with YaST. If you do not have a partition available, you can also use a swap file to extend the swap. Swap files are generally slower than partitions, but compared to physical RAM, both are extremely slow so the actual difference is negligible.

To add a swap file in the running system, proceed as follows:

1. Create an empty file in your system. For example, to add a swap file with 128 MB swap at `/var/lib/swap/swapfile`, use the commands:

```
tux > sudo mkdir -p /var/lib/swap
tux > sudo dd if=/dev/zero of=/var/lib/swap/swapfile bs=1M count=128
```

2. Initialize this swap file with the command

```
tux > sudo mkswap /var/lib/swap/swapfile
```



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.

3. Activate the swap with the command

```
tux > sudo swapon /var/lib/swap/swapfile
```

To disable this swap file, use the command

```
tux > sudo swapoff /var/lib/swap/swapfile
```

4. Check the current available swap spaces with the command

```
tux > cat /proc/swaps
```

Note that at this point, it is only temporary swap space. After the next reboot, it is no longer used.

5. To enable this swap file permanently, add the following line to `/etc/fstab`:

```
/var/lib/swap/swapfile swap swap defaults 0 0
```

5.1.7 Partitioning and LVM

From the *Expert partitioner*, access the LVM configuration by clicking the *Volume Management* item in the *System View* pane. However, if a working LVM configuration already exists on your system, it is automatically activated upon entering the initial LVM configuration of a session. In this case, all disks containing a partition (belonging to an activated volume group) cannot be repartitioned. The Linux kernel cannot reread the modified partition table of a hard disk when any partition on this disk is in use. If you already have a working LVM configuration on your system, physical repartitioning should not be necessary. Instead, change the configuration of the logical volumes.

At the beginning of the physical volumes (PVs), information about the volume is written to the partition. To reuse such a partition for other non-LVM purposes, it is advisable to delete the beginning of this volume. For example, in the VG `system` and PV `/dev/sda2`, do this with the command:

```
dd if=/dev/zero of=/dev/sda2 bs=512 count=1
```



Warning: File System for Booting

The file system used for booting (the root file system or `/boot`) must not be stored on an LVM logical volume. Instead, store it on a normal physical partition.

5.2 LVM Configuration

This section explains specific steps to take when configuring LVM.



Warning: Back up Your Data

Using LVM is sometimes associated with increased risk such as data loss. Risks also include application crashes, power failures, and faulty commands. Save your data before implementing LVM or reconfiguring volumes. Never work without a backup.

The YaST LVM configuration can be reached from the YaST Expert Partitioner (see [Section 5.1, “Using the Expert Partitioner”](#)) within the *Volume Management* item in the *System View* pane. The Expert Partitioner allows you to edit and delete existing partitions and create new ones that need to be used with LVM.

5.2.1 Create Physical Volume

The first task is to create physical volumes that provide space to a volume group:

1. Select a hard disk from *Hard Disks*.
2. Change to the *Partitions* tab.
3. Click *Add* and enter the desired size of the PV on this disk.
4. Use *Do not format partition* and change the *File System ID* to *0x8E Linux LVM*. Do not mount this partition.
5. Repeat this procedure until you have defined all the desired physical volumes on the available disks.

5.2.2 Creating Volume Groups

If no volume group exists on your system, you must add one (see [Figure 5.3, "Creating a Volume Group"](#)). It is possible to create additional groups by clicking *Volume Management* in the *System View* pane, and then on *Add Volume Group*. One single volume group is usually sufficient.

1. Enter a name for the VG, for example, system.
2. Select the desired *Physical Extend Size*. This value defines the size of a physical block in the volume group. All the disk space in a volume group is handled in blocks of this size.
3. Add the prepared PVs to the VG by selecting the device and clicking *Add*. Selecting several devices is possible by holding **Ctrl** while selecting the devices.
4. Select *Finish* to make the VG available to further configuration steps.

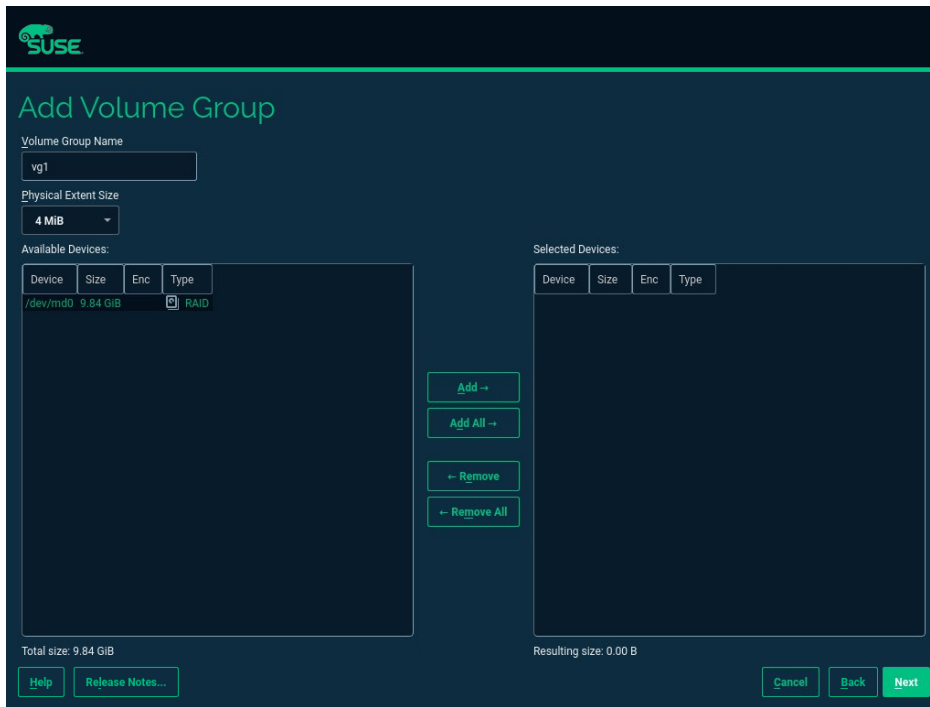


FIGURE 5.3: CREATING A VOLUME GROUP

If you have multiple volume groups defined and want to add or remove PVs, select the volume group in the *Volume Management* list and click *Resize*. In the following window, you can add or remove PVs to the selected volume group.

5.2.3 Configuring Logical Volumes

After the volume group has been filled with PVs, define the LVs which the operating system should use in the next dialog. Choose the current volume group and change to the *Logical Volumes* tab. *Add*, *Edit*, *Resize*, and *Delete* LVs as needed until all space in the volume group has been occupied. Assign at least one LV to each volume group.

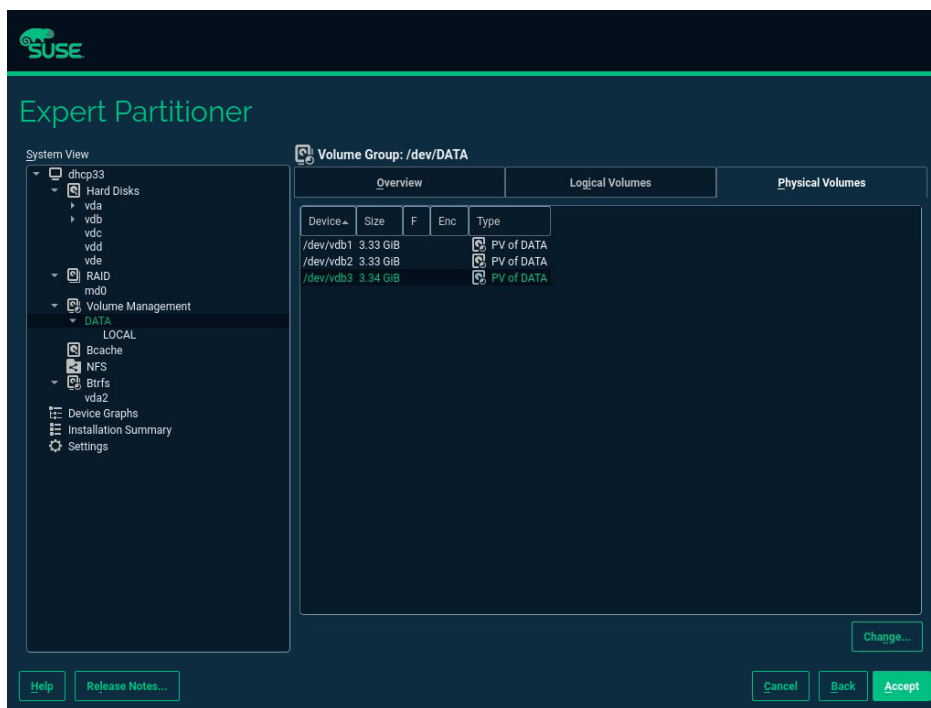


FIGURE 5.4: LOGICAL VOLUME MANAGEMENT

Click *Add* and go through the wizard-like pop-up that opens:

1. Enter the name of the LV. For a partition that should be mounted to `/home`, a name like `HOME` could be used.
2. Select the type of the LV. It can be either *Normal Volume*, *Thin Pool*, or *Thin Volume*. Note that you need to create a thin pool first, which can store individual thin volumes. The big advantage of thin provisioning is that the total sum of all thin volumes stored in a thin pool can exceed the size of the pool itself.
3. Select the size and the number of stripes of the LV. If you have only one PV, selecting more than one stripe is not useful.
4. Choose the file system to use on the LV and the mount point.

By using stripes it is possible to distribute the data stream in the LV among several PVs (striping). However, striping a volume can only be done over different PVs, each providing at least the amount of space of the volume. The maximum number of stripes equals to the number of PVs, where Stripe "1" means "no striping". Striping only makes sense with PVs on different hard disks, otherwise performance will decrease.



Warning: Striping

YaST cannot, at this point, verify the correctness of your entries concerning striping. Any mistake made here is apparent only later when the LVM is implemented on disk.

If you have already configured LVM on your system, the existing logical volumes can also be used. Before continuing, assign appropriate mount points to these LVs. With *Finish*, return to the YaST Expert Partitioner and finish your work there.

5.3 Soft RAID

This section describes actions required to create and configure various types of RAID. .

5.3.1 Soft RAID Configuration

The YaST *RAID* configuration can be reached from the YaST Expert Partitioner, described in [Section 5.1, “Using the Expert Partitioner”](#). This partitioning tool enables you to edit and delete existing partitions and create new ones to be used with soft RAID:

1. Select a hard disk from *Hard Disks*.
2. Change to the *Partitions* tab.
3. Click *Add* and enter the desired size of the raid partition on this disk.
4. Use *Do not Format the Partition* and change the *File System ID* to *OxFD Linux RAID*. Do not mount this partition.
5. Repeat this procedure until you have defined all the desired physical volumes on the available disks.

For RAID 0 and RAID 1, at least two partitions are needed—for RAID 1, usually exactly two and no more. If RAID 5 is used, at least three partitions are required, RAID 6 and RAID 10 require at least four partitions. It is recommended to use partitions of the same size only. The RAID partitions should be located on different hard disks to decrease the risk of losing data if one is defective (RAID 1 and 5) and to optimize the performance of RAID 0. After creating all the partitions to use with RAID, click *RAID > Add RAID* to start the RAID configuration.

In the next dialog, choose between RAID levels 0, 1, 5, 6 and 10. Then, select all partitions with either the “Linux RAID” or “Linux native” type that should be used by the RAID system. No swap or DOS partitions are shown.

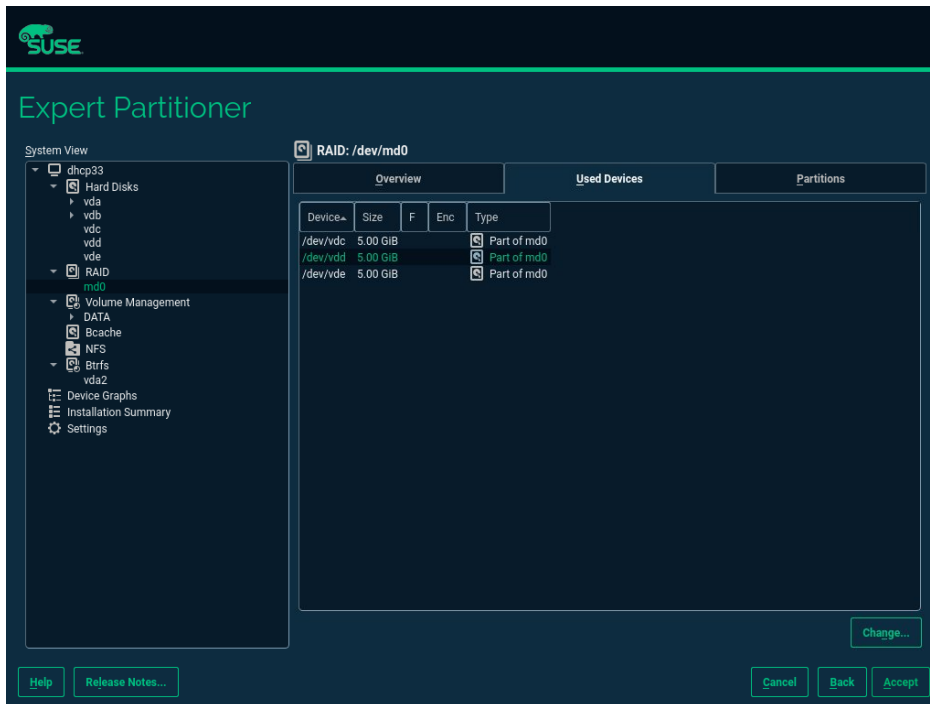


FIGURE 5.5: RAID PARTITIONS

To add a previously unassigned partition to the selected RAID volume, first click the partition then *Add*. Assign all partitions reserved for RAID. Otherwise, the space on the partition remains unused. After assigning all partitions, click *Next* to select the available *RAID Options*.

In this last step, set the file system to use, encryption and the mount point for the RAID volume. After completing the configuration with *Finish*, see the `/dev/md0` device and others indicated with *RAID* in the Expert Partitioner.

5.3.2 Troubleshooting

Check the file `/proc/mdstat` to find out whether a RAID partition has been damaged. If the system fails, shut down your Linux system and replace the defective hard disk with a new one partitioned the same way. Then restart your system and enter the command `mdadm /dev/mdX --add /dev/sdX`. Replace 'X' with your particular device identifiers. This integrates the hard disk automatically into the RAID system and fully reconstructs it.

Note that although you can access all data during the rebuild, you may encounter some performance issues until the RAID has been fully rebuilt.

5.3.3 For More Information

Configuration instructions and more details for soft RAID can be found at:

- <http://raid.wiki.kernel.org> ↗

Linux RAID mailing lists are available, such as <http://marc.info/?l=linux-raid> ↗.

6 Installing Multiple Kernel Versions

openSUSE Leap 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.



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 12.3, “Configuring the Boot Loader with YaST”](#) for more information.

6.1 Enabling and Configuring Multiversion Support

Installing multiple versions of a software package (multiversion support) is enabled by default from openSUSE Leap. 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 multiversion 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.

6.1.1 Automatically Deleting Unused Kernels

When frequently testing new kernels with multiversion support enabled, the boot menu quickly becomes confusing. Since a `/boot` partition usually has limited space you also might 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:

`5.3.18-8`: 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 openSUSE Leap)

`oldest+N`. keep the kernel with the Nth lowest version number

Here are some 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,5.3.18-8
```

Keep the latest kernel, the one currently running, and 5.3.18-8.



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 of 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.

6.1.2 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.

6.1.3 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`).

6.1.4 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,5.3.18-8,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 `5.3.18-8`).

6.2 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 Groups > Multiversion Packages*.

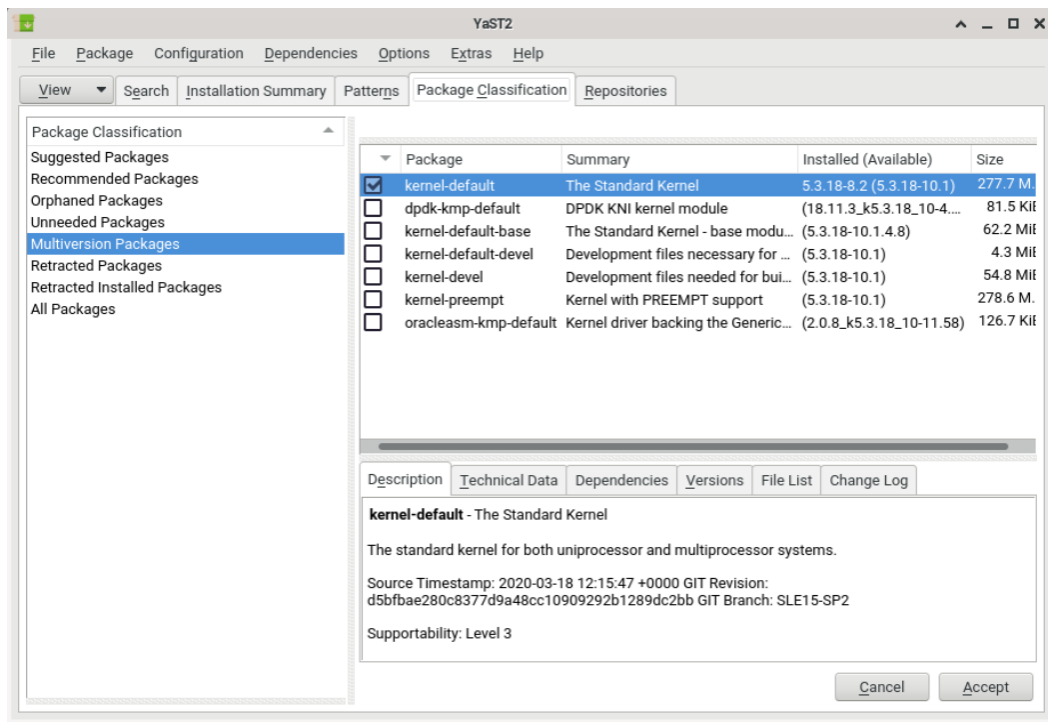


FIGURE 6.1: THE YAST SOFTWARE MANAGER: MULTIVERSION VIEW

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.

6.3 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 | 5.3.18-8.2 | x86_64 | (System Packages) |
| v | kernel-default | package | 5.3.18-10.1 | x86_64 | SLE-Module-Basesystem15-SP2-Pool |
| | kernel-default-base | package | 5.3.18-10.1.4.8 | x86_64 | SLE-Module-Basesystem15-SP2-Pool |
| | kernel-default-devel | package | 5.3.18-10.1 | x86_64 | SLE-Module-Basesystem15-SP2-Pool |
| | kernel-devel | package | 5.3.18-10.1 | noarch | SLE-Module-Basesystem15-SP2-Pool |
| i | kernel-firmware | package | 20200107-3.12.1 | noarch | SLE-Module-Basesystem15-SP2-Pool |

2. Specify the exact version when installing:

```
tux > sudo zypper in kernel-default-5.3.18-8.2
```

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.

6.4 Installing the Latest Kernel Version from the Repository `Kernel:HEAD`

1. Add the `Kernel:HEAD` repository with (the repository is added using the alias `kernel-repo`):

```
tux > sudo zypper ar \
http://download.opensuse.org/repositories/Kernel:/HEAD/standard/ \
kernel-repo
```

2. To refresh repositories, run:

```
tux > sudo zypper ref
```

3. To upgrade the kernel to the latest version in the `Kernel:HEAD` repository, run:

```
tux > sudo zypper dist-upgrade --allow-vendor-change --from kernel-repo
```

4. Reboot the machine.



Warning: Installing from Kernel:HEAD May Break the System

Installing a kernel from Kernel:HEAD should never be necessary, because important fixes are backported by SUSE and are made available as official updates. Installing the latest kernel only makes sense for kernel developers and kernel testers. If installing from Kernel:HEAD, be aware that it may break your system. Make sure to always have the original kernel available for booting as well.

7 Graphical User Interface

openSUSE Leap includes the X.org server, Wayland and the GNOME desktop. This chapter describes the configuration of the graphical user interface for all users.

7.1 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).

Usually, the X Window System needs no configuration. 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 openSUSE Leap 15.2 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 <http://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.

7.2 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 \(http://www.freetype.org/\)](http://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 \(http://www.fontconfig.org/\)](http://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 \(http://www.harfbuzz.org/\)](http://www.harfbuzz.org/) or [Pango \(http://www.pango.org/\)](http://www.pango.org/).

7.2.1 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:

```
tux > 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:

```
tux > 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.

7.2.2 Viewing Fonts

If you want to know what an installed font family looks like, either use the command **ftview** (package **ft2demos**) or visit <http://fontinfo.opensuse.org/>. For example, to display the FreeMono font in 14 point, use **ftview** like this:

```
tux > ftview 14 /usr/share/fonts/truetype/FreeMono.ttf
```

If you need further information, go to <http://fontinfo.opensuse.org/> to find out which styles (regular, bold, italic, etc.) and languages are supported.

7.2.3 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:

```
tux > 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:

```
tux > 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:

```
tux > 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.

7.2.4 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 7.2.5.2, “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](http://download.opensuse.org/repositories/M17N:/fonts/) repository. Add the repository to your list using the following command. For example, to add a repository for openSUSE Leap 15.2:

```
tux > sudo zypper ar
https://download.opensuse.org/repositories/M17N:/fonts/openSUSE_Leap_15.2/
```

To search for your `FONT_FAMILY_NAME` use this command:

```
tux > zypper se 'FONT_FAMILY_NAME*fonts'
```

7.2.5 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.

Important: Patent Issues with Subpixel Rendering

Subpixel rendering is not used in SUSE distributions. Although FreeType2 has support for this algorithm, it is covered by several patents expiring at the end of the year 2019. Therefore, setting subpixel rendering options in Fontconfig has no effect unless the system has a FreeType2 library with subpixel rendering compiled in.

Via Fontconfig, it is possible to select a rendering algorithms for every font individually or for a set of fonts.

7.2.5.1 Configuring Fonts via `sysconfig`

openSUSE Leap 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`:

```
tux > 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.

7.2.5.2 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 7.1, "Specifying Rendering Algorithms"](#)):

EXAMPLE 7.1: 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 7.2: 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 7.2, “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 7.1, “Generating PFL from Fontconfig rules”*.

TABLE 7.1: GENERATING PFL FROM FONTCONFIG RULES

| Order | Current PFL |
|---------------|---|
| Request | <u>Alegreya SC</u> |
| <i>Rule 1</i> | <u>Alegreya SC</u> , <u>serif</u> |
| <i>Rule 2</i> | <u>Alegreya SC</u> , <u>Droid Serif</u> , <u>serif</u> |
| <i>Rule 3</i> | <u>Alegreya SC</u> , <u>Droid Serif</u> , <u>serif</u> , <u>STIXGeneral</u> |

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 7.2, “Results from Generating PFL from Fontconfig Rules with Changed Order”*.

TABLE 7.2: RESULTS FROM GENERATING PFL FROM FONTCONFIG RULES WITH CHANGED ORDER

| Order | Current PFL | Note |
|---------|--------------------|-------------------------|
| Request | <u>Alegreya SC</u> | Same request performed. |

| Order | Current PFL | Note |
|---------------|-----------------------------------|--|
| <i>Rule 2</i> | <u>Alegreya SC</u> | <u>serif</u> not in PFL, nothing is substituted |
| <i>Rule 3</i> | <u>Alegreya SC</u> | <u>serif</u> not in PFL, nothing is substituted |
| <i>Rule 1</i> | <u>Alegreya SC</u> , <u>serif</u> | <u>Alegreya SC</u> 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 7.3, “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 7.3: 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 7.2, "Aliases and Family Name Substitutions" is expanded by Example 7.4, "Aliases and Family Names Substitutions".

EXAMPLE 7.4: 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 7.4, "Aliases and Family Names Substitutions"* would lead to the following PFL evolution:

TABLE 7.3: RESULTS FROM GENERATING PFL FROM FONTCONFIG RULES

| Order | Current PFL |
|---------|--------------------|
| Request | <u>Alegreya SC</u> |

| Order | Current PFL |
|---------------|---|
| <i>Rule 1</i> | <u>Alegreya SC</u> , <u>serif</u> |
| <i>Rule 2</i> | <u>Alegreya SC</u> , <u>Droid Serif</u> , <u>serif</u> |
| <i>Rule 3</i> | <u>Alegreya SC</u> , <u>Droid Serif</u> , <u>serif</u> , <u>STIXGeneral</u> |
| <i>Rule 4</i> | <u>Alegreya SC</u> , <u>Droid Serif</u> , <u>serif</u> , <u>Liberation Serif</u> , <u>STIXGeneral</u> |
| <i>Rule 5</i> | <u>Alegreya SC</u> , <u>Droid Serif</u> , <u>DejaVu Serif</u> , <u>serif</u> , <u>Liberation Serif</u> , <u>STIXGeneral</u> |



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.

7.3 GNOME Configuration for Administrators

7.3.1 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`.

7.3.2 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/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:

```
tux > sudo dconf update
```

3. Add the new `network` configuration database to the default user profile, by creating the file `/etc/dconf/profiles/user`. Then add the following content:

```
system-db:network
```


The file `/etc/dconf/profiles/user` is a GNOME default that will be used. 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>.

7.3.3 More Information

For more information, see <http://help.gnome.org/admin/>.

7.4 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 works only on systems running X11, not Wayland. If your system runs Wayland you may disable it and fall back to X11 (see [Section 7.4.1, "Prerequisites"](#)).

7.4.1 Prerequisites

You must have a configured and working NVIDIA Optimus GPU using the NVIDIA proprietary drivers from the openSUSE community repository (see [Section 7.4.3, “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 only works with X11. Use the `loginctl` command to see if your system is using X11 or Wayland:

```
tux > loginctl
  SESSION          UID USER          SEAT          TTY
      2            1000 tux            seat0
tux > 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.

7.4.2 Installing and Using SUSE Prime

Your NVIDIA graphics card should already be installed and working. If it is not, see [Section 7.4.3, “Installing NVIDIA Drivers”](#).

Install the `suse-prime` package:

```
tux > sudo zypper install suse-prime
```

To switch your GPU run one of the following commands, then log out and log back in:

```
tux > sudo prime-select intel
tux > sudo prime-select intel2
tux > sudo prime-select nvidia
```

Use the `intel` driver when it's the modesetting driver. `intel2` is for systems that use the `xf86-video-intel` driver. You can get this information by installing and running `inxi`:

```
tux > 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?

```
tux > sudo /usr/sbin/prime-select get-current
Driver configured: intel
```

7.4.3 Installing NVIDIA Drivers

If you need to identify your NVIDIA card so you know which driver to use, run the following command:

```
tux > /sbin/lspci | grep VGA
```

Follow these steps to install the drivers with Zypper. First install the community repository for your distribution. For openSUSE Tumbleweed:

```
tux > sudo zypper addrepo --refresh https://download.nvidia.com/opensuse/tumbleweed
nvidia
```

For openSUSE 15.2:

```
tux > sudo zypper addrepo --refresh https://download.nvidia.com/opensuse/leap/15.2 nvidia
```

List the available driver packages:

```
tux > sudo zypper se nvidia
```

Then install the drivers for your NVIDIA graphics card:

```
tux > sudo zypper se packagename
```

II System

- 8 32-Bit and 64-Bit Applications in a 64-Bit System Environment **160**
- 9 Introduction to the Boot Process **162**
- 10 The systemd Daemon **170**
- 11 **journalctl**: Query the systemd Journal **195**
- 12 The Boot Loader GRUB 2 **203**
- 13 Basic Networking **221**
- 14 UEFI (Unified Extensible Firmware Interface) **292**
- 15 Special System Features **301**
- 16 Dynamic Kernel Device Management with udev **313**

8 32-Bit and 64-Bit Applications in a 64-Bit System Environment

openSUSE® Leap is available for 64-bit platforms. The developers have not ported all 32-bit applications to 64-bit systems. But openSUSE Leap supports 32-bit application use in 64-bit system environments. This chapter offers a brief overview of 32-bit support implementation on 64-bit openSUSE Leap platforms.

openSUSE Leap 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

openSUSE Leap does not support compilation of 32-bit applications. It only offers runtime support for 32-bit binaries.

8.1 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). openSUSE Leap 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. Unfortunately, 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 (File System Hierarchy Standard).

8.2 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

Some 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.

9 Introduction to the Boot Process

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 means of the boot loader GRUB 2. After this point, the boot process is completely controlled by the operating system and handled by systemd. systemd provides a set of “targets” that boot configurations for everyday usage, maintenance or emergencies.

9.1 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. Once 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. openSUSE Leap 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.

9.2 The Linux Boot Process

The Linux boot process consists of several stages, each represented by a different component:

1. [Section 9.2.1, “The Initialization and Boot Loader Phase”](#)
2. [Section 9.2.2, “The Kernel Phase”](#)
3. [Section 9.2.3, “The init on initramfs Phase”](#)
4. [Section 9.2.4, “The systemd Phase”](#)

9.2.1 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 significantly between hardware architectures.

openSUSE Leap 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 12, The Boot Loader GRUB 2](#).

9.2.1.1 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. Once 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. This is completely independent of local hard disks.

9.2.2 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`.

9.2.2.1 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`.

9.2.2.1.1 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 will inform you about this by showing the output of the command that generates the `initramfs`. However, there are some occasions when you need to regenerate an `initramfs` manually:

- *Adding Drivers Because of Hardware Changes*
- *Moving System Directories to a RAID or LVM*
- *Adding Disks to an LVM Group or Btrfs RAID Containing the Root File System*
- *Changing Kernel Variables*

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 whitespace):

```
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 9.1, "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 9.1, "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 9.1, "Generate an initramfs"*.

Proceed with *Procedure 9.1, "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 9.1, "Generate an initramfs"*.

PROCEDURE 9.1: GENERATE AN INITRAMFS

Note that all commands in the following procedure need to be executed as user `root`.

1. Generate a new `initramfs` file by running

```
dracut MY_INITRAMFS
```

Replace `MY_INITRAMFS` with a file name of your choice. The new `initramfs` will be created as `/boot/MY_INITRAMFS`.

Alternatively, run `dracut -f`. This will overwrite the currently used, existing file.

2. (Skip this step if you ran `dracut -f` in the previous step.) Create a link to the `initramfs` file you created in the previous step:

```
(cd /boot && ln -sf MY_INITRAMFS initrd)
```

9.2.3 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 9.1, “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`. openSUSE Leap 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.

9.2.3.1 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. Note that 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 9.2.2.1, “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 10.6.4, “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.

9.2.4 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 10, The systemd Daemon](#) for details.

10 The systemd Daemon

The program `systemd` is the process with process ID 1. It is responsible for initializing the system in the required way. `systemd` is started directly by the kernel and resists signal 9, which normally terminates processes. All other programs are either started directly by `systemd` or by one of its child processes.

Systemd is a replacement for the System V init daemon. `systemd` is fully compatible with System V init (by supporting init scripts). One of the main advantages of `systemd` is that it considerably speeds up boot time by aggressively paralleling service starts. Furthermore, `systemd` only starts a service when it is really needed. Daemons are not started unconditionally at boot time, but rather when being required for the first time. `systemd` also supports Kernel Control Groups (cgroups), snapshotting and restoring the system state and more. See <http://www.freedesktop.org/wiki/Software/systemd/> for details.

10.1 The systemd Concept

This section will go into detail about the concept behind `systemd`.

10.1.1 What Is systemd

`systemd` is a system and session manager for Linux, compatible with System V and LSB init scripts. The main features are:

- provides aggressive parallelization capabilities
- uses socket and D-Bus activation for starting services
- offers on-demand starting of daemons
- keeps track of processes using Linux cgroups
- supports snapshotting and restoring of the system state
- maintains mount and automount points
- implements an elaborate transactional dependency-based service control logic

10.1.2 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.** Usually 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` see <http://www.freedesktop.org/software/systemd/man/systemd.unit.html> ↗

10.2 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, since there is only one command to memorize for the majority of service-handling 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 `→` to automatically complete it. This feature is only available in the bash shell and requires the installation of the package bash-completion.

10.2.1 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:

```
tux > sudo systemctl start MY_1ST_SERVICE MY_2ND_SERVICE
```

To list all services available on the system:

```
tux > sudo systemctl list-unit-files --type=service
```

The following table lists the most important service management commands for systemd and System V init:

TABLE 10.1: 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 will be 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 <code>httpd.conf</code> configuration file. Note that not all services support reloading. | reload | reload |
| Reloading or restarting. Reloads services if reloading is supported, otherwise restarts them. If a service is not yet running it will be started. | reload-or-restart | n/a |
| 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 |

| Task | systemd Command | System V init Command |
|--|-----------------|-----------------------|
| Getting detailed status information. Lists information about the status of services. The <code>systemd</code> command shows details such as description, executable, status, cgroup, and messages last issued by a service (see Section 10.6.9, "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 |

10.2.2 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.

10.2.2.1 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 10.2: COMMANDS FOR ENABLING AND DISABLING SERVICES

| Task | <u>systemd Command</u> | System V init Command |
|--|--|---|
| Enabling. | <u>systemctl enable MY_SERVICE(S)</u> | <u>insserv MY_SERVICE(S), chkconfig -a MY_SERVICE(S)</u> |
| Disabling. | <u>systemctl disable MY_SERVICE(S).service</u> | <u>insserv -r MY_SERVICE(S), chkconfig -d MY_SERVICE(S)</u> |
| Checking. Shows whether a service is enabled or not. | <u>systemctl is-enabled MY_SERVICE</u> | <u>chkconfig MY_SERVICE</u> |
| 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. | <u>systemctl reenable MY_SERVICE</u> | n/a |
| Masking. After “disabling” a service, it can still be started manually. To completely disable a service, you need to mask it. Use with care. | <u>systemctl mask MY_SERVICE</u> | n/a |
| Unmasking. A service that has been masked can only be used again after it has been unmasked. | <u>systemctl unmask MY_SERVICE</u> | n/a |

10.3 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.

10.3.1 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 10.4, “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 an emergency 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 system without network.

To remain compatible with the System V init runlevel system, systemd provides special targets named runlevelX.target mapping the corresponding runlevels numbered X.

If you want to know the current target, use the command: **systemctl get-default**

TABLE 10.3: SYSTEM V RUNLEVELS AND systemd TARGET UNITS

| System V runlevel | systemd target | Purpose |
|-------------------|--|---|
| 0 | <u>runlevel0.target</u> , <u>halt.target</u> , <u>poweroff.target</u> | System shutdown |
| 1, S | <u>runlevel1.target</u> , <u>rescue.target</u> , | Single-user mode |
| 2 | <u>runlevel2.target</u> , <u>multi-</u> <u>user.target</u> , | Local multiuser without remote network |
| 3 | <u>runlevel3.target</u> , <u>multi-</u> <u>user.target</u> , | Full multiuser with network |
| 4 | <u>runlevel4.target</u> | Unused/User-defined |
| 5 | <u>runlevel5.target</u> , <u>graphical.target</u> , | Full multiuser with network and display manager |

| System V runlevel | <u>systemd</u> target | Purpose |
|-------------------|---|---------------|
| 6 | <u>runlevel6.target</u> , <u>reboot.target</u> , | 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 10.5.3, "Creating Custom Targets"](#) for instructions on how to create your own bootable target.

10.3.1.1 Commands to Change Targets

Use the following commands to operate with target units:

| Task | <u>systemd</u> Command | System V init Command |
|---|---|--|
| Change the current target/ runlevel | <u>systemctl isolate</u> <u>MY_TARGET.target</u> | <u>telinit</u> <u>X</u> |
| Change to the default target/ runlevel | <u>systemctl default</u> | n/a |
| Get the current target/ runlevel | <u>systemctl list-units --type=target</u> With <u>systemd</u> there is usually more than one active target. The command lists all currently active targets. | <u>who -r</u> or <u>runlevel</u> |
| persistently change the default runlevel | Use the Services Manager or run the following command: <u>ln -sf /usr/lib/systemd/system/</u> <u>MY_TARGET.target /etc/systemd/system/</u> <u>default.target</u> | Use the Services Manager or change the line <u>id: X:initdefault:</u> <u>in /etc/inittab</u> |

| Task | systemd Command | System V init Command |
|--|---|---|
| Change the default runlevel for the current boot process | Enter the following option at the boot prompt systemd.unit= <u>MY_TARGET.target</u> | Enter the desired runlevel number at the boot prompt. |
| Show a target's/runlevel's dependencies | systemctl show -p "Requires" <u>MY_TARGET.target</u> systemctl show -p "Wants" <u>MY_TARGET.target</u> “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 |

10.3.2 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.

10.3.2.1 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 10.1: LIST ACTIVE SERVICES

```

root # 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 10.2: LIST FAILED SERVICES

```

root # 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

[...]

```

10.3.2.2 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

```

root # systemd-analyze
Startup finished in 2666ms (kernel) + 21961ms (userspace) = 24628ms

```

Listing the Services Start-Up Time

```

root # 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

```

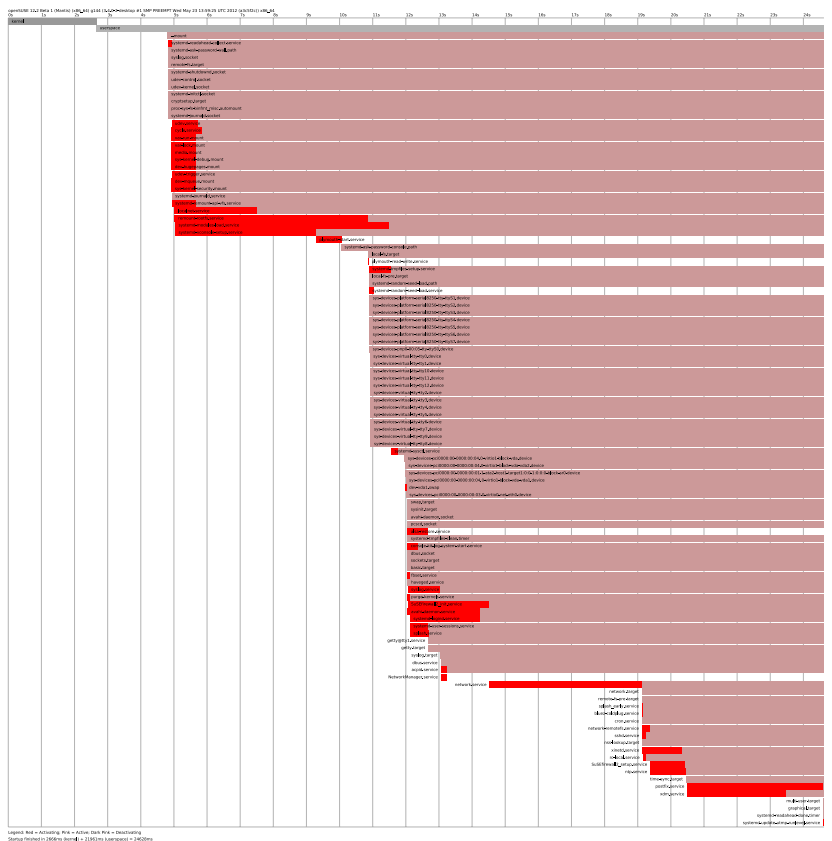
```

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

```
root # systemd-analyze plot > jupiter.example.com-startup.svg
```



10.3.2.3 Review the Complete Start-Up Process

The above-mentioned commands let you review the services that started and the time it took to start them. If you need to know more details, you can tell `systemd` to verbosely log the complete start-up procedure by entering the following parameters at the boot prompt:

```
systemd.log_level=debug systemd.log_target=kmsg
```

Now `systemd` writes its log messages into the kernel ring buffer. View that buffer with `dmesg`:

```
tux > dmesg -T | less
```

10.3.3 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 will result 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 will also be 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` here.

2. Start the daemon with `systemctl start APPLICATION`.

10.4 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*.

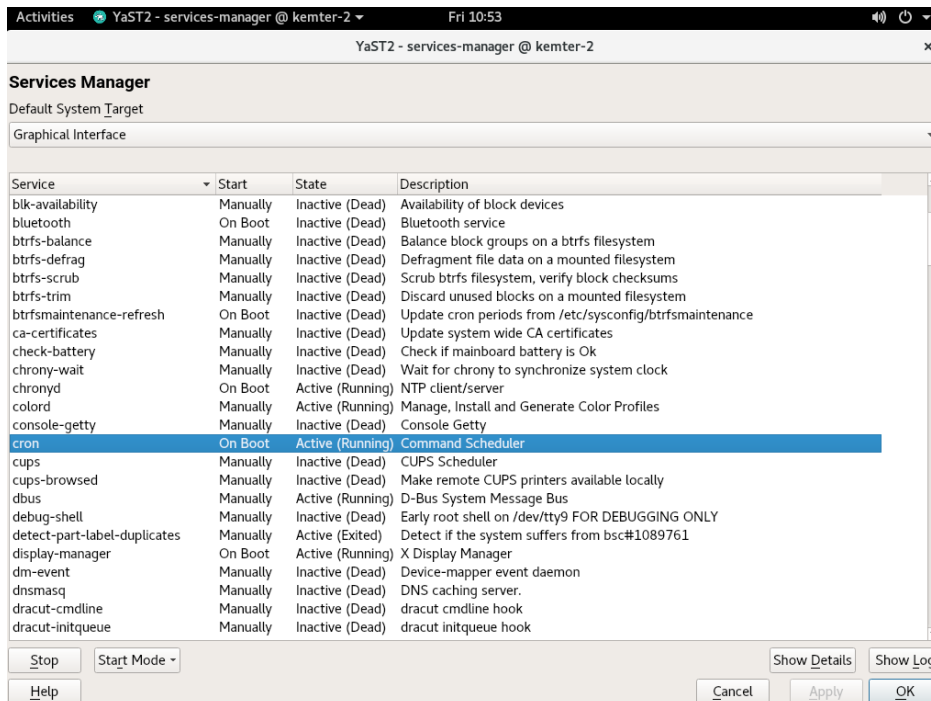


FIGURE 10.1: SERVICES MANAGER

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 *Active* column shows whether it is currently running (*Active*) or not (*Inactive*). Toggle its status by choosing *Start/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.

Enabling or Disabling a Service

Select a service from the table. The *Enabled* column shows whether it is currently *Enabled* or *Disabled*. Toggle its status by choosing *Enable/Disable*.

By enabling or disabling a service you configure whether it is started during booting (*Enabled*) or not (*Disabled*). This setting will not affect the current session. To change its status in the current session, you need to start or stop it.

View a Status Messages

To view the status message of a service, select it from the list and choose *Show Details*. The output you will see is identical to the one generated by the command `systemctl -l status MY_SERVICE`.



Warning: Faulty Runlevel Settings May Damage Your System

Faulty runlevel settings may make your system unusable. Before applying your changes, make absolutely sure that you know their consequences.

10.5 Customizing systemd

The following sections contain some examples for `systemd` customization.



Warning: Avoiding Your Customization from Being Overwritten

When customizing `systemd`, always use the directory `/etc/systemd/`, *never* use `/usr/lib/systemd/`. Otherwise your changes will be overwritten by the next update of `systemd`.

10.5.1 Customizing Unit Files

The recommended way to customize unit files is to use the `systemctl edit SERVICE` command. This command starts the default text editor and creates a directory with the `override.conf` file in `/etc/systemd/system/NAME.service.d/`. The command also ensures that the running `systemd` process is notified about the changes.

Alternatively, you can open a copy of the original file for editing instead of a blank file by running `systemctl edit --full SERVICE`. When editing the file, make sure that you do not remove any of the existing sections.

As an exercise, change how long the system waits for MariaDB to start. As root, run **systemctl edit --full mariadb.service**. The file opened will look similar to the following:

```
[Unit]
Description=MySQL server
Wants=basic.target
Conflicts=mariadb.target
After=basic.target network.target

[Install]
WantedBy=multi-user.target
Alias=mysql.service

[Service]
Restart=on-abort
Type=notify
ExecStartPre=/usr/lib/mysql/mysql-systemd-helper install
ExecStartPre=/usr/lib/mysql/mysql-systemd-helper upgrade
ExecStart=/usr/lib/mysql/mysql-systemd-helper start

# Configures the time to wait for start-up/stop
TimeoutSec=300

# Prevent writes to /usr, /boot, and /etc
ProtectSystem=full

# Prevent accessing /home, /root and /run/user
ProtectHome=true

UMask=007
```

Adjust the TimeoutSec value and save the changes. To enable the changes, as root, run **systemctl daemon-reload**.

For further information, refer to the man pages that can be evoked with the man 1 systemctl command.

10.5.1.1 Converting xinetd Services to systemd

Since the release of openSUSE Leap 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:

```
root # 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:

```
root # cat /usr/lib/systemd/system/example.socket
[Socket]
ListenStream=0.0.0.0:10085
Accept=false

[Install]
WantedBy=sockets.target
```

```
root # 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](#)).

10.5.2 Creating “Drop-in” Files

If you only want to add a few lines to a configuration file or modify a small part of it, you can use so-called “drop-in” files. Drop-in files let you extend the configuration of unit files without having to edit or override the unit files themselves.

For example, to change one value for the `FOOBAR` service located in `/usr/lib/systemd/system/FOOBAR.SERVICE`, proceed as follows:

1. Create a directory called `/etc/systemd/system/FOOBAR.service.d/`.
Note the `.d` suffix. The directory must otherwise be named like the service that you want to patch with the drop-in file.
2. In that directory, create a file `WHATEVERMODIFICATION.conf`.
Make sure it only contains the line with the value that you want to modify.
3. Save your changes to the file. It will be used as an extension of the original file.

10.5.3 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:

```
tux > sudo systemctl daemon-reload
```

10.6 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 <http://0pointer.de/blog/projects> ↗.

10.6.1 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:

```
tux > 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](#).

10.6.2 System Log

[Section 10.6.9, “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.

10.6.3 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

```
tux > sudo systemctl snapshot MY_SNAPSHOT.snapshot
```

Delete a Snapshot

```
tux > sudo systemctl delete MY_SNAPSHOT.snapshot
```

View a Snapshot

```
tux > sudo systemctl show MY_SNAPSHOT.snapshot
```

Activate a Snapshot

```
tux > sudo systemctl isolate MY_SNAPSHOT.snapshot
```

10.6.4 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` takes precedence.

For more information, see the `modules-load.d(5)` man page.

10.6.5 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`):

```
tux > sudo systemctl daemon-reload
tux > sudo systemctl enable before
```

Every time you modify the service file, you need to run:

```
tux > sudo systemctl daemon-reload
```

10.6.6 Kernel Control Groups (cgroups)

On a traditional System V init system it is not always possible to clearly assign a process to the service that spawned it. Some 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 terminate correctly, leaving some 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`. The result will look like the following (shortened) example:

EXAMPLE 10.3: LIST ALL PROCESSES BELONGING TO A SERVICE

```
root # 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 9 “Kernel Control Groups” for more information about cgroups.

10.6.7 Terminating Services (Sending Signals)

As explained in [Section 10.6.6, “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 terminate a service and all of its children. Child processes that have not been terminated will remain as zombie processes.

systemd's concept of confining each service into a cgroup makes it possible to clearly 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.

```
tux > sudo systemctl kill MY_SERVICE
```

Sending `SIGNAL` to a Service

Use the `-s` option to specify the signal that should be sent.

```
tux > 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`:

```
tux > sudo systemctl kill -s SIGHUP --kill-who=main MY_SERVICE
```

10.6.8 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.

Terminating `dbus` or restarting it in the running system is similar to an attempt to terminate or restart pid 1. It will break systemd client/server communication and make 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.

10.6.9 Debugging Services

By default, systemd is not overly verbose. If a service was started successfully, no output will be produced. In case of a failure, a short error message will be displayed. However, `systemctl status` provides means to debug 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:

```
root # systemctl start apache2
Job failed. See system journal and 'systemctl status' for details.
root # 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:

```
tux > sudo systemctl status chronyd
tux > 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`:

```
tux > 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.

10.7 More Information

For more information on systemd refer to the following online resources:

Homepage

<http://www.freedesktop.org/wiki/Software/systemd> ↗

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 <http://0pointer.de/blog/projects> ↗.

11 journalctl: Query the systemd Journal

When `systemd` replaced traditional init scripts in openSUSE Leap (see [Chapter 10, The systemd Daemon](#)), it introduced its own logging system called `journal`. There is no need to run a `syslog` based service anymore, as all system events are written in 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:

```
tux > 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
[...]
```

11.1 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, the directory `/var/log/journal/` must exist with correct ownership and permissions so the `systemd-journald` service can store its data. `systemd` will create the directory for you—and switch to persistent logging—if you do the following:

1. As `root`, open `/etc/systemd/journald.conf` for editing.

```
root # vi /etc/systemd/journald.conf
```

2. Uncomment the line containing `Storage=` and change it to

```
[...]
[Journal]
Storage=persistent
#Compress=yes
```



```
[...]
```

3. Save the file and restart systemd-journald:

```
root # systemctl restart systemd-journald
```

11.2 journalctl Useful Switches

This section introduces several common useful options to enhance the default `journalctl` behavior. All switches are described in the `journalctl` manual 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:

```
tux > 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.

-e

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 11.3.3, "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 11.3.3, "Filtering Based on Fields"](#)).

```
tux > 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.
```

11.3 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.

11.3.1 Filtering Based on a Boot Number

journalctl can filter messages based on a specific system boot. To list all available boots, run

```
tux > 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 prior to 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:

```
tux > 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:

```
tux > sudo journalctl -b -1
```

Another way is to list boot messages based on the boot ID. For this purpose, use the `_BOOT_ID` field:

```
tux > sudo journalctl _BOOT_ID=156019a44a774a0bb0148a92df4af81b
```

11.3.2 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:

```
tux > sudo journalctl --since "now" -f
```

Show all messages since last midnight till 3:20am:

```
tux > sudo journalctl --since "today" --until "3:20"
```

11.3.3 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:

```
tux > 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):

```
tux > sudo journalctl _TRANSPORT=kernel
```

Show messages from the service's standard or error output:

```
tux > sudo journalctl _TRANSPORT=stdout
```

Show messages produced by a specified service only:

```
tux > 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:

```
tux > sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=1488
```

If two matches refer to the same field, all entries matching either expression are shown:

```
tux > 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:

```
tux > sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=1480 +
_SYSTEMD_UNIT=dbus.service
```

11.4 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:

```
tux > 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:

```
tux > 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:

```
tux > 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 [...]
```

11.5 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](#). Note that you need to restart the journal for the changes to take effect with

```
tux > sudo systemctl restart systemd-journald
```

11.5.1 Changing the Journal Size Limit

If the journal log data is saved to a persistent location (see [Section 11.1, "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
```

11.5.2 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
```

11.5.3 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.

```
tux > sudo rpm -q rsyslog
rsyslog-7.4.8-2.16.x86_64
```

- `rsyslog` service is enabled.

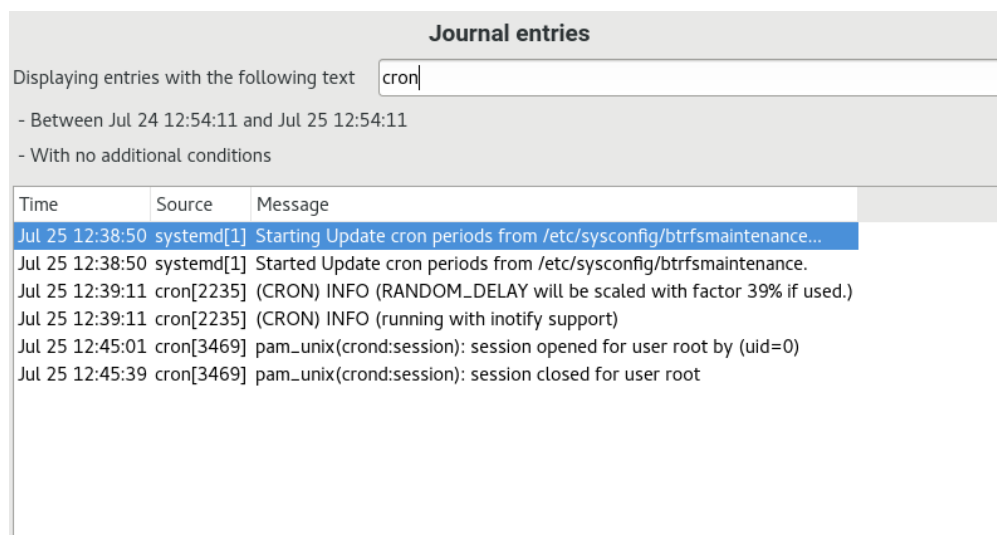
```
tux > sudo systemctl is-enabled rsyslog
enabled
```

- Forwarding to syslog is enabled in `/etc/systemd/journald.conf`.

```
ForwardToSyslog=yes
```

11.6 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`.



The screenshot shows the YaST Systemd Journal interface. At the top, it says "Journal entries". Below that, there is a search bar with the text "cron" entered. Underneath the search bar, it says "Displaying entries with the following text" and "cron". There are also two filter options: "- Between Jul 24 12:54:11 and Jul 25 12:54:11" and "- With no additional conditions". Below these filters is a table with three columns: "Time", "Source", and "Message". The table contains several entries related to cron jobs and pam_unix sessions.

| Time | Source | Message |
|-----------------|------------|--|
| Jul 25 12:38:50 | systemd[1] | Starting Update cron periods from /etc/sysconfig/btrfsmaintenance... |
| Jul 25 12:38:50 | systemd[1] | Started Update cron periods from /etc/sysconfig/btrfsmaintenance. |
| Jul 25 12:39:11 | cron[2235] | (CRON) INFO (RANDOM_DELAY will be scaled with factor 39% if used.) |
| Jul 25 12:39:11 | cron[2235] | (CRON) INFO (running with inotify support) |
| Jul 25 12:45:01 | cron[3469] | pam_unix(cron:session): session opened for user root by (uid=0) |
| Jul 25 12:45:39 | cron[3469] | pam_unix(cron:session): session closed for user root |

FIGURE 11.1: 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.

11.7 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**.

12 The Boot Loader GRUB 2

This chapter describes how to configure GRUB 2, the boot loader used in openSUSE® Leap. It is the successor to the traditional GRUB boot loader—now called “GRUB Legacy”. A YaST module is available for configuring the most important settings. The boot procedure as a whole is outlined in [Chapter 9, Introduction to the Boot Process](#). For details on Secure Boot support for UEFI machines, see [Chapter 14, UEFI \(Unified Extensible Firmware Interface\)](#).

12.1 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.

12.2 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 openSUSE Leap 42.2 these entries will also be parsed when using `grub-once`.

/etc/default/grub

This file controls the user settings of GRUB 2 and usually 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 some basic settings like the boot loader type and whether to enable UEFI Secure Boot support.

/boot/grub2/x86_64-efi,

These configuration files contain architecture-specific options.

GRUB 2 can be controlled in various 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 will automatically run this command.

12.2.1 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 12.3, “Configuring the Boot Loader with YaST”](#).

12.2.2 The File `/etc/default/grub`

More general options of GRUB 2 belong here, 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:

```
tux > grep "export GRUB_DEFAULT" -A50 /usr/sbin/grub2-mkconfig | grep GRUB_
```

In addition to already defined variables, the user may introduce their own 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 set in this file are general options that affect all boot entries. Specific options for Xen kernels or the Xen hypervisor can be set via the `GRUB*_XEN_*` configuration options. See below for details.

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 will cause 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 mode. 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 will completely replace 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 will only replace parameters of GRUB_CMDLINE_LINUX_DEFAULT.

GRUB_CMDLINE_XEN

This entry specifies the kernel parameters for the Xen guest kernel only—the operation principle is the same as for GRUB_CMDLINE_LINUX.

GRUB_CMDLINE_XEN_DEFAULT

Same as GRUB_CMDLINE_XEN—the operation principle is the same as for GRUB_CMDLINE_LINUX_DEFAULT.

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. Note that 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 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 will be 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 3.3, "System Rollback by Booting from Snapshots"*.

For a complete list of options, see the [GNU GRUB manual \(http://www.gnu.org/software/grub/manual/grub.html#Simple-configuration\)](http://www.gnu.org/software/grub/manual/grub.html#Simple-configuration).

12.2.3 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. If you want 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 will be 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 will be automatically included into /boot/grub2/grub.cfg just after 40_custom at boot time.

12.2.4 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 usually sufficient, particularly in the case of 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. Note that GRUB 2 partition numbers start with 1 and not with 0 as in GRUB Legacy.

```
(hd1) /dev/disk-by-id/DISK3 ID
(hd2) /dev/disk-by-id/DISK1 ID
(hd3) /dev/disk-by-id/DISK2 ID
```

12.2.5 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.



```
GNU GRUB version 2.04

set root='hd0,gpt2'
if [ x$feature_platform_search_hint = xy ]; then
  search --no-floppy --fs-uuid --set=root --hint='hd0,gpt2' 3c2\
51c37-7ebb-4aaa-a658-eca1e810198d
else
  search --no-floppy --fs-uuid --set=root 3c251c37-7ebb-4aaa-a65\
8-eca1e810198d
fi
echo      'Loading Linux 5.3.18-8-default ...'
linux     /boot/vmlinuz-5.3.18-8-default root=UUID=3c251c37-7\
ebb-4aaa-a658-eca1e810198d ${extra_cmdline} splash=silent resume=/dev/v\
da4 mitigations=auto quiet crashkernel=195M,high crashkernel=72M,low
echo      'Loading initial ramdisk ...'
initrd   /boot/initrd-5.3.18-8-default

Minimum Emacs-like screen editing is supported. TAB lists
completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for a
command-line or ESC to discard edits and return to the GRUB
menu.
```

FIGURE 12.1: 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 <http://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 "Start-Up", Chapter 4 "Troubleshooting", Section 4.3 "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 will be permanently saved in the installed system.

12.2.6 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 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`:

```
tux > 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:

```
tux > sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

After you reboot, you will be prompted 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 will boot the selected boot entry.

For more information, see <https://www.gnu.org/software/grub/manual/grub.html#Security>.

12.3 Configuring the Boot Loader with YaST

The easiest way to configure general options of the boot loader in your openSUSE Leap 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.

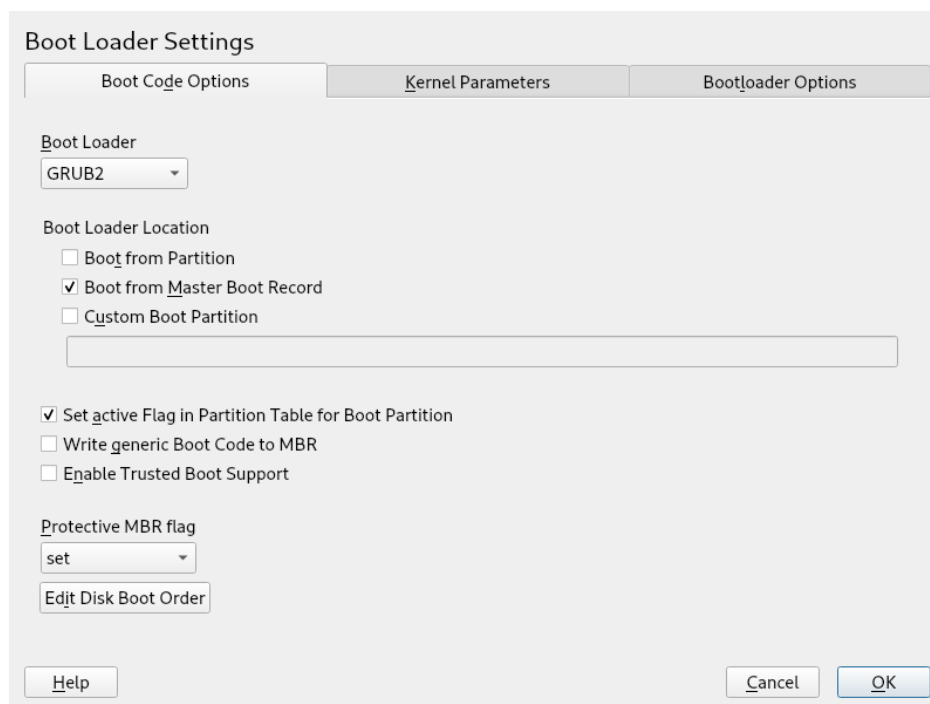


FIGURE 12.2: BOOT CODE OPTIONS

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.

12.3.1 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 12.1: 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 Boot Partition

Use this option to specify the location of the boot loader manually.

2. Click *OK* to apply your changes.

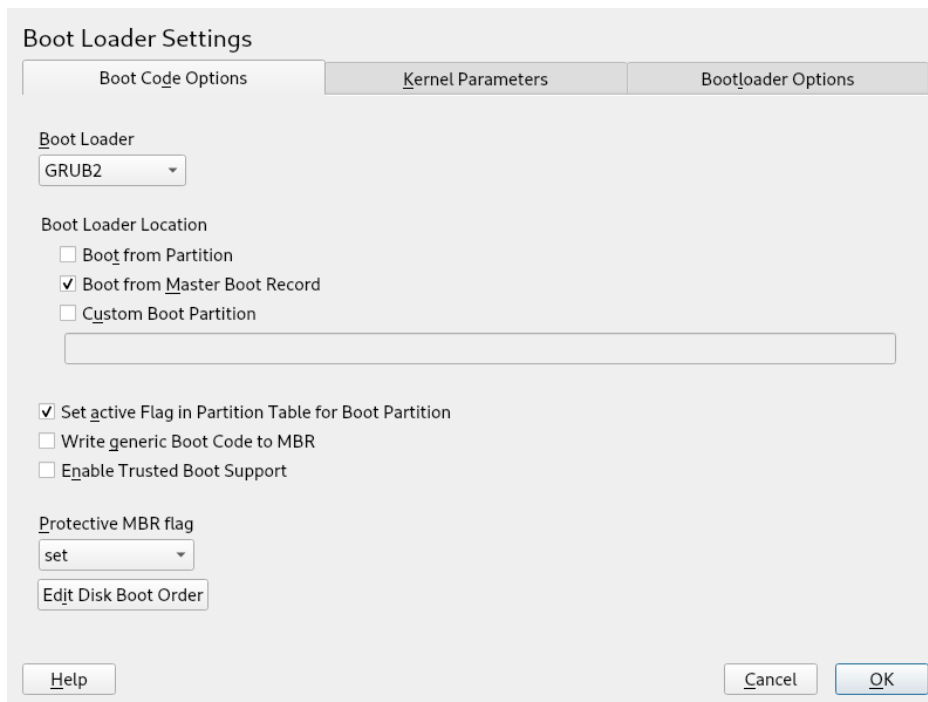


FIGURE 12.3: CODE OPTIONS

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.

12.3.2 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 openSUSE Leap is installed by default. The rest of the list is a hint for GRUB 2's device mapper (see [Section 12.2.4, "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 12.2: 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.

12.3.3 Configuring Advanced Options

Advanced boot parameters can be configured via the *Boot Loader Options* tab.

12.3.3.1 *Boot Loader Options* Tab

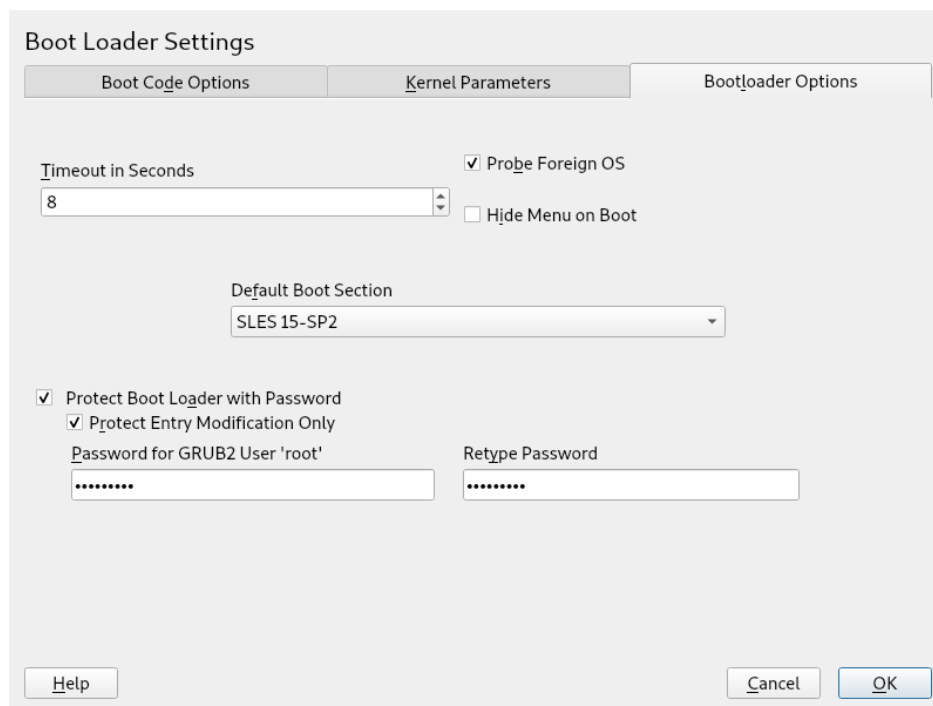


FIGURE 12.4: **BOOT LOADER OPTIONS**

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 12.2.6, “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.

12.3.3.2 Kernel Parameters Tab

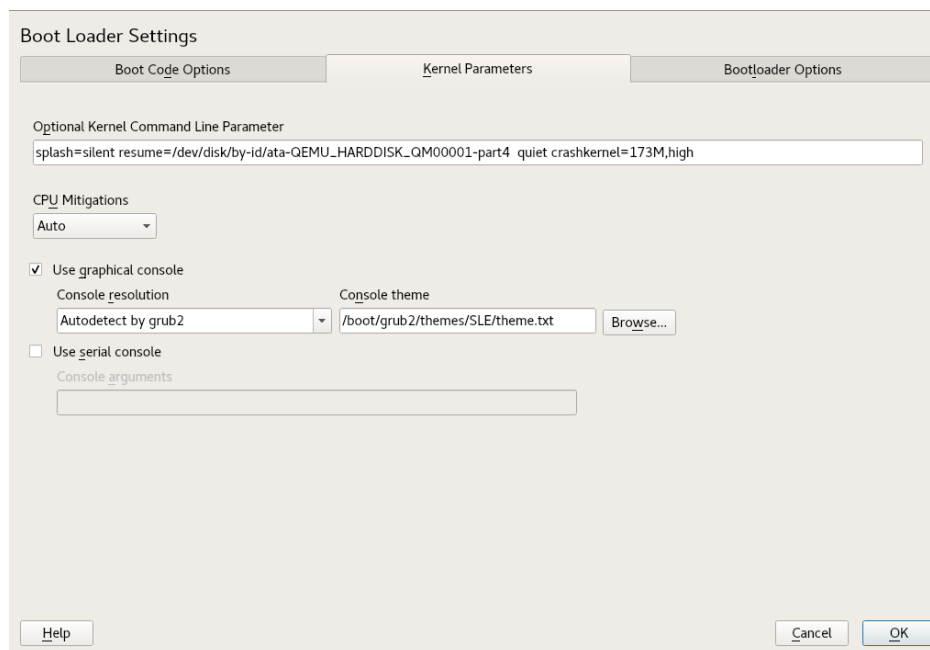


FIGURE 12.5: KERNEL PARAMETERS

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 <http://www.gnu.org/software/grub/manual/grub.html#Serial-terminal> ↗

12.4 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 12.1: 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 Shim and regenerate the UEFI boot table. To do so, use:

```
root # shim-install --config-file=/boot/grub2/grub.cfg
```

grub2-mkrescue

Creates a bootable rescue image of your installed GRUB 2 configuration.

EXAMPLE 12.2: USAGE OF GRUB2-MKRESCUE

```
grub2-mkrescue -o save_path/name.iso iso
```

grub2-script-check

Checks the given file for syntax errors.

EXAMPLE 12.3: 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 12.4: 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.

12.5 More Information

Extensive information about GRUB 2 is available at <https://www.gnu.org/software/grub/> . Also refer to the [grub](#) info page.

13 Basic Networking

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 various 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 various 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 <http://www.ietf.org/rfc.html>.

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 13.1, "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.

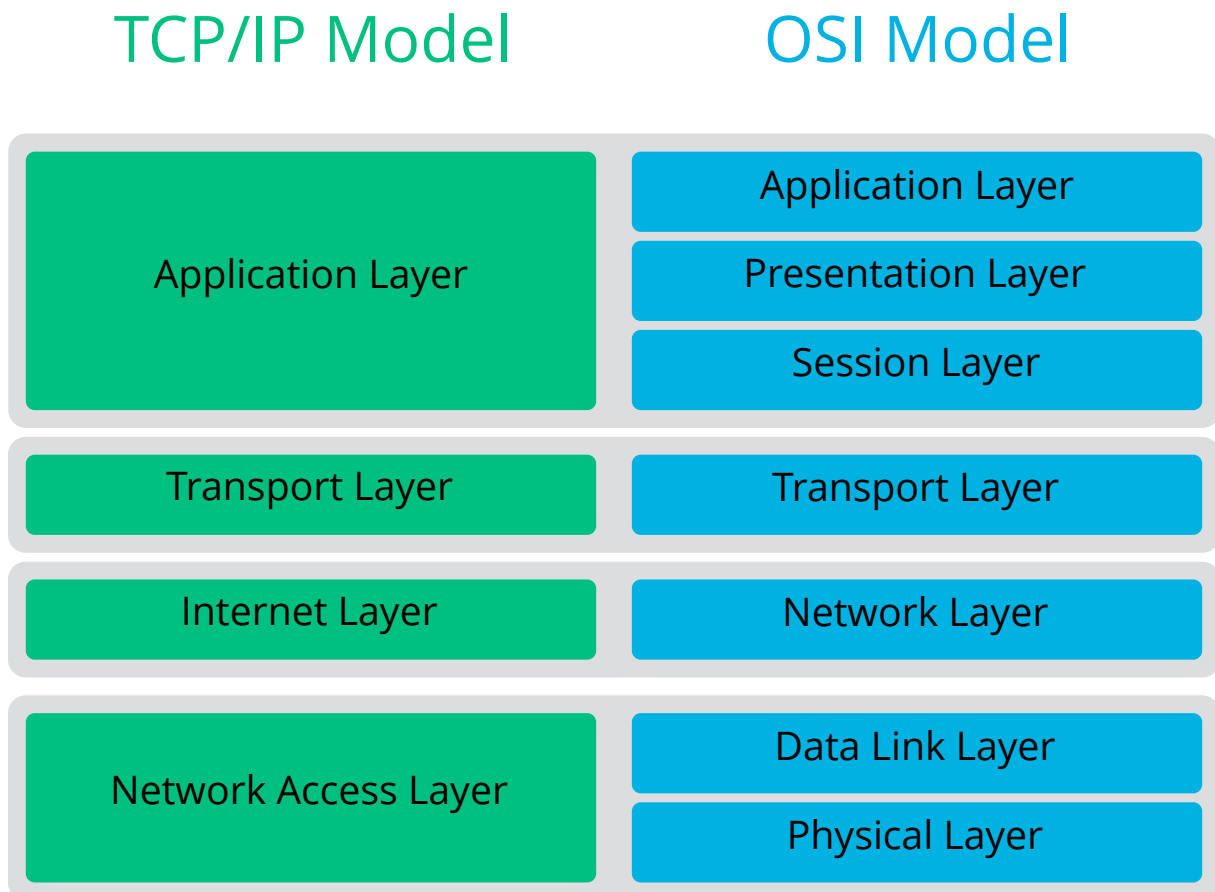


FIGURE 13.1: SIMPLIFIED LAYER MODEL FOR TCP/IP

The diagram provides one or two examples for each layer. The layers are ordered according to *abstraction levels*. The lowest layer is very 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 mostly 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 quite smaller, as the network hardware can be a limiting factor. The maximum size of a data packet on an Ethernet is about fifteen hundred bytes. The size of a TCP/IP packet is limited to this amount when the data is sent over an 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 13.2, "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.

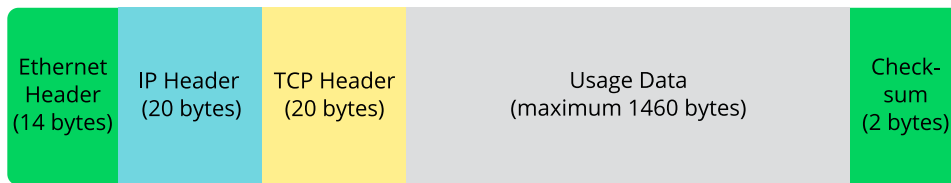


FIGURE 13.2: TCP/IP ETHERNET PACKET

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 is responsible for 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 is responsible for 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, as long as packets are in the correct format.

13.1 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 13.2, “IPv6—The Next Generation Internet”](#).

13.1.1 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 13.1, “Writing IP Addresses”](#).

EXAMPLE 13.1: 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.

13.1.2 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 13.2, “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 13.2, “Linking IP Addresses to the Netmask”](#) the first net with 24 bits could also be written as 192.168.0.0/24.

EXAMPLE 13.2: 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 usually 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 13.2, "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 if you want 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 13.1, "Private IP Address Domains"*.

TABLE 13.1: PRIVATE IP ADDRESS DOMAINS

| Network/Netmask | Domain |
|----------------------------------|--------------------------------|
| <u>10.0.0.0 / 255.0.0.0</u> | <u>10.x.x.x</u> |
| <u>172.16.0.0 / 255.240.0.0</u> | <u>172.16.x.x – 172.31.x.x</u> |
| <u>192.168.0.0 / 255.255.0.0</u> | <u>192.168.x.x</u> |

13.2 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 (<http://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, quite a few IP addresses are lost—they cannot be used because of the way in which 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. The problem with them lies in their configuration, which is a chore to set up and a burden to maintain. 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.

13.2.1 Advantages

The most important and most visible improvement brought by the new 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 13.2.2, “Address Types and Structure”](#).

The following is a list of other advantages of the new protocol:

Autoconfiguration

IPv6 makes the network “plug and play” capable, which means that a newly set up 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 13.2.3, “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 completely separate, such that there is no interference between the two protocol versions.

Custom Tailored Services through Multicasting

With IPv4, some 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 some predefined groups to address all name servers (the *all name servers multicast group*), for example, or all routers (the *all routers multicast group*).

13.2.2 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 for some reason, 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 13.3, "Sample IPv6 Address"*, where all three lines represent the same address.

EXAMPLE 13.3: 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 13.4, "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 13.4: IPV6 ADDRESS SPECIFYING THE PREFIX LENGTH

```
fe80::10:1000:1a4/64
```

IPv6 knows about several predefined types of prefixes. Some are shown in *Various IPv6 Prefixes*.

VARIOUS 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 substantially

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 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 13.2.3, "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 completely 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 completely 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 the completely 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. On the other hand, those packets destined to the care-of address are directly transferred to the mobile host without any special detours.

13.2.3 Coexistence of IPv4 and IPv6

The migration of all hosts connected to the Internet from IPv4 to IPv6 is a gradual process. Both protocols will coexist for some 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 13.2.2, "Address Types and Structure"](#)).

IPv6 hosts that are more or less 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 huge local area network (LAN). This makes it possible to determine the receiving end of the IPv4 tunnel automatically. However, this method does not scale very 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.

13.2.4 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 `zebra/quagga` for automatic configuration of both addresses and routing.

For information about how to set up various types of tunnels using the `/etc/sysconfig/network` files, see the man page of `ifcfg-tunnel` (`man ifcfg-tunnel`).

13.2.5 For More Information

The above overview does not cover the topic of IPv6 comprehensively. For a more in-depth look at the new protocol, refer to the following online documentation and books:

<http://www.ipv6.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.

<http://www.bieringer.de/linux/IPv6/> ↗

Here, find the Linux IPv6-HOWTO and many links related to the topic.

RFC 2460

The fundamental RFC about IPv6.

IPv6 Essentials

A book describing all the important aspects of the topic is *IPv6 Essentials* by Silvia Hagen (ISBN 0-596-00125-8).

13.3 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 usually 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 quite 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. In addition to that, 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 readily 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 <http://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 configuration of name server access with openSUSE® Leap is described in [Section 13.4.1.4, “Configuring Host Name and DNS”](#). Setting up your own name server is described in [Chapter 19, The Domain Name System](#).

The protocol `whois` is closely related to DNS. With this program, quickly find out who is responsible 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` manual 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>.

13.4 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 13.6, “Configuring a Network Connection Manually”](#).

All network interfaces with link up (with a network cable connected) are 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 openSUSE Leap.

13.4.1 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 13.4.1.1, “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 13.4.1.3, “Configuring an Undetected Network Card”](#). To change the configuration of an already configured card, see [Section 13.4.1.2, “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 13.4.1.4, “Configuring Host Name and DNS”](#).

The *Routing* tab is used for the configuration of routing. See [Section 13.4.1.5, “Configuring Routing”](#) for more information.

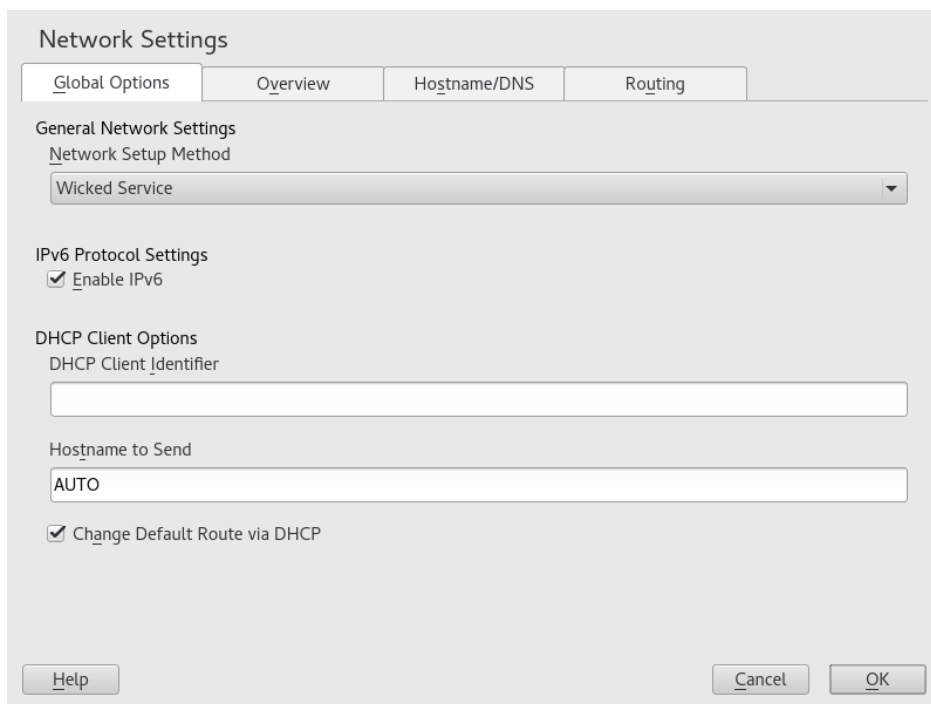


FIGURE 13.3: CONFIGURING NETWORK SETTINGS

13.4.1.1 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 28, 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*.

13.4.1.2 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.

13.4.1.2.1 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.

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 13.6.2.5, “/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 13.4.1.4, “Configuring Host Name and DNS”](#). To configure a gateway, proceed as described in [Section 13.4.1.5, “Configuring Routing”](#).

13.4.1.2.2 Configuring Multiple Addresses

One network device can have multiple IP addresses.



Note: Aliases Are a Compatibility Feature

These so-called aliases or labels, respectively, work with IPv4 only. With IPv6 they will be ignored. Using [iproute2](#) network interfaces can have one or more addresses.

Using YaST to set additional addresses for your network card, 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*. Do not include the interface name in the alias name.
4. To activate the configuration, confirm the settings.

13.4.1.2.3 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 *Hardware* 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.

13.4.1.2.4 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.

13.4.1.2.5 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 13.4.1.2.5, “Activating the Network Device”](#) and choose *On NFSroot* in the *Device Activation* pane.

13.4.1.2.6 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.

13.4.1.2.7 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.

13.4.1.2.8 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>.

13.4.1.2.9 Configuring the Firewall

Without having to perform the detailed firewall setup as described in *Book "Security and Hardening Guide", Chapter 25 "Masquerading and Firewalls", Section 25.4 "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.

13.4.1.3 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 13.4.1.2, “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.

13.4.1.4 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*.

Assign Hostname to Loopback IP associates your host name with 127.0.0.2 (loopback) IP address in /etc/hosts. This is a useful option if you want to have the host name resolvable at all times, even without active network.

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:

```
root # yast dns edit hostname=HOSTNAME
```

To change the name servers, use the following commands:

```
root # yast dns edit nameserver1=192.168.1.116
root # yast dns edit nameserver2=192.168.1.117
root # yast dns edit nameserver3=192.168.1.118
```

13.4.1.5 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 route with the highest metric is used as default. If the network device is disconnected, its route will be removed and the next one will be used. However, the current kernel does not use metric in static routing, only routing daemons like `multipathd` do.

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.

13.5 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.

13.5.1 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 28.2, “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 28, 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.

13.5.2 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 28.3, "Configuring Network Connections"](#).

13.5.3 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 19 "Authorization with PolKit"*.

13.6 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.

13.6.1 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.

13.6.1.1 wicked Architecture and Features

The `wicked` service comprises several parts as depicted in *Figure 13.4, "wicked architecture"*.

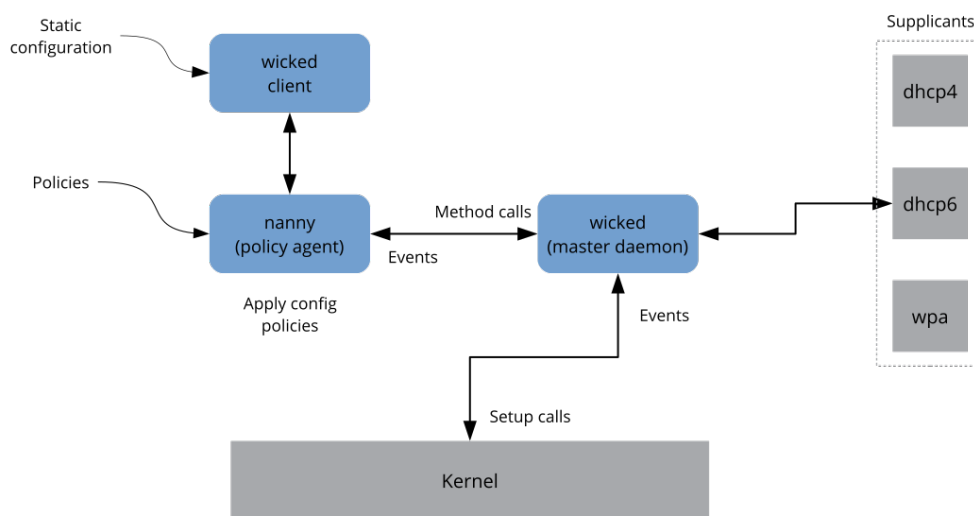


FIGURE 13.4: wicked ARCHITECTURE

`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 13.6.1.3, “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.

13.6.1.2 Using wicked

On openSUSE Leap, wicked runs by default on desktop or server hardware. On mobile hardware NetworkManager 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.

13.6.1.3 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).

13.6.1.4 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
```

13.6.1.5 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`](#).

13.6.1.6 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.

13.6.1.7 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.

13.6.1.8 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.

13.6.2 Configuration Files

This section provides an overview of the network configuration files and explains their purpose and the format used.

13.6.2.1 `/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:

```
tux > 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.

13.6.2.2 `/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.

13.6.2.3 `/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.

13.6.2.4 [/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.

13.6.2.5 [/etc/sysconfig/network/ifcfg-*](#)

These files contain the traditional configurations for network interfaces. In openSUSE prior to Leap, this was the only supported format besides iBFT firmware.



Note: **wicked** and the `ifcfg-*` Files

wicked reads these files if you specify the `compat:` prefix. According to the openSUSE Leap 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 `macvtap` 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 13.6.2.6, “/etc/sysconfig/network/config, /etc/sysconfig/network/dhcp, and /etc/sysconfig/network/wireless”](#).

13.6.2.6 `/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 openSUSE prior to Leap, DHCPv6 used to work even on networks where IPv6 Router Advertisements (RAs) were not configured properly. Starting with openSUSE Leap, DHCPv6 will correctly require 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
```

13.6.2.7 `/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 13.5: 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
```

13.6.2.8 `/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 13.6, “/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`).

```
# Our domain
search example.com
#
# We use dns.example.com (192.168.1.116) as nameserver
nameserver 192.168.1.116
```

13.6.2.9 /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`.

13.6.2.10 `/etc/hosts`

In this file, shown in *Example 13.7, “/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 13.7: `/etc/hosts`

```
127.0.0.1 localhost
192.168.2.100 jupiter.example.com jupiter
192.168.2.101 venus.example.com venus
```

13.6.2.11 `/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 13.8, “/etc/networks”*.

EXAMPLE 13.8: `/etc/networks`

```
loopback    127.0.0.0
localnet    192.168.0.0
```

13.6.2.12 `/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 13.2, "Parameters for `/etc/host.conf`"* shows the parameters available. A sample `/etc/host.conf` is shown in *Example 13.9, "`/etc/host.conf`".*

TABLE 13.2: PARAMETERS FOR `/ETC/HOST.CONF`

| | |
|---|---|
| <code>order hosts, bind</code> | Specifies in which order the services are accessed for the name resolution. Available arguments are (separated by blank spaces or commas): |
| | <code>hosts</code> : searches the <code>/etc/hosts</code> file |
| | <code>bind</code> : accesses a name server |
| | <code>nis</code> : uses NIS |
| <code>multi on/off</code> | Defines if a host entered in <code>/etc/hosts</code> can have multiple IP addresses. |
| <code>nospoof on spoofalert on/off</code> | These parameters influence the name server <i>spoofing</i> but do not exert any influence on the network configuration. |
| <code>trim domainname</code> | 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 <code>/etc/hosts</code> file, but should still be recognized with the attached domain names. |

EXAMPLE 13.9: `/etc/host.conf`

```
# We have named running
order hosts bind
# Allow multiple address
multi on
```

13.6.2.13 `/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 13.10, “/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 (see *Chapter 19, The Domain Name System*).

EXAMPLE 13.10: `/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 13.3, “Databases Available via `/etc/nsswitch.conf`”*. The configuration options for NSS databases are listed in *Table 13.4, “Configuration Options for NSS “Databases””*.

TABLE 13.3: DATABASES AVAILABLE VIA `/ETC/NSSWITCH.CONF`

| | |
|-----------------------|--|
| <code>aliases</code> | Mail aliases implemented by <code>sendmail</code> ; see <code>man 5 aliases</code> . |
| <code>ethers</code> | Ethernet addresses. |
| <code>netmasks</code> | List of networks and their subnet masks. Only needed, if you use subnetting. |

| | |
|------------------|---|
| <u>group</u> | User groups used by <u>getgrent</u> . See also the man page for <u>group</u> . |
| <u>hosts</u> | Host names and IP addresses, used by <u>gethostbyname</u> and similar functions. |
| <u>netgroup</u> | Valid host and user lists in the network for controlling access permissions; see the <u>netgroup(5)</u> man page. |
| <u>networks</u> | Network names and addresses, used by <u>getnetent</u> . |
| <u>publickey</u> | Public and secret keys for Secure_RPC used by NFS and NIS + . |
| <u>passwd</u> | User passwords, used by <u>getpwent</u> ; see the <u>passwd(5)</u> man page. |
| <u>protocols</u> | Network protocols, used by <u>getprotoent</u> ; see the <u>protocols(5)</u> man page. |
| <u>rpc</u> | Remote procedure call names and addresses, used by <u>getrpcbyname</u> and similar functions. |
| <u>services</u> | Network services, used by <u>getservent</u> . |
| <u>shadow</u> | Shadow passwords of users, used by <u>getspnam</u> ; see the <u>shadow(5)</u> man page. |

TABLE 13.4: CONFIGURATION OPTIONS FOR NSS “DATABASES”

| | |
|---------------------|---|
| <u>files</u> | directly access files, for example, <u>/etc/aliases</u> |
| <u>db</u> | access via a database |
| <u>nis, nisplus</u> | NIS, see also <i>Book “Security and Hardening Guide”, Chapter 4 “Using NIS”</i> |

| | |
|---------------|---|
| <u>dns</u> | can only be used as an extension for <u>hosts</u> and <u>networks</u> |
| <u>compat</u> | can only be used as an extension for <u>passwd</u> , <u>shadow</u> and <u>group</u> |

13.6.2.14 `/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:

```
tux > sudo systemctl restart nscd
```

13.6.2.15 `/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.

13.6.3 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.

13.6.3.1 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:

```
tux > sudo ip link set DEV_NAME
```

For example, to deactivate device eth0, enter

```
tux > sudo ip link set eth0 down
```

To activate it again, use

```
tux > sudo ip link set eth0 up
```



Tip: Disconnecting NIC Device

If you deactivate a device with

```
tux > 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

```
tux > 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

```
tux > sudo ip link set DEV_NAME carrier on
```

After activating a device, you can configure it. To set the IP address, use

```
tux > 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

```
tux > 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

```
tux > sudo ip route add default via gateway_ip_address
```

To display all devices, use

```
tux > sudo ip link ls
```

To display the running interfaces only, use

```
tux > sudo ip link ls up
```

To print interface statistics for a device, enter

```
tux > sudo ip -s link ls DEV_NAME
```

To view additional useful information, specifically about virtual network devices, enter

```
tux > sudo ip -d link ls DEV_NAME
```

Moreover, to view network layer (IPv4, IPv6) addresses of your devices, enter

```
tux > sudo ip addr
```

In the output, you can find information about MAC addresses of your devices. To show all routes, use

```
tux > 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:

```
tux > sudo ip addr help
```

Find the **ip** manual in </usr/share/doc/packages/iproute2/ip-cref.pdf>.

13.6.3.2 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 13.11, "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 13.11: 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
```

13.6.4 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 10, 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 <http://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.

13.7 Basic Router Setup

A router is a networking device that delivers and receives data (network packets) to or from more than one network back and forth. You often use a router to connect your local network to the remote network (Internet) or to connect local network segments. With openSUSE Leap you can build a router with features such as NAT (Network Address Translation) or advanced firewalling.

The following are basic steps to turn openSUSE Leap into a router.

1. Enable forwarding, for example in `/etc/sysctl.d/50-router.conf`

```
net.ipv4.conf.all.forwarding = 1
net.ipv6.conf.all.forwarding = 1
```

Then provide a static IPv4 and IPv6 IP setup for the interfaces. Enabling forwarding disables several mechanisms, for example IPv6 does not accept an IPv6 RA (router advertisement) anymore, which also prevents the creation of a default route.

2. In many situations (for example, when you can reach the same network via more than one interface, or when VPN usually is used and already on “normal multi-home hosts”), you must disable the IPv4 reverse path filter (this feature does not currently exist for IPv6):

```
net.ipv4.conf.all.rp_filter = 0
```

You can also filter with firewall settings instead.

3. To accept an IPv6 RA (from the router on an external, uplink, or ISP interface) and create a default (or also a more specific) IPv6 route again, set:

```
net.ipv6.conf.${ifname}.accept_ra = 2
net.ipv6.conf.${ifname}.autoconf = 0
```

(Note: “`eth0.42`” needs to be written as `eth0/42` in a dot-separated sysfs path.)

More router behavior and forwarding dependencies are described in <https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt>.

To provide IPv6 on your internal (DMZ) interfaces, and announce yourself as an IPv6 router and “autoconf networks” to the clients, install and configure `radvd` in `/etc/radvd.conf`, for example:

```
interface eth0
{
    IgnoreIfMissing on;          # do not fail if interface missed

    AdvSendAdvert on;           # enable sending RAs
    AdvManagedFlag on;         # IPv6 addresses managed via DHCPv6
    AdvOtherConfigFlag on;      # DNS, NTP... only via DHCPv6

    AdvDefaultLifetime 3600;    # client default route lifetime of 1 hour
}
```

```

prefix 2001:db8:0:1::/64    # (/64 is default and required for autoconf)
{
    AdvAutonomous off;      # Disable address autoconf (DHCPv6 only)

    AdvValidLifetime 3600;  # prefix (autoconf addr) is valid 1 h
    AdvPreferredLifetime 1800; # prefix (autoconf addr) is preferred 1/2 h
}
}

```

Configure the firewall to masquerade traffic with NAT from the LAN into the WAN and to block inbound traffic on the WAN interface:

```

tux > sudo firewall-cmd --permanent --zone=external --change-interface=WAN_INTERFACE
tux > sudo firewall-cmd --permanent --zone=external --add-masquerade
tux > sudo firewall-cmd --permanent --zone=internal --change-interface=LAN_INTERFACE
tux > sudo firewall-cmd --reload

```

13.8 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 slave device will become active if the active slave 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 openSUSE Leap. Provides fault tolerance.

2 (balance-xor)

Traffic is split between all available interfaces based on the following policy: [(source MAC address XOR'd with destination MAC address XOR packet type ID) modulo slave count] 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.

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*.

Network Card Setup

General Address Hardware Bond Slaves

Device Type: Bond Configuration Name: bond0

No Link and IP Setup (Bonding Slaves)
 Dynamic Address: DHCP DHCP both version 4 and 6
 Statically Assigned IP Address

IP Address: Subnet Mask: 255.255.255.0 Hostname:

| IPv4 Address Label | IP Address | Netmask |
|--------------------|------------|---------|
| | | |

Add Edit Delete

Help Cancel Back 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 Slaves* 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.

13.8.1 Hotplugging of Bonding Slaves

In specific network environments (such as High Availability), there are cases when you need to replace a bonding slave interface with another one. The reason may be a constantly failing network device. The solution is to set up hotplugging of bonding slaves.

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 slaves 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 slave interface is controlled by the bond master.

[STARTMODE=hotplug](#) causes the slave 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 slaves.

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 slaves, but for the bond to become ready, which requires at least one available slave. When one of the slave interfaces gets removed (unbind from NIC driver, `rmmmod` 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 slave, and calls `ifup` for it. The `ifup` call automatically joins it into the bond.

13.9 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 truncating”, 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 13.8, “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 13.5, “Feature Comparison between Bonding and Team”](#).

TABLE 13.5: 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 |

| Feature | Bonding | Team |
|---|---------|------|
| 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: http://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 13.1: GENERAL PROCEDURE

1. Make sure you have all the necessary packages installed. Install the packages `libteam-tools`, `libteamctl0`, and `python-libteam`.

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:

```
tux > sudo wicked show-config
```

5. Start the Network Teaming device `team0`:

```
tux > sudo wicked ifup all 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:

```
tux > sudo wicked ifstatus --verbose team0
```

- Get the state of the entire instance:

```
tux > sudo teamdctl team0 state
```

- Get the systemd state of the teamd instance:

```
tux > 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:

```
tux > 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 13.2: REMOVING A TEAM DEVICE

1. Stop the Network Teaming device `team0`:

```
tux > 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:

```
tux > sudo wicked ifreload all
```

13.9.1 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 13.1, “General Procedure”](#) to set up the device. Check the output with `teamdctl`.

EXAMPLE 13.12: CONFIGURATION FOR LOAD BALANCING WITH NETWORK TEAMING

```
STARTMODE=auto ①
BOOTPROTO=static ②
IPADDRESS="192.168.1.1/24" ②
IPADDR6="fd00:deca:fbad:50::1/64" ②

TEAM_RUNNER="loadbalance" ③
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" 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 `192.168.1.1` 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`.
- ③ Sets `TEAM_RUNNER` to `loadbalance` to activate the load balancing 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). 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.

13.9.2 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 13.1, "General Procedure"* to set up the device. Check the output with `teamctl`.

EXAMPLE 13.13: 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.

13.9.3 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 13.9.1, "Use Case: Load Balancing with Network Teaming"](#) or [Section 13.9.2, "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-vlanVLAN_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:

```
root # wicked ifup vlan0 vlan1
```

6. Check the output of `ifconfig`:

```
root # 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)
```

13.10 Software-Defined Networking with Open vSwitch

Software-defined networking (SDN) means separating the system that controls where traffic is sent (the *control plane*) from the underlying system that forwards traffic to the selected destination (the *data plane*, also called the *forwarding plane*). This means that the functions previously fulfilled by a single, usually inflexible switch can now be separated between a switch (data plane) and its controller (control plane). In this model, the controller is programmable and can be very flexible and adapt quickly to changing network conditions.

Open vSwitch is software that implements a distributed virtual multilayer switch that is compatible with the OpenFlow protocol. OpenFlow allows a controller application to modify the configuration of a switch. OpenFlow is layered onto the TCP protocol and is implemented in a range of hardware and software. A single controller can thus drive multiple, very different switches.

13.10.1 Advantages of Open vSwitch

Software-defined networking with Open vSwitch brings several advantages with it, especially when you used together with virtual machines:

- Networking states can be identified easily.
- Networks and their live state can be moved from one host to another.
- Network dynamics are traceable and external software can be enabled to respond to them.
- You can apply and manipulate tags in network packets to identify which machine they are coming from or going to and maintain other networking context. Tagging rules can be configured and migrated.
- Open vSwitch implements the GRE protocol (*Generic Routing Encapsulation*). This allows you, for example, to connect private VM networks to each other.
- Open vSwitch can be used on its own, but is designed to integrate with networking hardware and can control hardware switches.

13.10.2 Installing Open vSwitch

1. Install Open vSwitch and supplementary packages:

```
root # zypper install openvswitch openvswitch-switch
```

If you plan to use Open vSwitch together with the KVM hypervisor, additionally install `tunctl` . If you plan to use Open vSwitch together with the Xen hypervisor, additionally install `openvswitch-kmp-xen` .

2. Enable the Open vSwitch service:

```
root # systemctl enable openvswitch
```

3. Either restart the computer or use `systemctl` to start the Open vSwitch service immediately:

```
root # systemctl start openvswitch
```

4. To check whether Open vSwitch was activated correctly, use:

```
root # systemctl status openvswitch
```

13.10.3 Overview of Open vSwitch Daemons and Utilities

Open vSwitch consists of several components. Among them are a kernel module and various user space components. The kernel module is used for accelerating the data path, but is not necessary for a minimal Open vSwitch installation.

13.10.3.1 Daemons

The central executables of Open vSwitch are its two daemons. When you start the `openvswitch` service, you are indirectly starting them.

The main Open vSwitch daemon (`ovs-vswitchd`) provides the implementation of a switch. The Open vSwitch database daemon (`ovsdb-server`) serves the database that stores the configuration and state of Open vSwitch.

13.10.3.2 Utilities

Open vSwitch also comes with several utilities that help you work with it. The following list is not exhaustive, but instead describes important commands only.

ovsdb-tool

Create, upgrade, compact, and query Open vSwitch databases. Do transactions on Open vSwitch databases.

ovs-appctl

Configure a running `ovs-vswitchd` or `ovsdb-server` daemon.

ovs-dpctl, ovs-dpctl-top

Create, modify, visualize, and delete data paths. Using this tool can interfere with `ovs-vswitchd` also performing data path management. Therefore, it is often used for diagnostics only.

`ovs-dpctl-top` creates a `top`-like visualization for data paths.

ovs-ofctl

Manage any switches adhering to the OpenFlow protocol. `ovs-ofctl` is not limited to interacting with Open vSwitch.

ovs-vsctl

Provides a high-level interface to the configuration database. It can be used to query and modify the database. In effect, it shows the status of `ovs-vsitchd` and can be used to configure it.

13.10.4 Creating a Bridge with Open vSwitch

The following example configuration uses the Wicked network service that is used by default on openSUSE Leap. To learn more about Wicked, see [Section 13.6, “Configuring a Network Connection Manually”](#).

When you have installed and started Open vSwitch, proceed as follows:

1. To configure a bridge for use by your virtual machine, create a file with content like this:

```
STARTMODE='auto' ❶  
BOOTPROTO='dhcp' ❷  
OVS_BRIDGE='yes' ❸  
OVS_BRIDGE_PORT_DEVICE_1='eth0' ❹
```

- ❶ Set up the bridge automatically when the network service is started.
- ❷ The protocol to use for configuring the IP address.
- ❸ Mark the configuration as an Open vSwitch bridge.
- ❹ Choose which device/devices should be added to the bridge. To add more devices, append additional lines for each of them to the file:

```
OVS_BRIDGE_PORT_DEVICE_SUFFIX='DEVICE'
```

The SUFFIX can be any alphanumeric string. However, to avoid overwriting a previous definition, make sure the SUFFIX of each device is unique.

Save the file in the directory `/etc/sysconfig/network` under the name `ifcfg-br0`. Instead of `br0`, you can use any name you want. However, the file name needs to begin with `ifcfg-`.

To learn about further options, refer to the man pages of `ifcfg` ([man 5 ifcfg](#)) and `ifcfg-ovs-bridge` ([man 5 ifcfg-ovs-bridge](#)).

2. Now start the bridge:

```
root # wicked ifup br0
```

When Wicked is done, it should output the name of the bridge and next to it the state up.

13.10.5 Using Open vSwitch Directly with KVM

After having created the bridge as described in *Section 13.10.4, "Creating a Bridge with Open vSwitch"*, you can use Open vSwitch to manage the network access of virtual machines created with KVM/QEMU.

1. To be able to best use the capabilities of Wicked, make some further changes to the bridge configured before. Open the previously created `/etc/sysconfig/network/ifcfg-br0` and append a line for another port device:

```
OVS_BRIDGE_PORT_DEVICE_2='tap0'
```

Additionally, set `BOOTPROTO` to `none`. The file should now look like this:

```
STARTMODE='auto'  
BOOTPROTO='none'  
OVS_BRIDGE='yes'  
OVS_BRIDGE_PORT_DEVICE_1='eth0'  
OVS_BRIDGE_PORT_DEVICE_2='tap0'
```

The new port device `tap0` will be configured in the next step.

2. Now add a configuration file for the `tap0` device:

```
STARTMODE='auto'  
BOOTPROTO='none'  
TUNNEL='tap'
```

Save the file in the directory `/etc/sysconfig/network` under the name `ifcfg-tap0`.



Tip: Allowing Other Users to Access the Tap Device

To be able to use this tap device from a virtual machine started as a user who is not `root`, append:

```
TUNNEL_SET_OWNER=USER_NAME
```

To allow access for an entire group, append:

```
TUNNEL_SET_GROUP=GROUP_NAME
```

3. Finally, open the configuration for the device defined as the first `OVS_BRIDGE_PORT_DEVICE`. If you did not change the name, that should be `eth0`. Therefore, open `/etc/sysconfig/network/ifcfg-eth0` and make sure that the following options are set:

```
STARTMODE='auto'  
BOOTPROTO='none'
```

If the file does not exist yet, create it.

4. Restart the bridge interface using Wicked:

```
root # wicked ifreload br0
```

This will also trigger a reload of the newly defined bridge port devices.

5. To start a virtual machine, use, for example:

```
root # qemu-kvm \  
-drive file=/PATH/TO/DISK-IMAGE ❶ \  
-m 512 -net nic,vlan=0,macaddr=00:11:22:EE:EE:EE \  
-net tap,ifname=tap0,script=no,downscript=no ❷
```

❶ The path to the QEMU disk image you want to start.

❷ Use the tap device (`tap0`) created before.

For further information on the usage of KVM/QEMU, see *Book "Virtualization Guide"*.

13.10.6 Using Open vSwitch with libvirt

After having created the bridge as described before in *Section 13.10.4, "Creating a Bridge with Open vSwitch"*, you can add the bridge to an existing virtual machine managed with `libvirt`. Since `libvirt` has some support for Open vSwitch bridges already, you can use the bridge created in *Section 13.10.4, "Creating a Bridge with Open vSwitch"* without further changes to the networking configuration.

1. Open the domain XML file for the intended virtual machine:

```
root # virsh edit VM_NAME
```

Replace *VM_NAME* with the name of the desired virtual machine. This will open your default text editor.

2. Find the networking section of the document by looking for a section starting with `<interface type="...">` and ending in `</interface>`.

Replace the existing section with a networking section that looks somewhat like this:

```
<interface type='bridge'>
  <source bridge='br0' />
  <virtualport type='openvswitch' />
</interface>
```

Important: Compatibility of **virsh iface-*** and Virtual Machine Manager with Open vSwitch

At the moment, the Open vSwitch compatibility of libvirt is not exposed through the virsh iface-* tools and Virtual Machine Manager. If you use any of these tools, your configuration can break.

3. You can now start or restart the virtual machine as usual.

For further information on the usage of libvirt, see *Book "Virtualization Guide"*.

13.10.7 For More Information

<http://openvswitch.org/support/> ↗

The documentation section of the Open vSwitch project Web site

<https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf> ↗

Whitepaper by the Open Networking Foundation about software-defined networking and the OpenFlow protocol

14 UEFI (Unified Extensible Firmware Interface)

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 http://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 some features are implemented in openSUSE Leap.

14.1 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 openSUSE Leap, 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, some driver located in the flash memory of some 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, as long as they are signed with the private part of the PK.

14.1.1 Implementation on openSUSE Leap

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.

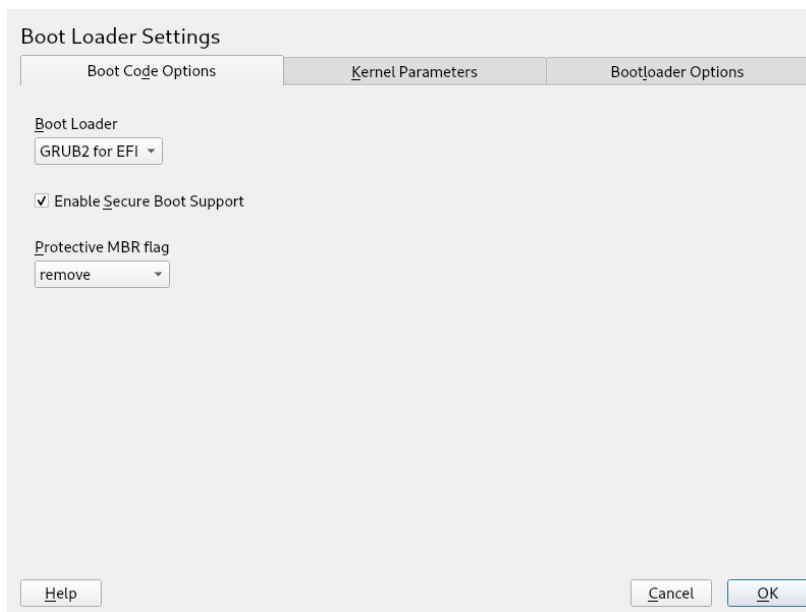


FIGURE 14.1: SECURE BOOT SUPPORT

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).

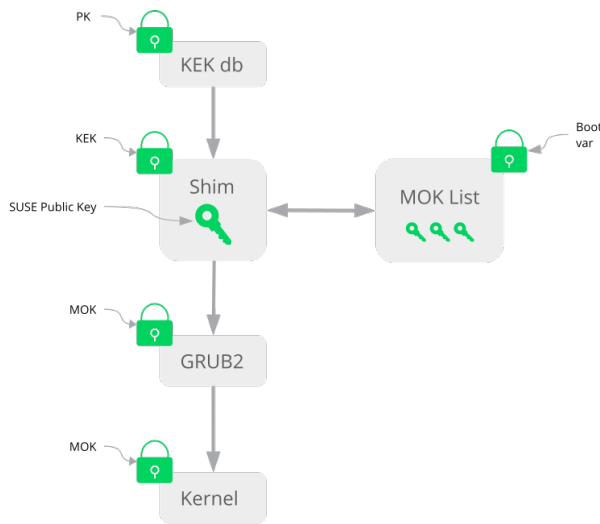


FIGURE 14.2: UEFI: SECURE BOOT PROCESS

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.

The various 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.

14.1.2 MOK (Machine Owner Key)

To replace specific kernels, drivers, or other components that are part of the boot process, you have 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` bootloader 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 usually 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.

14.1.3 Booting a Custom Kernel

The following is based on http://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 http://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:

```
tux > openssl pkcs12 -export -inkey key.asc -in cert.pem \  
-name kernel_cert -out cert.p12
```

3. Generate an NSS database for use with `pesign`:

```
tux > certutil -d . -N
```

4. Import the key and the certificate contained in PKCS#12 into the NSS database:

```
tux > pk12util -d . -i cert.p12
```

5. “Bless” the kernel with the new signature using `pesign`:

```
tux > pesign -n . -c kernel_cert -i arch/x86/boot/bzImage \  
-o vmlinuz.signed -s
```

6. List the signatures on the kernel image:

```
tux > 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:

```
tux > openssl x509 -in cert.pem -outform der -out cert.der
```

8. Copy the certificate to the ESP for easier access:

```
tux > sudo cp cert.der /boot/efi/
```

9. Use `mokutil` to launch the MOK list automatically.

- a. Import the certificate to MOK:

```
tux > 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:

```
tux > 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:

```
tux > mokutil --list-enrolled
```

- a. Alternatively, this is the procedure if you want 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.

14.1.4 Using Non-Inbox Drivers

There is no support for adding non-inbox drivers (that is, drivers that do not come with openSUSE Leap) 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.

14.1.5 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 videomode support, the kernel can retrieve video mode from UEFI to configure KMS mode with the same parameters.
- UEFI booting from USB devices is supported.

When booting in Secure Boot mode, the following limitations apply:

- To ensure that Secure Boot cannot be easily circumvented, some kernel features are disabled when running under Secure Boot.
- Boot loader, kernel, and kernel modules must be signed.
- Kexec and Kdump are disabled.
- 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.

14.2 For More Information

- <https://www.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.

15 Special System Features

This chapter starts with information about various 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).

15.1 Information about Special Software Packages

The following chapter provides basic information about the following tools: bash, cron, logrotate, locate, ulimit and free.

15.1.1 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:

```
tux > mv ~/.bashrc ~/.bashrc.old
```

```
tux > cp /etc/skel/.bashrc ~/.bashrc
tux > mv ~/.profile ~/.profile.old
tux > cp /etc/skel/.profile ~/.profile
```

Then copy personal adjustments back from the *.old files.

15.1.2 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 15.1, "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 15.1: 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 15.2, "/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 15.2: /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.

The daily system maintenance jobs are distributed to various scripts for reasons of clarity. They are contained in the package `aaa_base`. `/etc/cron.daily` contains, for example, the components `suse.de-backup-rpmdb`, `suse.de-clean-tmp` or `suse.de-cron-local`.

15.1.3 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.

15.1.4 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 "Analyzing and Managing System Log Files", Section 3.3 "Managing Log Files with logrotate"*.

15.1.5 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.

15.1.6 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 15.1, “`ulimit`: Setting Resources for the User”](#).

TABLE 15.1: `ulimit`: SETTING RESOURCES FOR THE USER

| | |
|-----------------|---|
| <code>-m</code> | The maximum resident set size |
| <code>-v</code> | The maximum amount of virtual memory available to the shell |
| <code>-s</code> | The maximum size of the stack |
| <code>-c</code> | The maximum size of core files created |
| <code>-a</code> | 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 15.3: `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`.

15.1.7 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 actually 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`.

15.1.8 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.

15.1.9 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`.

15.1.10 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 <http://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 openSUSE Leap, 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.

15.2 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`.

15.3 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: For More Information

Information about XKB is available in the documents listed in [/usr/share/doc/packages/xkeyboard-config](#) (part of the [xkeyboard-config](#) package).

15.4 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>]
```

15.4.1 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 openSUSE Leap openSUSE Leap

Earlier versions of openSUSE Leap read locale settings from `/etc/sysconfig/language`, `/etc/sysconfig/keyboard`, and `/etc/sysconfig/console`. Starting with openSUSE Leap 15.0, 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 15.4.2, "Some Examples"](#)).

To set the system-wide locale, you can either:

- Write your settings in `/etc/locale.conf`. Each line is an 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`:

```
root # 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.

15.4.2 Some Examples

You should always set the language and country codes together. Language settings follow the standard ISO 639 available at <http://www.evertype.com/standards/iso639/iso639-en.html> and <http://www.loc.gov/standards/iso639-2/>. Country codes are listed in ISO 3166, see http://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.

15.4.3 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
```

15.4.4 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"
```

Note that 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.

15.4.5 For 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 <http://www.cl.cam.ac.uk/~mgk25/unicode.html> .

16 Dynamic Kernel Device Management with udev

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.

16.1 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.

16.2 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 16.6, “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.

16.3 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. Usually 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`.

16.4 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.

16.5 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`.

16.6 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 16.1: 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/usb/lp0`.

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.

Some 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.

16.6.1 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.

16.6.2 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.

16.6.3 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}=="0n[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.

16.6.4 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 very 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 completely.
- 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.

16.7 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

```

16.8 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.

16.9 For 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.

III Services

- 17 SLP [326](#)
- 18 Time Synchronization with NTP [330](#)
- 19 The Domain Name System [337](#)
- 20 DHCP [362](#)
- 21 Samba [377](#)
- 22 Sharing File Systems with NFS [401](#)
- 23 On-Demand Mounting with Autofs [412](#)
- 24 The Apache HTTP Server [420](#)
- 25 Setting Up an FTP Server with YaST [462](#)
- 26 Squid Caching Proxy Server [466](#)

17 SLP

Configuring a network client requires detailed knowledge about services provided over the network (such as printing or LDAP, for example). To make it easier to configure such services on a network client, the “service location protocol” (SLP) was developed. SLP makes the availability and configuration data of selected services known to all clients in the local network. Applications that support SLP can use this information to be configured automatically.

openSUSE® Leap supports installation using installation sources provided with SLP and contains many system services with integrated support for SLP. You can use SLP to provide networked clients with central functions, such as an installation server, file server, or print server on your system. Services that offer SLP support include cupsd, login, ntp, openldap2, postfix, rpasswd, rsyncd, saned, sshd (via fish), vnc, and ypserv.

All packages necessary to use SLP services on a network client are installed by default. However, if you want to *provide* services via SLP, check that the `openslp-server` package is installed.

17.1 The SLP Front-End `slptool`

`slptool` is a command line tool to query and register SLP services. The query functions are useful for diagnostic purposes. The most important `slptool` subcommands are listed below. `slptool --help` lists all available options and functions.

`findsrvtypes`

List all service types available on the network.

```
tux > slptool findsrvtypes
service:install.suse:nfs
service:install.suse:ftp
service:install.suse:http
service:install.suse:smb
service:ssh
service:fish
service:YaST.installation.suse:vnc
service:smtp
service:domain
service:management-software.IBM:hardware-management-console
service:rsync
```

```
service:ntp
service:yppserv
```

findsrvs SERVICE_TYPE

List all servers providing SERVICE_TYPE

```
tux > slptool findsrvs service:ntp
service:ntp://ntp.example.com:123,57810
service:ntp://ntp2.example.com:123,57810
```

findattrs SERVICE_TYPE // HOST

List attributes for SERVICE_TYPE on HOST

```
tux > slptool findattrs service:ntp://ntp.example.com
(owner=tux),(email=tux@example.com)
```

register SERVICE type // HOST:PORT "(ATTRIBUTE=VALUE),(ATTRIBUTE=VALUE)"

Registers SERVICE_TYPE on HOST with an optional list of attributes

```
slptool register service:ntp://ntp.example.com:57810 \
"(owner=tux),(email=tux@example.com)"
```

deregister SERVICE_TYPE // host

De-registers SERVICE_TYPE on HOST

```
slptool deregister service:ntp://ntp.example.com
```

For more information run **slptool --help**.

17.2 Providing Services via SLP

To provide SLP services, the SLP daemon (slpd) must be running. Like most system services in openSUSE Leap, slpd is controlled by means of a separate start script. After the installation, the daemon is inactive by default. To activate it for the current session, run **sudo systemctl start slpd**. If slpd should be activated on system start-up, run **sudo systemctl enable slpd**.

Many applications in openSUSE Leap have integrated SLP support via the libslp library. If a service has not been compiled with SLP support, use one of the following methods to make it available via SLP:

Static Registration with /etc/slp.reg.d

Create a separate registration file for each new service. The following example registers a scanner service:

```
## Register a saned service on this system
## en means english language
## 65535 disables the timeout, so the service registration does
## not need refreshes
service:scanner.sane://$HOSTNAME:6566,en,65535
watch-port-tcp=6566
description=SANE scanner daemon
```

The most important line in this file is the *service URL*, which begins with `service: .` This contains the service type (`scanner.sane`) and the address under which the service is available on the server. `$HOSTNAME` is automatically replaced with the full host name. The name of the TCP port on which the relevant service can be found follows, separated by a colon. Then enter the language in which the service should appear and the duration of registration in seconds. These should be separated from the service URL by commas. Set the value for the duration of registration between `0` and `65535`. `0` prevents registration. `65535` removes all restrictions.

The registration file also contains the two variables `watch-port-tcp` and `description`. `watch-port-tcp` links the SLP service announcement to whether the relevant service is active by having `slpd` check the status of the service. The second variable contains a more precise description of the service that is displayed in suitable browsers.



Tip: YaST and SLP

Some services brokered by YaST, such as an installation server or YOU server, perform this registration automatically when you activate SLP in the module dialogs. YaST then creates registration files for these services.

Static Registration with `/etc/slp.reg`

The only difference between this method and the procedure with `/etc/slp.reg.d` is that all services are grouped within a central file.

Dynamic Registration with `slptool`

If a service needs to be registered dynamically without the need of configuration files, use the `slptool` command line utility. The same utility can also be used to de-register an existing service offering without restarting `slpd`. See [Section 17.1, “The SLP Front-End `slptool`”](#) for details.

17.2.1 Setting up an SLP Installation Server

Announcing the installation data via SLP within your network makes the network installation much easier, since the installation data such as IP address of the server or the path to the installation media are automatically required via SLP query.

17.3 For More Information

RFC 2608, 2609, 2610

RFC 2608 generally deals with the definition of SLP. RFC 2609 deals with the syntax of the service URLs used in greater detail and RFC 2610 deals with DHCP via SLP.

<http://www.openslp.org> ↗

The home page of the OpenSLP project.

</usr/share/doc/packages/openslp>

This directory contains the documentation for SLP coming with the [openslp-server](#) package, including a [README.SUSE](#) containing the openSUSE Leap details, the RFCs, and two introductory HTML documents. Programmers who want to use the SLP functions will find more information in the *Programmers Guide* that is included in the [openslp-devel](#) package.

18 Time Synchronization with NTP

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 openSUSE Leap 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 various operating parameters at runtime.

Starting with openSUSE Leap 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 by means of active directory, follow the instructions found at *Book “Security and Hardening Guide”, Chapter 8 “Active Directory Support”, Section 8.3.3 “Joining Active Directory Using Windows Domain Membership”, Joining an Active Directory Domain Using Windows Domain Membership.*

18.1 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 a hardware clock heavily depends on its time source. For example, an atomic clock or GPS receiver is a very 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.

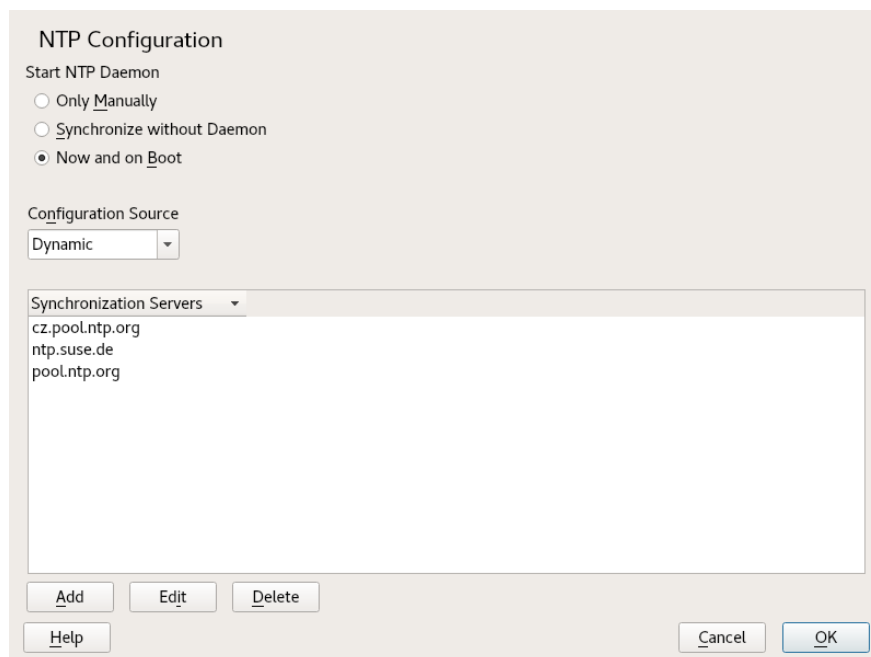


FIGURE 18.1: NTP CONFIGURATION WINDOW

18.1.1 NTP Daemon Start

You can choose from three options for when to start the NTP daemon:

Only Manually

Select *Only Manually*, if you want 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.

18.1.2 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.

18.1.3 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:

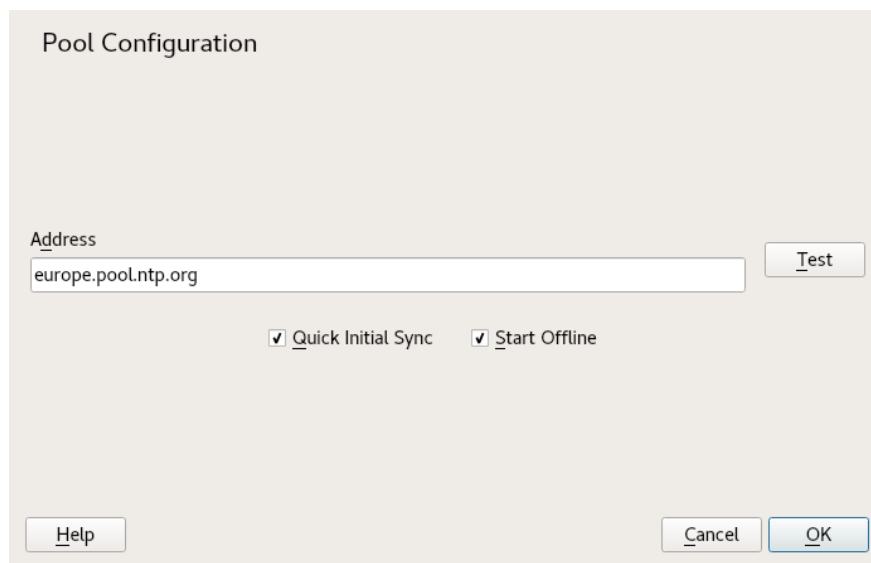


FIGURE 18.2: ADDING A TIME SERVER

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 whose network connection is managed by NetworkManager.
4. Confirm with *OK*.

18.2 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:

```
server 0.europe.pool.ntp.org
server 1.europe.pool.ntp.org
server 2.europe.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 all of them 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 some time 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
```

18.3 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:

```
root # 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:

```
root # 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 manual page (`man 1 chronyc`).

18.4 Dynamic Time Synchronization at Runtime

If the system boots without network connection, `chronyd` starts up, but it cannot resolve DNS names of the time servers set in the configuration file. This can happen if you use NetworkManager with an encrypted Wi-Fi.

`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` will then not try to poll the server until it is enabled using the following command:

```
root # 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.

18.5 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](#).

19 The Domain Name System

DNS (domain name system) is needed to resolve the domain names and host names into IP addresses. In this way, the IP address 192.168.2.100 is assigned to the host name jupiter, for example. Before setting up your own name server, read the general information about DNS in *Section 13.3, "Name Resolution"*. The following configuration examples refer to BIND, the default DNS server.

19.1 DNS Terminology

Zone

The domain name space is divided into regions called zones. For example, if you have example.com, you have the example section (or zone) of the com domain.

DNS server

The DNS server is a server that maintains the name and IP information for a domain. You can have a primary DNS server for master zone, a secondary server for slave zone, or a slave server without any zones for caching.

Master zone DNS server

The master zone includes all hosts from your network and a DNS server master zone stores up-to-date records for all the hosts in your domain.

Slave zone DNS server

A slave zone is a copy of the master zone. The slave zone DNS server obtains its zone data with zone transfer operations from its master server. The slave zone DNS server responds authoritatively for the zone as long as it has valid (not expired) zone data. If the slave cannot obtain a new copy of the zone data, it stops responding for the zone.

Forwarder

Forwarders are DNS servers to which your DNS server should send queries it cannot answer. To enable different configuration sources in one configuration, netconfig is used (see also man 8 netconfig).

Record

The record is information about name and IP address. Supported records and their syntax are described in BIND documentation. Some special records are:

NS record

An NS record tells name servers which machines are in charge of a given domain zone.

MX record

The MX (mail exchange) records describe the machines to contact for directing mail across the Internet.

SOA record

SOA (Start of Authority) record is the first record in a zone file. The SOA record is used when using DNS to synchronize data between multiple computers.

19.2 Installation

To install a DNS server, start YaST and select *Software > Software Management*. Choose *View > Patterns* and select *DHCP and DNS Server*. Confirm the installation of the dependent packages to finish the installation process.

Alternatively use the following command on the command line:

```
tux > sudo zypper in -t pattern dhcp_dns_server
```

19.3 Configuration with YaST

Use the YaST DNS module to configure a DNS server for the local network. When starting the module for the first time, a wizard starts, prompting you to make a few decisions concerning administration of the server. Completing this initial setup produces a basic server configuration. Use the expert mode to deal with more advanced configuration tasks, such as setting up ACLs, logging, TSIG keys, and other options.

19.3.1 Wizard Configuration

The wizard consists of three steps or dialogs. At the appropriate places in the dialogs, you can enter the expert configuration mode.

1. When starting the module for the first time, the *Forwarder Settings* dialog, shown in *Figure 19.1, “DNS Server Installation: Forwarder Settings”*, opens. The *Local DNS Resolution Policy* allows to set the following options:

- *Merging forwarders is disabled*
- *Automatic merging*
- *Merging forwarders is enabled*
- *Custom configuration*—If *Custom configuration* is selected, *Custom policy* can be specified; by default (with *Automatic merging* selected), *Custom policy* is set to auto, but here you can either set interface names or select from the two special policy names STATIC and STATIC_FALLBACK.

In *Local DNS Resolution Forwarder*, specify which service to use: *Using system name servers*, *This name server (bind)*, or *Local dnsmasq server*.

For more information about all these settings, see man 8 netconfig.

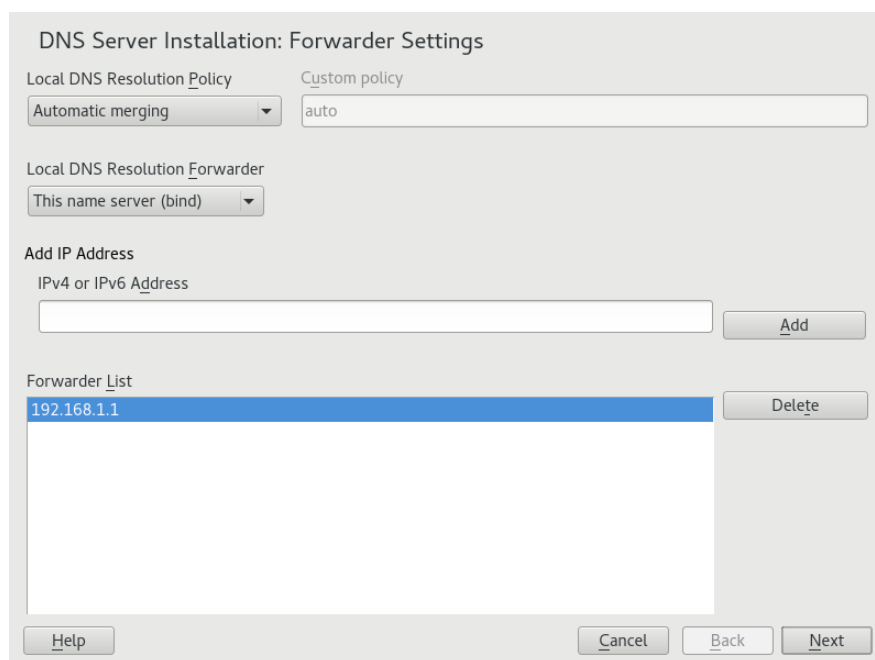


FIGURE 19.1: DNS SERVER INSTALLATION: FORWARDER SETTINGS

Forwarders are DNS servers to which your DNS server sends queries it cannot answer itself. Enter their IP address and click *Add*.

- The *DNS Zones* dialog consists of several parts and is responsible for the management of zone files, described in [Section 19.6, “Zone Files”](#). For a new zone, provide a name for it in *Name*. To add a reverse zone, the name must end in `.in-addr.arpa`. Finally, select the *Type* (master, slave, or forward). See [Figure 19.2, “DNS Server Installation: DNS Zones”](#). Click *Edit* to configure other settings of an existing zone. To remove a zone, click *Delete*.

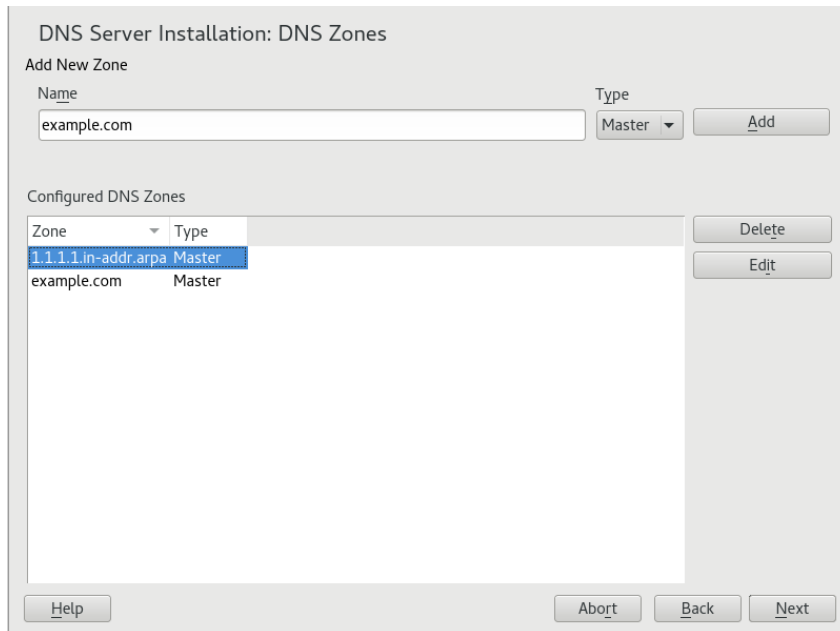


FIGURE 19.2: DNS SERVER INSTALLATION: DNS ZONES

- In the final dialog, you can open the DNS port in the firewall by clicking *Open Port in Firewall*. Then decide whether to start the DNS server when booting (*On* or *Off*). You can also activate LDAP support. See [Figure 19.3, “DNS Server Installation: Finish Wizard”](#).

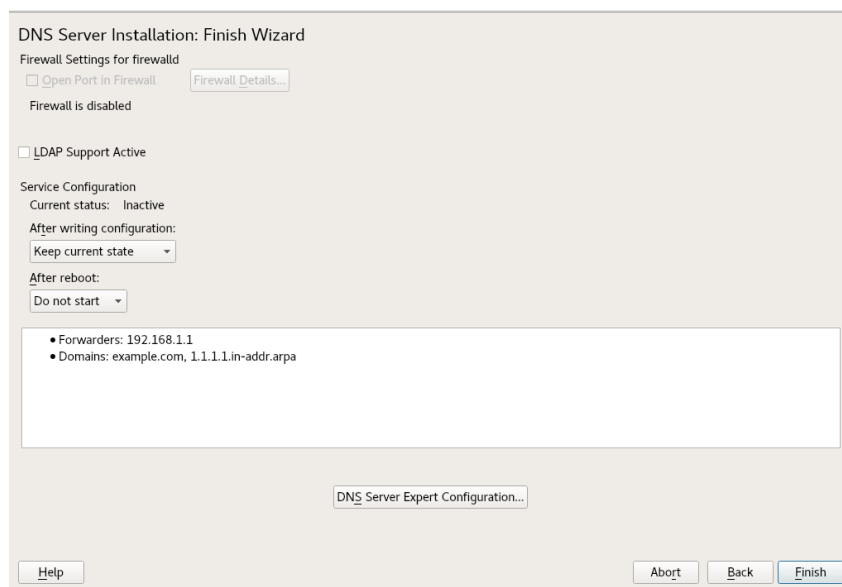


FIGURE 19.3: DNS SERVER INSTALLATION: FINISH WIZARD

19.3.2 Expert Configuration

After starting the module, YaST opens a window displaying several configuration options. Completing it results in a DNS server configuration with the basic functions in place:

19.3.2.1 Start-Up

Under *Start-Up*, define whether the DNS server should be started when booting the system or manually. To start the DNS server immediately, click *Start DNS Server Now*. To stop the DNS server, click *Stop DNS Server Now*. To save the current settings, select *Save Settings and Reload DNS Server Now*. You can open the DNS port in the firewall with *Open Port in Firewall* and modify the firewall settings with *Firewall Details*.

By selecting *LDAP Support Active*, the zone files are managed by an LDAP database. Any changes to zone data written to the LDAP database are picked up by the DNS server when it is restarted or prompted to reload its configuration.

19.3.2.2 Forwarders

If your local DNS server cannot answer a request, it tries to forward the request to a *Forwarder*, if configured so. This forwarder may be added manually to the *Forwarder List*. If the forwarder is not static like in dial-up connections, *netconfig* handles the configuration. For more information about *netconfig*, see [man 8 netconfig](#).

19.3.2.3 Basic Options

In this section, set basic server options. From the *Option* menu, select the desired item then specify the value in the corresponding text box. Include the new entry by selecting *Add*.

19.3.2.4 Logging

To set what the DNS server should log and how, select *Logging*. Under *Log Type*, specify where the DNS server should write the log data. Use the system-wide log by selecting *System Log* or specify a different file by selecting *File*. In the latter case, additionally specify a name, the maximum file size in megabytes and the number of log file versions to store.

Further options are available under *Additional Logging*. Enabling *Log All DNS Queries* causes every query to be logged, in which case the log file could grow extremely large. For this reason, it is not a good idea to enable this option for other than debugging purposes. To log the data traffic during zone updates between DHCP and DNS server, enable *Log Zone Updates*. To log the data traffic during a zone transfer from master to slave, enable *Log Zone Transfer*. See [Figure 19.4](#), “DNS Server: Logging”.

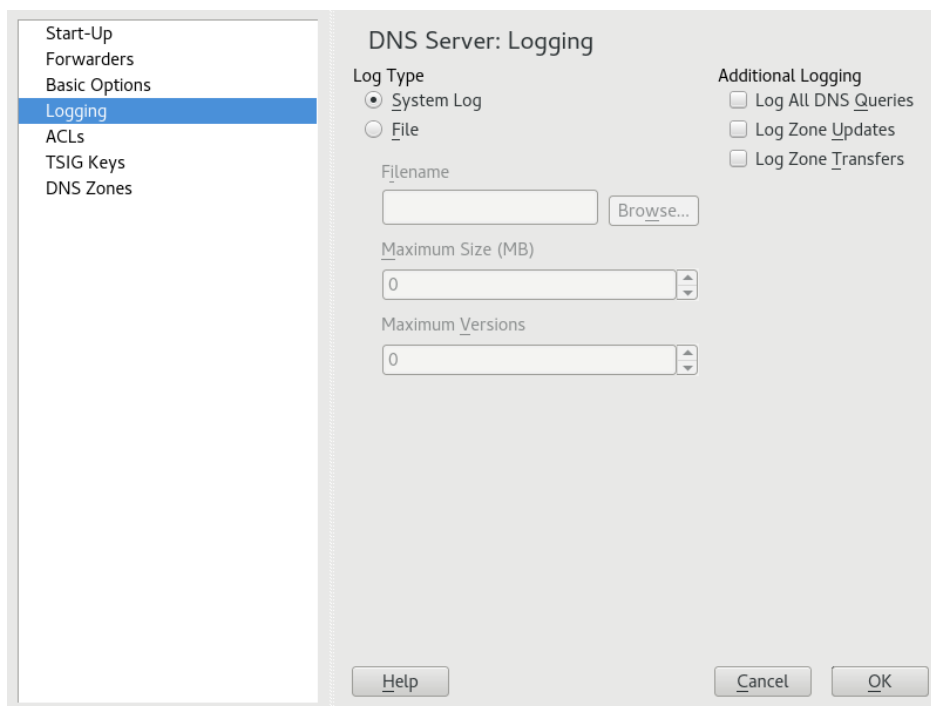


FIGURE 19.4: DNS SERVER: LOGGING

19.3.2.5 ACLs

Use this dialog to define ACLs (access control lists) to enforce access restrictions. After providing a distinct name under *Name*, specify an IP address (with or without netmask) under *Value* in the following fashion:

```
{ 192.168.1/24; }
```

The syntax of the configuration file requires that the address ends with a semicolon and is put into curly braces.

19.3.2.6 TSIG Keys

The main purpose of TSIGs (transaction signatures) is to secure communications between DHCP and DNS servers. They are described in [Section 19.8, "Secure Transactions"](#).

To generate a TSIG key, enter a distinctive name in the field labeled *Key ID* and specify the file where the key should be stored (*Filename*). Confirm your choices with *Generate*.

To use a previously created key, leave the *Key ID* field blank and select the file where it is stored under *Filename*. After that, confirm with *Add*.

19.3.2.7 DNS Zones (Adding a Slave Zone)

To add a slave zone, select *DNS Zones*, choose the zone type *Slave*, write the name of the new zone, and click *Add*.

In the *Zone Editor* sub-dialog under *Master DNS Server IP*, specify the master from which the slave should pull its data. To limit access to the server, select one of the ACLs from the list.

19.3.2.8 DNS Zones (Adding a Master Zone)

To add a master zone, select *DNS Zones*, choose the zone type *Master*, write the name of the new zone, and click *Add*. When adding a master zone, a reverse zone is also needed. For example, when adding the zone example.com that points to hosts in a subnet 192.168.1.0/24, you should also add a reverse zone for the IP-address range covered. By definition, this should be named 1.168.192.in-addr.arpa.

19.3.2.9 DNS Zones (Editing a Master Zone)

To edit a master zone, select *DNS Zones*, select the master zone from the table, and click *Edit*. The dialog consists of several pages: *Basics* (the one opened first), *NS Records*, *MX Records*, *SOA*, and *Records*.

The basic dialog, shown in *Figure 19.5, "DNS Server: Zone Editor (Basics)"*, lets you define settings for dynamic DNS and access options for zone transfers to clients and slave name servers. To permit the dynamic updating of zones, select *Allow Dynamic Updates* and the corresponding TSIG key. The key must have been defined before the update action starts. To enable zone transfers, select the corresponding ACLs. ACLs must have been defined already.

In the *Basics* dialog, select whether to enable zone transfers. Use the listed ACLs to define who can download zones.

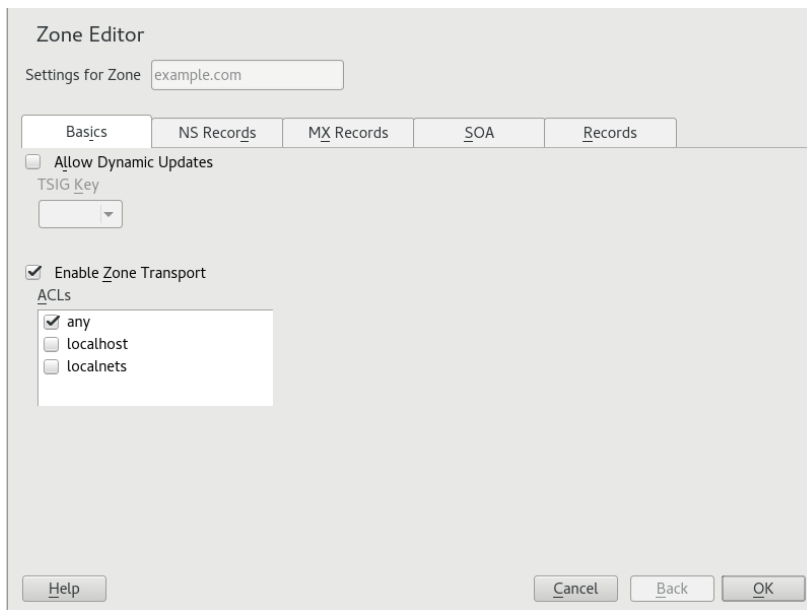


FIGURE 19.5: DNS SERVER: ZONE EDITOR (BASICS)

Zone Editor (NS Records)

The *NS Records* dialog allows you to define alternative name servers for the zones specified. Make sure that your own name server is included in the list. To add a record, enter its name under *Name Server to Add* then confirm with *Add*. See [Figure 19.6, “DNS Server: Zone Editor \(NS Records\)”](#).

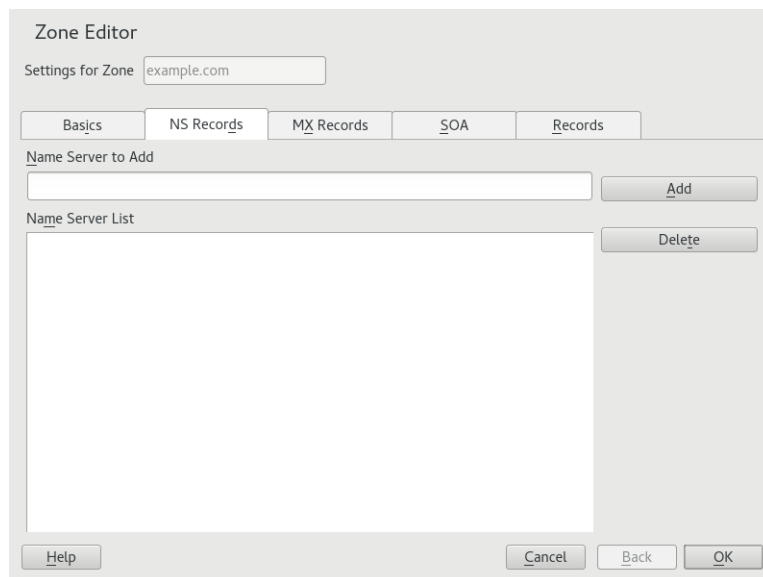


FIGURE 19.6: DNS SERVER: ZONE EDITOR (NS RECORDS)

Zone Editor (MX Records)

To add a mail server for the current zone to the existing list, enter the corresponding address and priority value. After doing so, confirm by selecting *Add*. See *Figure 19.7, “DNS Server: Zone Editor (MX Records)”*.

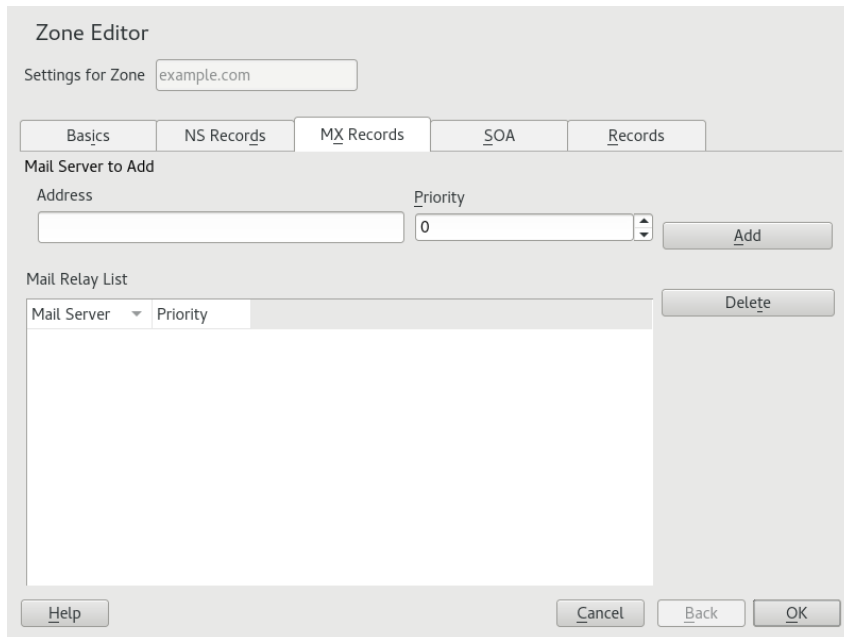


FIGURE 19.7: DNS SERVER: ZONE EDITOR (MX RECORDS)

Zone Editor (SOA)

This page allows you to create SOA (start of authority) records. For an explanation of the individual options, refer to *Example 19.6, “The `/var/lib/named/example.com.zone` File”*. Changing SOA records is not supported for dynamic zones managed via LDAP.

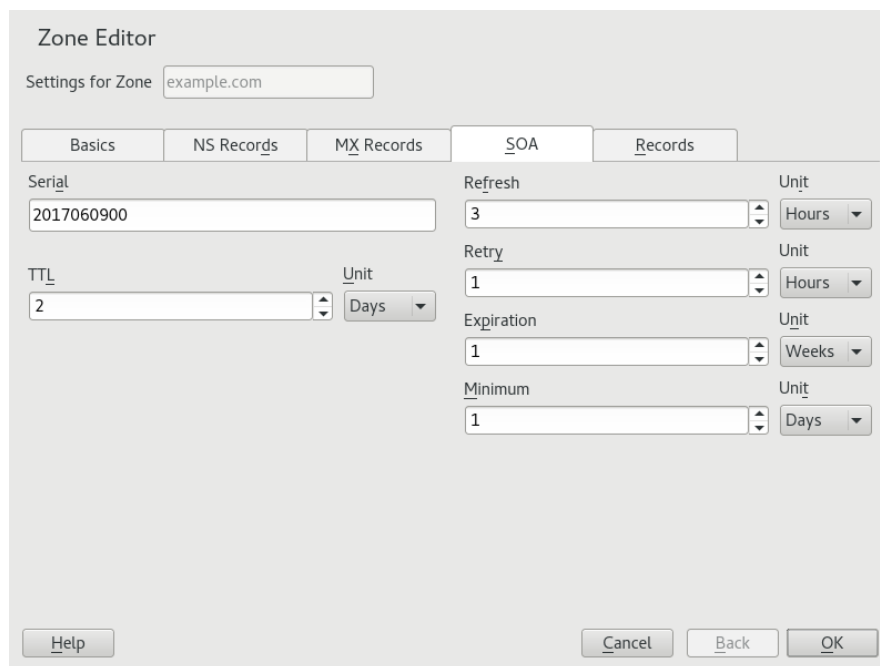


FIGURE 19.8: DNS SERVER: ZONE EDITOR (SOA)

Zone Editor (Records)

This dialog manages name resolution. In *Record Key*, enter the host name then select its type. The *A* type represents the main entry. The value for this should be an IP address (IPv4). Use *AAAA* for IPv6 addresses. *CNAME* is an alias. Use the types *NS* and *MX* for detailed or partial records that expand on the information provided in the *NS Records* and *MX Records* tabs. These three types resolve to an existing *A* record. *PTR* is for reverse zones. It is the opposite of an *A* record, for example:

```
hostname.example.com. IN A 192.168.0.1
1.0.168.192.in-addr.arpa IN PTR hostname.example.com.
```

19.3.2.9.1 Adding Reverse Zones

To add a reverse zone, follow this procedure:

1. Start *YaST* > *DNS Server* > *DNS Zones*.
2. If you have not added a master forward zone, add it and *Edit* it.
3. In the *Records* tab, fill the corresponding *Record Key* and *Value*, then add the record with *Add* and confirm with *OK*. If *YaST* complains about a non-existing record for a name server, add it in the *NS Records* tab.

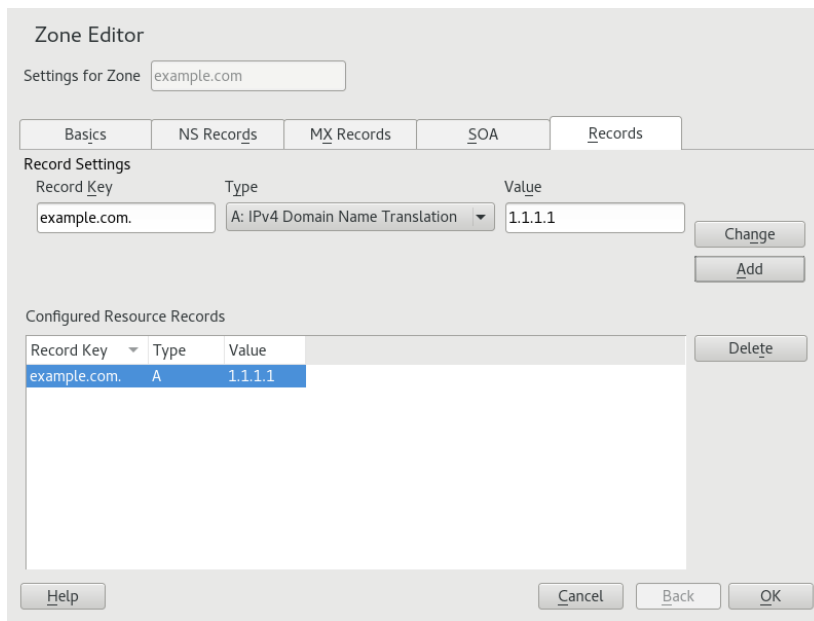


FIGURE 19.9: ADDING A RECORD FOR A MASTER ZONE

4. Back in the *DNS Zones* window, add a reverse master zone.

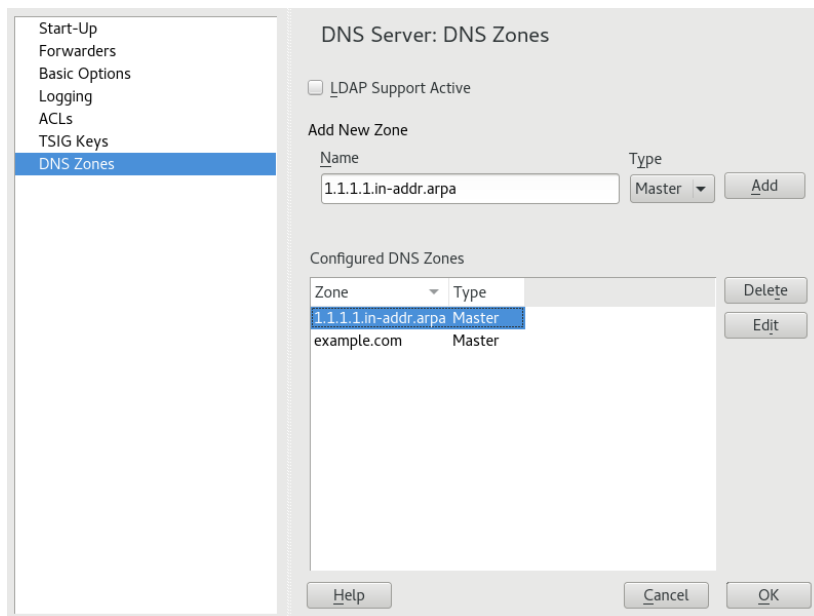


FIGURE 19.10: ADDING A REVERSE ZONE

5. *Edit* the reverse zone, and in the *Records* tab, you can see the *PTR: Reverse translation* record type. Add the corresponding *Record Key* and *Value*, then click *Add* and confirm with *OK*.

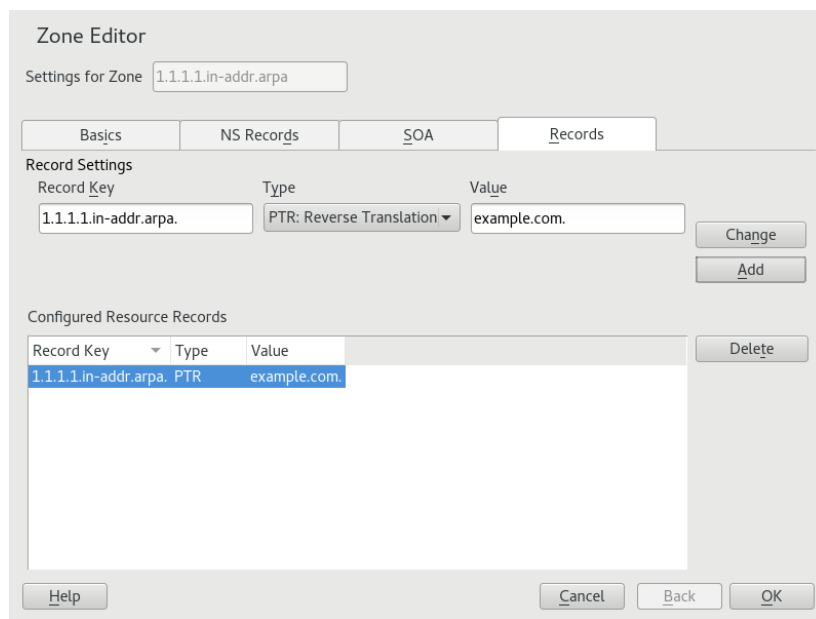


FIGURE 19.11: ADDING A REVERSE RECORD

Add a name server record if needed.



Tip: Editing the Reverse Zone

After adding a forward zone, go back to the main menu and select the reverse zone for editing. There in the tab *Basics* activate the check box *Automatically Generate Records From* and select your forward zone. That way, all changes to the forward zone are automatically updated in the reverse zone.

19.4 Starting the BIND Name Server

On a openSUSE® Leap system, the name server BIND (*Berkeley Internet Name Domain*) comes preconfigured, so it can be started right after installation without any problems. Normally, if you already have an Internet connection and entered `127.0.0.1` as the name server address for `localhost` in `/var/run/netconfig/resolv.conf`, you already have a working name resolution without needing to know the DNS of the provider. BIND carries out name resolution via the root name server, a notably slower process. Normally, the DNS of the provider should be entered with its IP address in the configuration file `/etc/named.conf` under `forwarders` to

ensure effective and secure name resolution. If this works so far, the name server runs as a pure *caching-only* name server. Only when you configure its own zones it becomes a proper DNS. Find a simple example documented in [/usr/share/doc/packages/bind/config](#).



Tip: Automatic Adaptation of the Name Server Information

Depending on the type of Internet connection or the network connection, the name server information can automatically be adapted to the current conditions. To do this, set the `NETCONFIG_DNS_POLICY` variable in the `/etc/sysconfig/network/config` file to `auto`.

However, do not set up an official domain until one is assigned to you by the responsible institution. Even if you have your own domain and it is managed by the provider, you are better off not using it, because BIND would otherwise not forward requests for this domain. The Web server at the provider, for example, would not be accessible for this domain.

To start the name server, enter the command `systemctl start named` as `root`. Check with `systemctl status named` whether `named` (as the name server process is called) has been started successfully. Test the name server immediately on the local system with the `host` or `dig` programs, which should return `localhost` as the default server with the address `127.0.0.1`. If this is not the case, `/var/run/netconfig/resolv.conf` probably contains an incorrect name server entry or the file does not exist. For the first test, enter `host 127.0.0.1`, which should always work. If you get an error message, use `systemctl status named` to see whether the server is actually running. If the name server does not start or behaves unexpectedly, check the output of `journalctl -e`.

To use the name server of the provider (or one already running on your network) as the forwarder, enter the corresponding IP address or addresses in the `options` section under `forwarders`. The addresses included in *Example 19.1, "Forwarding Options in named.conf"* are examples only. Adjust these entries to your own setup.

EXAMPLE 19.1: FORWARDING OPTIONS IN NAMED.CONF

```
options {
    directory "/var/lib/named";
    forwarders { 10.11.12.13; 10.11.12.14; };
    listen-on { 127.0.0.1; 192.168.1.116; };
    allow-query { 127/8; 192.168/16 };
    notify no;
};
```

The `options` entry is followed by entries for the zone, `localhost`, and `0.0.127.in-addr.arpa`. The `type hint` entry under “.” should always be present. The corresponding files do not need to be modified and should work as they are. Also make sure that each entry is closed with a “;” and that the curly braces are in the correct places. After changing the configuration file `/etc/named.conf` or the zone files, tell BIND to reread them with `systemctl reload named`. Achieve the same by stopping and restarting the name server with `systemctl restart named`. Stop the server at any time by entering `systemctl stop named`.

19.5 The `/etc/named.conf` Configuration File

All the settings for the BIND name server itself are stored in the `/etc/named.conf` file. However, the zone data for the domains to handle (consisting of the host names, IP addresses, and so on) are stored in separate files in the `/var/lib/named` directory. The details of this are described later.

`/etc/named.conf` is roughly divided into two areas. One is the `options` section for general settings and the other consists of `zone` entries for the individual domains. A `logging` section and `acl` (access control list) entries are optional. Comment lines begin with a `#` sign or `//`. A minimal `/etc/named.conf` is shown in *Example 19.2, “A Basic `/etc/named.conf`”*.

EXAMPLE 19.2: A BASIC `/ETC/NAMED.CONF`

```
options {
    directory "/var/lib/named";
    forwarders { 10.0.0.1; };
    notify no;
};

zone "localhost" in {
    type master;
    file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" in {
    type master;
    file "127.0.0.zone";
};

zone "." in {
    type hint;
    file "root.hint";
};
```

19.5.1 Important Configuration Options

directory " FILENAME ";

Specifies the directory in which BIND can find the files containing the zone data. Usually, this is /var/lib/named.

forwarders { IP-ADDRESS ;};

Specifies the name servers (mostly of the provider) to which DNS requests should be forwarded if they cannot be resolved directly. Replace IP-ADDRESS with an IP address like 192.168.1.116.

forward first;

Causes DNS requests to be forwarded before an attempt is made to resolve them via the root name servers. Instead of forward first, forward only can be written to have all requests forwarded and none sent to the root name servers. This makes sense for firewall configurations.

listen-on port 53 { 127.0.0.1; IP-ADDRESS ;};

Tells BIND on which network interfaces and port to accept client queries. port 53 does not need to be specified explicitly, because 53 is the default port. Enter 127.0.0.1 to permit requests from the local host. If you omit this entry entirely, all interfaces are used by default.

listen-on-v6 port 53 {any;};

Tells BIND on which port it should listen for IPv6 client requests. The only alternative to any is none. As far as IPv6 is concerned, the server only accepts wild card addresses.

query-source address * port 53;

This entry is necessary if a firewall is blocking outgoing DNS requests. This tells BIND to post requests externally from port 53 and not from any of the high ports above 1024.

query-source-v6 address * port 53;

Tells BIND which port to use for IPv6 queries.

allow-query { 127.0.0.1; NET ;};

Defines the networks from which clients can post DNS requests. Replace NET with address information like 192.168.2.0/24. The /24 at the end is an abbreviated expression for the netmask (in this case 255.255.255.0).

allow-transfer !*;;

Controls which hosts can request zone transfers. In the example, such requests are completely denied with `! *`. Without this entry, zone transfers can be requested from anywhere without restrictions.

`statistics-interval 0;`

In the absence of this entry, BIND generates several lines of statistical information per hour in the system's journal. Set it to 0 to suppress these statistics completely or set an interval in minutes.

`cleaning-interval 720;`

This option defines at which time intervals BIND clears its cache. This triggers an entry in the system's journal each time it occurs. The time specification is in minutes. The default is 60 minutes.

`interface-interval 0;`

BIND regularly searches the network interfaces for new or nonexistent interfaces. If this value is set to `0`, this is not done and BIND only listens at the interfaces detected at start-up. Otherwise, the interval can be defined in minutes. The default is sixty minutes.

`notify no;`

`no` prevents other name servers from being informed when changes are made to the zone data or when the name server is restarted.

For a list of available options, read the manual page [man 5 named.conf](#).

19.5.2 Logging

What, how, and where logging takes place can be extensively configured in BIND. Normally, the default settings should be sufficient. *Example 19.3, "Entry to Disable Logging"*, shows the simplest form of such an entry and completely suppresses any logging.

EXAMPLE 19.3: ENTRY TO DISABLE LOGGING

```
logging {
    category default { null; };
};
```

19.5.3 Zone Entries

EXAMPLE 19.4: ZONE ENTRY FOR EXAMPLE.COM

```
zone "example.com" in {
    type master;
    file "example.com.zone";
    notify no;
};
```

After zone, specify the name of the domain to administer (example.com) followed by in and a block of relevant options enclosed in curly braces, as shown in *Example 19.4, "Zone Entry for example.com"*. To define a *slave zone*, switch the type to slave and specify a name server that administers this zone as master (which, in turn, may be a slave of another master), as shown in *Example 19.5, "Zone Entry for example.net"*.

EXAMPLE 19.5: ZONE ENTRY FOR EXAMPLE.NET

```
zone "example.net" in {
    type slave;
    file "slave/example.net.zone";
    masters { 10.0.0.1; };
};
```

The zone options:

type master;

By specifying master, tell BIND that the zone is handled by the local name server. This assumes that a zone file has been created in the correct format.

type slave;

This zone is transferred from another name server. It must be used together with masters.

type hint;

The zone . of the hint type is used to set the root name servers. This zone definition can be left as is.

file example.com.zone or file "slave/example.net.zone";

This entry specifies the file where zone data for the domain is located. This file is not required for a slave, because this data is pulled from another name server. To differentiate master and slave files, use the directory slave for the slave files.

```
masters { SERVER_IP_ADDRESS ;};
```

This entry is only needed for slave zones. It specifies from which name server the zone file should be transferred.

```
allow-update {! *};
```

This option controls external write access, which would allow clients to make a DNS entry—something not normally desirable for security reasons. Without this entry, zone updates are not allowed. The above entry achieves the same because ! * effectively bans any such activity.

19.6 Zone Files

Two types of zone files are needed. One assigns IP addresses to host names and the other does the reverse: it supplies a host name for an IP address.



Tip: Using the Dot (Period, Fullstop) in Zone Files

The "." has an important meaning in the zone files. If host names are given without a final dot (.), the zone is appended. Complete host names specified with a full domain name must end with a dot (.) to avoid having the domain added to it again. A missing or wrongly placed "." is probably the most frequent cause of name server configuration errors.

The first case to consider is the zone file example.com.zone, responsible for the domain example.com, shown in *Example 19.6, "The /var/lib/named/example.com.zone File"*.

EXAMPLE 19.6: THE /VAR/LIB/NAMED/EXAMPLE.COM.ZONE FILE

```
$TTL 2D ①
example.com. IN SOA      dns root.example.com. ( ②
                2003072441 ; serial ③
                1D        ; refresh ④
                2H        ; retry ⑤
                1W        ; expiry ⑥
                2D )      ; minimum ⑦

                IN NS     dns ⑧
                IN MX     10 mail dns ⑨
gate           IN A      192.168.5.1 ⑩
                IN A      10.0.0.1
```


| | | |
|---------|----------|-----------------|
| dns | IN A | 192.168.1.116 |
| mail | IN A | 192.168.3.108 |
| jupiter | IN A | 192.168.2.100 |
| venus | IN A | 192.168.2.101 |
| saturn | IN A | 192.168.2.102 |
| mercury | IN A | 192.168.2.103 |
| ntp | IN CNAME | dns 11 |
| dns6 | IN A6 0 | 2002:c0a8:174:: |

- ① \$TTL defines the default time to live that should apply to all the entries in this file. In this example, entries are valid for a period of two days (2 D).
- ② This is where the SOA (start of authority) control record begins:
 - The name of the domain to administer is example.com in the first position. This ends with ".", because otherwise the zone would be appended a second time. Alternatively, @ can be entered here, in which case the zone would be extracted from the corresponding entry in /etc/named.conf.
 - After IN SOA is the name of the name server in charge as master for this zone. The name is expanded from dns to dns.example.com, because it does not end with a ".".
 - An e-mail address of the person in charge of this name server follows. Because the @ sign already has a special meaning, "." is entered here instead. For root@example.com the entry must read root.example.com.. The "." must be included at the end to prevent the zone from being added.
 - The (includes all lines up to) into the SOA record.
- ③ The serial number is a 10 digit number. It must be changed each time this file is changed. It is needed to inform the secondary name servers (slave servers) of changes. For this, a 10 digit number of the date and run number, written as YYYYMMDDNN, has become the customary format (YYYY = year, MM = month and DD = day. NN is a sequence number in case you update it more than once on the given day).
- ④ The refresh rate specifies the time interval at which the secondary name servers verify the zone serial number. In this case, one day.
- ⑤ The retry rate specifies the time interval at which a secondary name server, in case of error, attempts to contact the primary server again. Here, two hours.
- ⑥ The expiration time specifies the time frame after which a secondary name server discards the cached data if it has not regained contact to the primary server. Here, a week.

- 7 The last entry in the SOA record specifies the negative caching TTL—the time for which results of unresolved DNS queries from other servers may be cached.
- 8 The IN NS specifies the name server responsible for this domain. dns is extended to dns.example.com because it does not end with a .". There can be several lines like this—one for the primary and one for each secondary name server. If notify is not set to no in /etc/named.conf, all the name servers listed here are informed of the changes made to the zone data.
- 9 The MX record specifies the mail server that accepts, processes, and forwards e-mails for the domain example.com. In this example, this is the host mail.example.com. The number in front of the host name is the preference value. If there are multiple MX entries, the mail server with the smallest value is taken first. If mail delivery to this server fails, the entry with the next-smallest value is used.
- 10 This and the following lines are the actual address records where one or more IP addresses are assigned to host names. The names are listed here without a ." because they do not include their domain, so example.com is added to all of them. Two IP addresses are assigned to the host gate, as it has two network cards. Wherever the host address is a traditional one (IPv4), the record is marked with A. If the address is an IPv6 address, the entry is marked with AAAA.



Note: IPv6 Syntax

The IPv6 record has a slightly different syntax than IPv4. Because of the fragmentation possibility, it is necessary to provide information about missed bits before the address. To fill up the IPv6 address with the needed number of “0”, add two colons at the correct place in the address.

```
pluto      AAAA 2345:00C1:CA11::1234:5678:9ABC:DEF0
pluto      AAAA 2345:00D2:DA11::1234:5678:9ABC:DEF0
```

- 11 The alias ntp can be used to address dns (CNAME means *canonical name*).

The pseudo domain in-addr.arpa is used for the reverse lookup of IP addresses into host names. It is appended to the network part of the address in reverse notation. So 192.168 is resolved into 168.192.in-addr.arpa. See [Example 19.7, “Reverse Lookup”](#).

EXAMPLE 19.7: REVERSE LOOKUP

```
$TTL 2D 1
```

```

168.192.in-addr.arpa.  IN SOA dns.example.com. root.example.com. ( ②
                    2003072441      ; serial
                    1D              ; refresh
                    2H              ; retry
                    1W              ; expiry
                    2D )            ; minimum

                    IN NS          dns.example.com. ③

1.5                  IN PTR          gate.example.com. ④
100.3                IN PTR          www.example.com.
253.2                IN PTR          cups.example.com.

```

- ① \$TTL defines the standard TTL that applies to all entries here.
- ② The configuration file should activate reverse lookup for the network `192.168`. Given that the zone is called `168.192.in-addr.arpa`, it should not be added to the host names. Therefore, all host names are entered in their complete form—with their domain and with a `."` at the end. The remaining entries correspond to those described for the previous `example.com` example.
See *Example 19.6, "The `/var/lib/named/example.com.zone` File"* for detail on the entries within this record.
- ③ This line specifies the name server responsible for this zone. This time, however, the name is entered in its complete form with the domain and a `."` at the end.
- ④ This, and the following lines, are the pointer records hinting at the IP addresses on the respective hosts. Only the last part of the IP address is entered at the beginning of the line, without the `."` at the end. Appending the zone to this (without the `.in-addr.arpa`) results in the complete IP address in reverse order.

Normally, zone transfers between different versions of BIND should be possible without any problems.

19.7 Dynamic Update of Zone Data

The term *dynamic update* refers to operations by which entries in the zone files of a master server are added, changed, or deleted. This mechanism is described in RFC 2136. Dynamic update is configured individually for each zone entry by adding an optional `allow-update` or `update-policy` rule. Zones to update dynamically should not be edited by hand.

Transmit the entries to update to the server with the command `nsupdate`. For the exact syntax of this command, check the manual page for `nsupdate` (`man 8 nsupdate`). For security reasons, any such update should be performed using TSIG keys as described in [Section 19.8, "Secure Transactions"](#).

19.8 Secure Transactions

Secure transactions can be made with transaction signatures (TSIGs) based on shared secret keys (also called TSIG keys). This section describes how to generate and use such keys.

Secure transactions are needed for communication between different servers and for the dynamic update of zone data. Making the access control dependent on keys is much more secure than merely relying on IP addresses.

Generate a TSIG key with the following command (for details, see `man dnssec-keygen`):

```
tux > sudo dnssec-keygen -a hmac-md5 -b 128 -n HOST host1-host2
```

This creates two files with names similar to these:

```
Khost1-host2.+157+34265.private Khost1-host2.+157+34265.key
```

The key itself (a string like `ejIkuCyyGJwwuN3xAteKgg==`) is found in both files. To use it for transactions, the second file (`Khost1-host2.+157+34265.key`) must be transferred to the remote host, preferably in a secure way (using `scp`, for example). On the remote server, the key must be included in the `/etc/named.conf` file to enable a secure communication between `host1` and `host2`:

```
key host1-host2 {
    algorithm hmac-md5;
    secret "ejIkuCyyGJwwuN3xAteKgg==";
};
```



Warning: File Permissions of `/etc/named.conf`

Make sure that the permissions of `/etc/named.conf` are properly restricted. The default for this file is `0640`, with the owner being `root` and the group `named`. As an alternative, move the keys to an extra file with specially limited permissions, which is then included from `/etc/named.conf`. To include an external file, use:

```
include "filename"
```

Replace filename with an absolute path to your file with keys.

To enable the server host1 to use the key for host2 (which has the address 10.1.2.3 in this example), the server's /etc/named.conf must include the following rule:

```
server 10.1.2.3 {
    keys { host1-host2. ;};
};
```

Analogous entries must be included in the configuration files of host2.

Add TSIG keys for any ACLs (access control lists, not to be confused with file system ACLs) that are defined for IP addresses and address ranges to enable transaction security. The corresponding entry could look like this:

```
allow-update { key host1-host2. ;};
```

This topic is discussed in more detail in the *BIND Administrator Reference Manual* under update-policy.

19.9 DNS Security

DNSSEC, or DNS security, is described in RFC 2535. The tools available for DNSSEC are discussed in the BIND Manual.

A zone considered secure must have one or several zone keys associated with it. These are generated with **dnssec-keygen**, as are the host keys. The DSA encryption algorithm is currently used to generate these keys. The public keys generated should be included in the corresponding zone file with an \$INCLUDE rule.

With the command **dnssec-signzone**, you can create sets of generated keys (keyset- files), transfer them to the parent zone in a secure manner, and sign them. This generates the files to include for each zone in /etc/named.conf.

19.10 For More Information

For more information, see the *BIND Administrator Reference Manual* from the `bind-doc` package, which is installed under `/usr/share/doc/packages/bind/arm`. Consider additionally consulting the RFCs referenced by the manual and the manual pages included with BIND. `/usr/share/doc/packages/bind/README.SUSE` contains up-to-date information about BIND in openSUSE Leap.

20 DHCP

The purpose of the *Dynamic Host Configuration Protocol* (DHCP) is to assign network settings centrally (from a server) rather than configuring them locally on every workstation. A host configured to use DHCP does not have control over its own static address. It is enabled to configure itself completely and automatically according to directions from the server. If you use the NetworkManager on the client side, you do not need to configure the client. This is useful if you have changing environments and only one interface active at a time. Never use NetworkManager on a machine that runs a DHCP server.

One way to configure a DHCP server is to identify each client using the hardware address of its network card (which should usually be fixed), then supply that client with identical settings each time it connects to the server. DHCP can also be configured to assign addresses to each relevant client dynamically from an address pool set up for this purpose. In the latter case, the DHCP server tries to assign the same address to the client each time it receives a request, even over extended periods. This works only if the network does not have more clients than addresses.

DHCP makes life easier for system administrators. Any changes, even bigger ones, related to addresses and the network configuration in general can be implemented centrally by editing the server's configuration file. This is much more convenient than reconfiguring numerous workstations. It is also much easier to integrate machines, particularly new machines, into the network, because they can be given an IP address from the pool. Retrieving the appropriate network settings from a DHCP server is especially useful in case of laptops regularly used in different networks.

In this chapter, the DHCP server will run in the same subnet as the workstations, 192.168.2.0/24 with 192.168.2.1 as gateway. It has the fixed IP address 192.168.2.254 and serves two address ranges, 192.168.2.10 to 192.168.2.20 and 192.168.2.100 to 192.168.2.200.

A DHCP server supplies not only the IP address and the netmask, but also the host name, domain name, gateway, and name server addresses for the client to use. In addition to that, DHCP allows several parameters to be configured in a centralized way, for example, a time server from which clients may poll the current time or even a print server.

20.1 Configuring a DHCP Server with YaST

To install a DHCP server, start YaST and select *Software > Software Management*. Choose *Filter > Patterns* and select *DHCP and DNS Server*. Confirm the installation of the dependent packages to finish the installation process.

Important: LDAP Support

The YaST DHCP module can be set up to store the server configuration locally (on the host that runs the DHCP server) or to have its configuration data managed by an LDAP server. To use LDAP, set up your LDAP environment before configuring the DHCP server.

For more information about LDAP, see *Book "Security and Hardening Guide", Chapter 6 "LDAP — A Directory Service"*.

The YaST DHCP module (`yast2-dhcp-server`) allows you to set up your own DHCP server for the local network. The module can run in wizard mode or expert configuration mode.

20.1.1 Initial Configuration (Wizard)

When the module is started for the first time, a wizard starts, prompting you to make a few basic decisions concerning server administration. Completing this initial setup produces a very basic server configuration that should function in its essential aspects. The expert mode can be used to deal with more advanced configuration tasks. Proceed as follows:

1. Select the interface from the list to which the DHCP server should listen and click *Select* and then *Next*. See *Figure 20.1, "DHCP Server: Card Selection"*.



Note: DHCP and **firewalld**

Please note that the option *Open Firewall for Selected Interfaces* does not (yet) support **firewalld** in openSUSE Leap 15.2. To manually open the DHCP port, run

```
tux > sudo firewall-cmd --zone=public --permanent --add-service=dhcp
tux > sudo firewall-cmd --reload
```

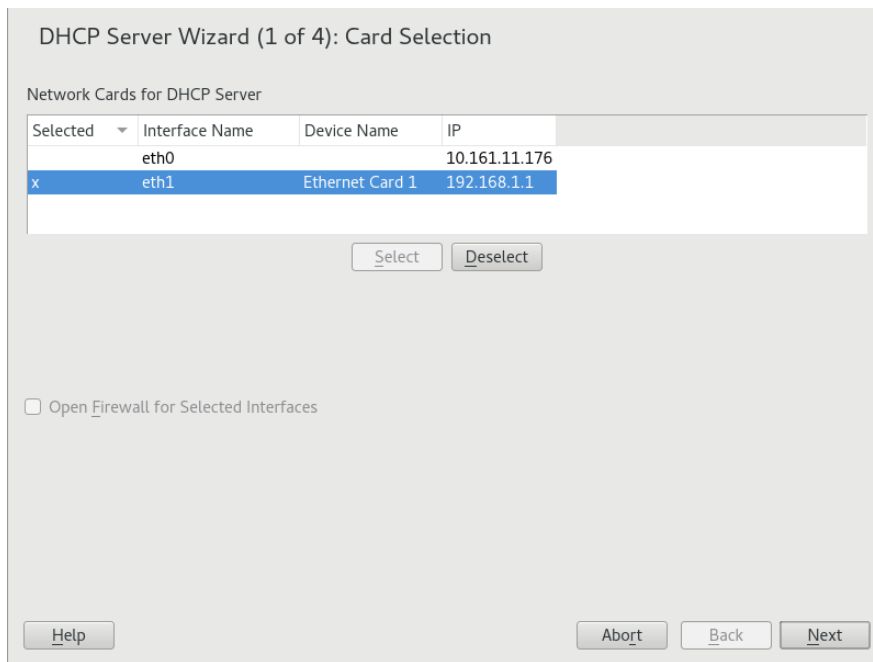



FIGURE 20.1: DHCP SERVER: CARD SELECTION

2. Use the check box to determine whether your DHCP settings should be automatically stored by an LDAP server. In the text boxes, provide the network specifics for all clients the DHCP server should manage. These specifics are the domain name, address of a time server, addresses of the primary and secondary name server, addresses of a print and a WINS server (for a mixed network with both Windows and Linux clients), gateway address, and lease time. See *Figure 20.2, "DHCP Server: Global Settings"*.

DHCP Server Wizard (2 of 4): Global Settings

LDAP Support

DHCP Server Name (optional)

Domain Name

Primary Name Server IP

Secondary Name Server IP

Default Gateway (Router)

NTP Time Server

Print Server

WINS Server

Default Lease Time

Units
 ▾

FIGURE 20.2: DHCP SERVER: GLOBAL SETTINGS

3. Configure how dynamic IP addresses should be assigned to clients. To do so, specify an IP range from which the server can assign addresses to DHCP clients. All these addresses must be covered by the same netmask. Also specify the lease time during which a client may keep its IP address without needing to request an extension of the lease. Optionally, specify the maximum lease time—the period during which the server reserves an IP address for a particular client. See *Figure 20.3, “DHCP Server: Dynamic DHCP”*.

DHCP Server Wizard (3 of 4): Dynamic DHCP

Subnet Information

| | | |
|--|--|---------------------------------|
| Current Network | Current Netmask | Netmask Bits |
| <input type="text" value="192.168.1.0"/> | <input type="text" value="255.255.255.0"/> | <input type="text" value="24"/> |
| Minimum IP Address | Maximum IP Address | |
| <input type="text" value="192.168.1.1"/> | <input type="text" value="192.168.1.254"/> | |

IP Address Range

| | |
|---|--|
| First IP Address | Last IP Address |
| <input type="text" value="192.168.200.11"/> | <input type="text" value="192.168.200.254"/> |

Allow Dynamic BOOTP

Lease Time

| | | | |
|--------------------------------|-------|--------------------------------|-------|
| Default | Units | Maximum | Units |
| <input type="text" value="4"/> | Hours | <input type="text" value="2"/> | Days |

FIGURE 20.3: DHCP SERVER: DYNAMIC DHCP

4. Define how the DHCP server should be started. Specify whether to start the DHCP server automatically when the system is booted or manually when needed (for example, for testing purposes). Click *Finish* to complete the configuration of the server. See [Figure 20.4, "DHCP Server: Start-Up"](#).

DHCP Server Wizard (4 of 4): Start-Up

Service Configuration

Current status: Inactive

After writing configuration:

After reboot:

FIGURE 20.4: DHCP SERVER: START-UP

5. Instead of using dynamic DHCP in the way described in the preceding steps, you can also configure the server to assign addresses in quasi-static fashion. Use the text boxes provided in the lower part to specify a list of the clients to manage in this way. Specifically, provide the *Name* and the *IP Address* to give to such a client, the *Hardware Address*, and the *Network Type* (token ring or Ethernet). Modify the list of clients, which is shown in the upper part with *Add*, *Edit*, and *Delete from List*. See [Figure 20.5, “DHCP Server: Host Management”](#).

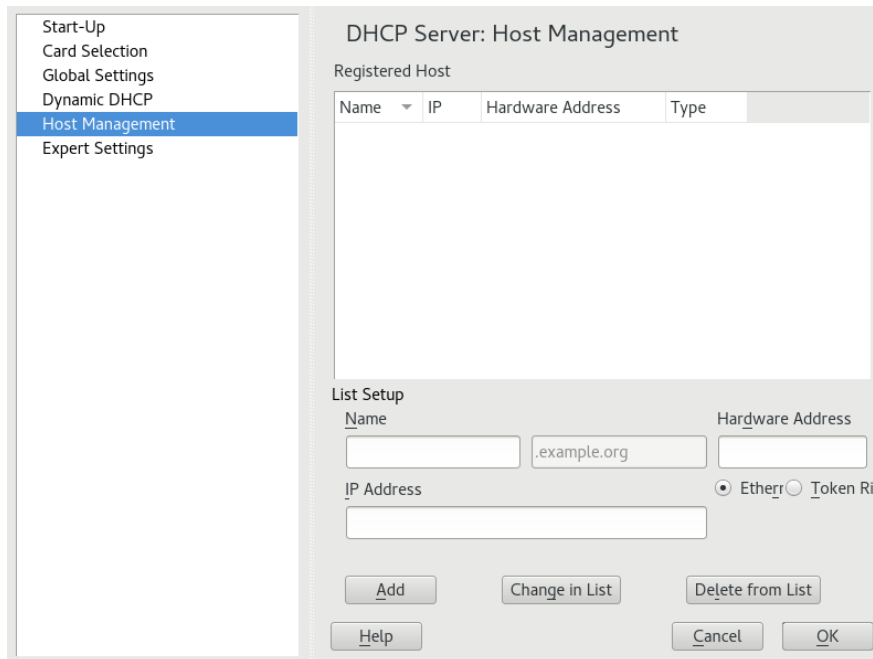


FIGURE 20.5: DHCP SERVER: HOST MANAGEMENT

20.1.2 DHCP Server Configuration (Expert)

In addition to the configuration method discussed earlier, there is also an expert configuration mode that allows you to change the DHCP server setup in every detail. Start the expert configuration by clicking *DHCP Server Expert Configuration* in the *Start-Up* dialog (see [Figure 20.4, “DHCP Server: Start-Up”](#)).

Chroot Environment and Declarations

In this first dialog, make the existing configuration editable by selecting *Start DHCP Server*. An important feature of the behavior of the DHCP server is its ability to run in a chroot environment, or chroot jail, to secure the server host. If the DHCP server should ever be compromised by an outside attack, the attacker will still be in the chroot jail, which prevents them from accessing the rest of the system. The lower part of the dialog displays

a tree view with the declarations that have already been defined. Modify these with *Add*, *Delete*, and *Edit*. Selecting *Advanced* takes you to additional expert dialogs. See [Figure 20.6, “DHCP Server: Chroot Jail and Declarations”](#). After selecting *Add*, define the type of declaration to add. With *Advanced*, view the log file of the server, configure TSIG key management, and adjust the configuration of the firewall according to the setup of the DHCP server.

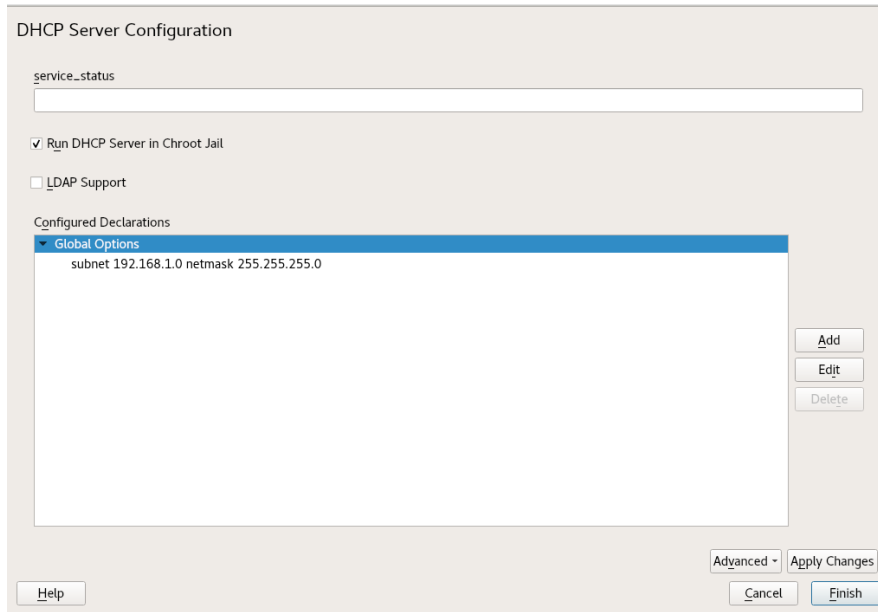


FIGURE 20.6: DHCP SERVER: CHROOT JAIL AND DECLARATIONS

Selecting the Declaration Type

The *Global Options* of the DHCP server are made up of several declarations. This dialog lets you set the declaration types *Subnet*, *Host*, *Shared Network*, *Group*, *Pool of Addresses*, and *Class*. This example shows the selection of a new subnet (see [Figure 20.7, “DHCP Server: Selecting a Declaration Type”](#)).

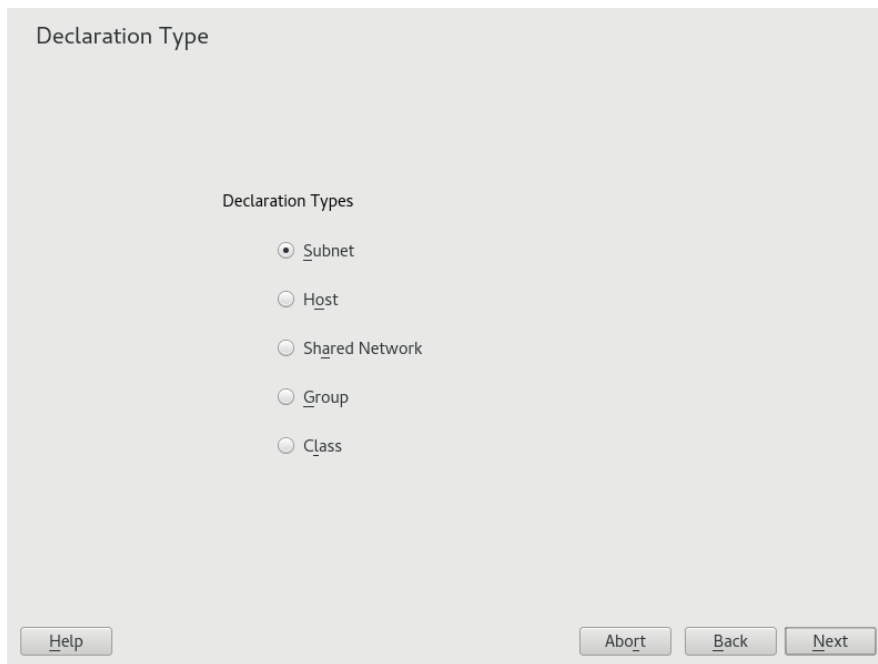


FIGURE 20.7: DHCP SERVER: SELECTING A DECLARATION TYPE

Subnet Configuration

This dialog allows you specify a new subnet with its IP address and netmask. In the middle part of the dialog, modify the DHCP server start options for the selected subnet using *Add*, *Edit*, and *Delete*. To set up dynamic DNS for the subnet, select *Dynamic DNS*.

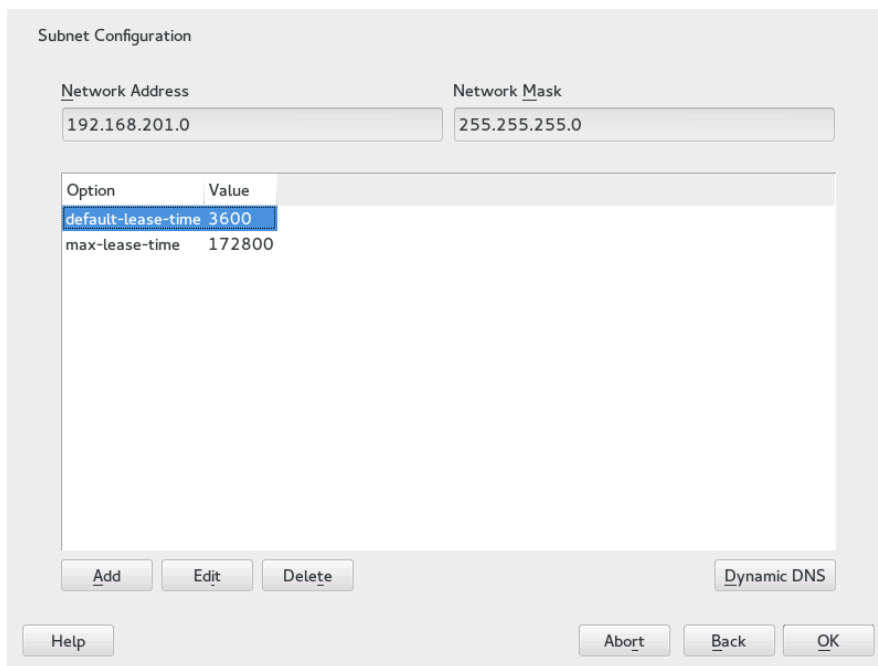


FIGURE 20.8: DHCP SERVER: CONFIGURING SUBNETS

TSIG Key Management

If you chose to configure dynamic DNS in the previous dialog, you can now configure the key management for a secure zone transfer. Selecting *OK* takes you to another dialog in which to configure the interface for dynamic DNS (see [Figure 20.10, “DHCP Server: Interface Configuration for Dynamic DNS”](#)).

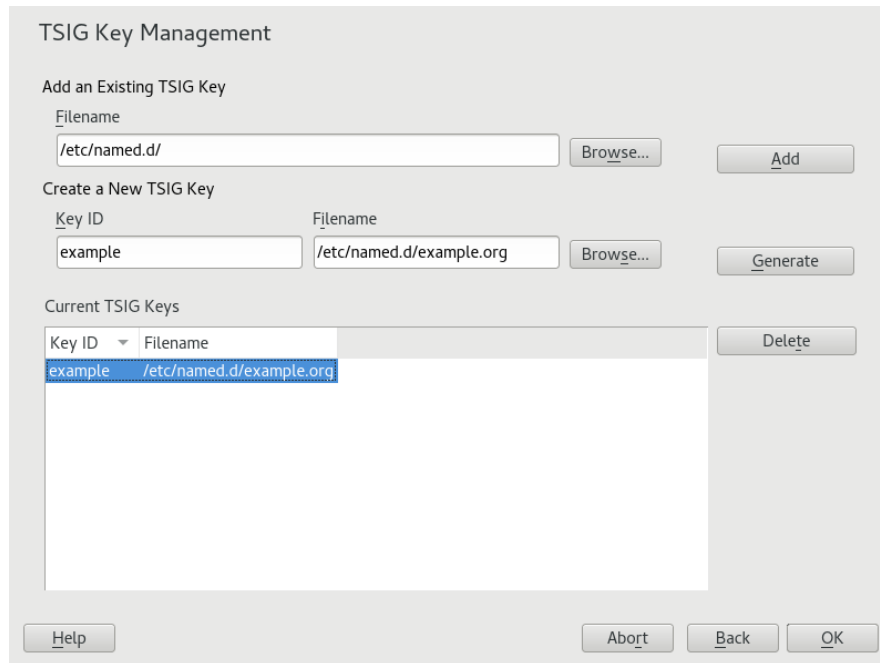


FIGURE 20.9: DHCP SERVER: TSIG CONFIGURATION

Dynamic DNS: Interface Configuration

You can now activate dynamic DNS for the subnet by selecting *Enable Dynamic DNS for This Subnet*. After doing so, use the drop-down box to activate the TSIG keys for forward and reverse zones, making sure that the keys are the same for the DNS and the DHCP server. With *Update Global Dynamic DNS Settings*, enable the automatic update and adjustment of the global DHCP server settings according to the dynamic DNS environment. Finally, define which forward and reverse zones should be updated per dynamic DNS, specifying the name of the primary name server for each of the two zones. Selecting *OK* returns to the subnet configuration dialog (see [Figure 20.8, “DHCP Server: Configuring Subnets”](#)). Selecting *OK* again returns to the original expert configuration dialog.

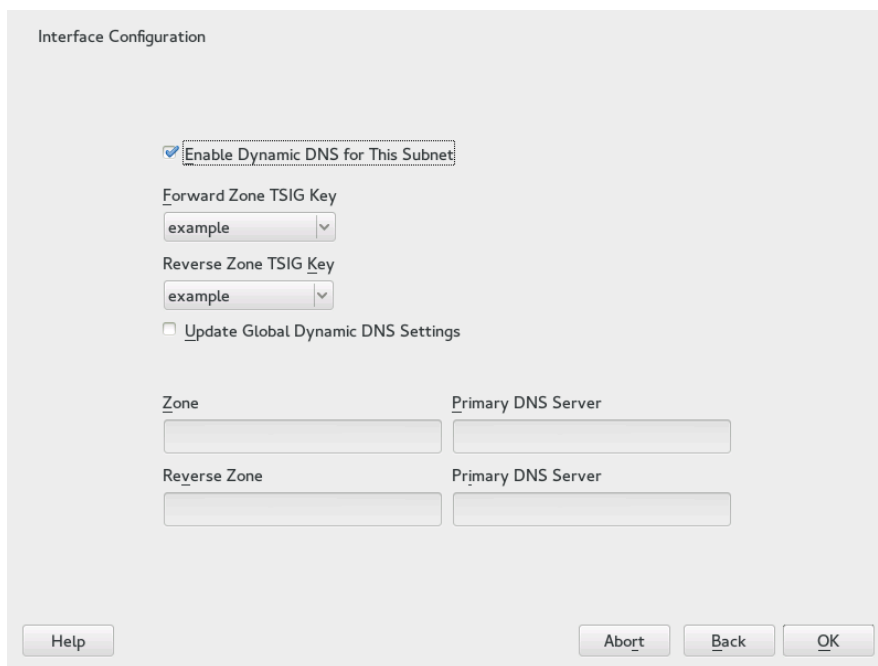


FIGURE 20.10: DHCP SERVER: INTERFACE CONFIGURATION FOR DYNAMIC DNS

Network Interface Configuration

To define the interfaces the DHCP server should listen to and to adjust the firewall configuration, select *Advanced > Interface Configuration* from the expert configuration dialog. From the list of interfaces displayed, select one or more that should be attended by the DHCP server. If clients in all subnets need to be able to communicate with the server and the server host also runs a firewall, adjust the firewall accordingly.



Note: DHCP and **firewalld**

Please note that the option *Open Firewall for Selected Interfaces* does not (yet) support **firewalld** in openSUSE Leap 15.2. To manually open the DHCP port, run

```
tux > sudo firewall-cmd --zone=public --permanent --add-service=dhcp
tux > sudo firewall-cmd --reload
```

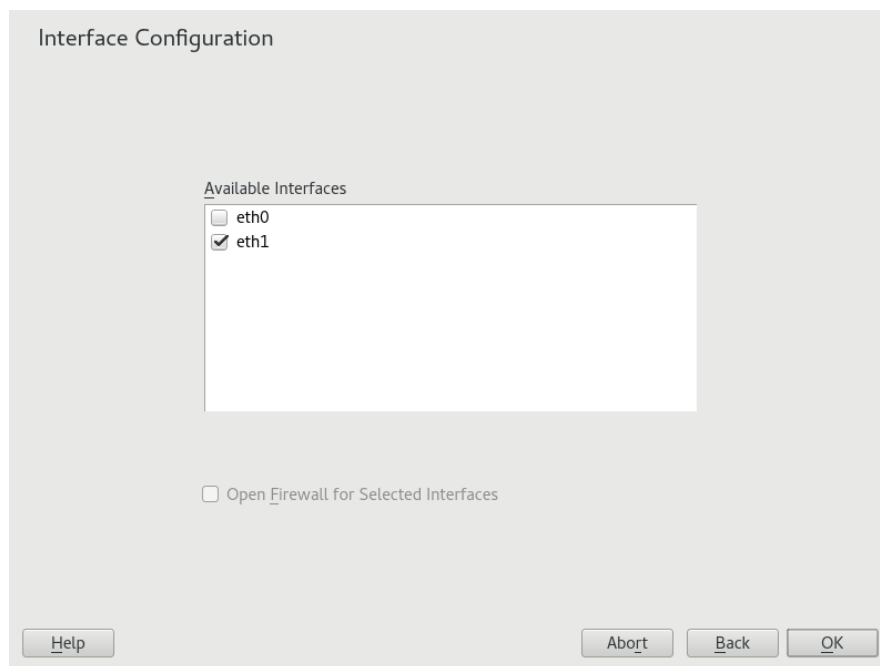



FIGURE 20.11: DHCP SERVER: NETWORK INTERFACE AND FIREWALL

After completing all configuration steps, close the dialog with *OK*. The server is now started with its new configuration.

20.2 DHCP Software Packages

Both the DHCP server and the DHCP clients are available for openSUSE Leap. The DHCP server available is dhcpcd (published by the Internet Systems Consortium). On the client side, there is dhcp-client (also from ISC) and tools coming with the wicked package.

By default, the wicked tools are installed with the services wickedd-dhcp4 and wickedd-dhcp6. Both are launched automatically on each system boot to watch for a DHCP server. They do not need a configuration file to do their job and work out of the box in most standard setups. For more complex situations, use the ISC dhcp-client, which is controlled by means of the configuration files /etc/dhclient.conf and /etc/dhclient6.conf.

20.3 The DHCP Server dhcpd

The core of any DHCP system is the dynamic host configuration protocol daemon. This server *leases* addresses and watches how they are used, according to the settings defined in the configuration file `/etc/dhcpd.conf`. By changing the parameters and values in this file, a system administrator can influence the program's behavior in numerous ways. Look at the basic sample `/etc/dhcpd.conf` file in *Example 20.1, "The Configuration File /etc/dhcpd.conf"*.

EXAMPLE 20.1: THE CONFIGURATION FILE /ETC/DHCPD.CONF

```
default-lease-time 600;          # 10 minutes
max-lease-time 7200;            # 2  hours

option domain-name "example.com";
option domain-name-servers 192.168.1.116;
option broadcast-address 192.168.2.255;
option routers 192.168.2.1;
option subnet-mask 255.255.255.0;

subnet 192.168.2.0 netmask 255.255.255.0
{
    range 192.168.2.10 192.168.2.20;
    range 192.168.2.100 192.168.2.200;
}
```

This simple configuration file should be sufficient to get the DHCP server to assign IP addresses in the network. Make sure that a semicolon is inserted at the end of each line, because otherwise `dhcpd` is not started.

The sample file can be divided into three sections. The first one defines how many seconds an IP address is leased to a requesting client by default (`default-lease-time`) before it should apply for renewal. This section also includes a statement of the maximum period for which a machine may keep an IP address assigned by the DHCP server without applying for renewal (`max-lease-time`).

In the second part, some basic network parameters are defined on a global level:

- The line `option domain-name` defines the default domain of your network.
- With the entry `option domain-name-servers`, specify up to three values for the DNS servers used to resolve IP addresses into host names and vice versa. Ideally, configure a name server on your machine or somewhere else in your network before setting up DHCP. That name server should also define a host name for each dynamic address and vice versa. To learn how to configure your own name server, read *Chapter 19, The Domain Name System*.

- The line `option broadcast-address` defines the broadcast address the requesting client should use.
- With `option routers`, set where the server should send data packets that cannot be delivered to a host on the local network (according to the source and target host address and the subnet mask provided). Usually, especially in smaller networks, this router is identical to the Internet gateway.
- With `option subnet-mask`, specify the netmask assigned to clients.

The last section of the file defines a network, including a subnet mask. To finish, specify the address range that the DHCP daemon should use to assign IP addresses to interested clients. In *Example 20.1, “The Configuration File `/etc/dhcpd.conf`”*, clients may be given any address between `192.168.2.10` and `192.168.2.20` or `192.168.2.100` and `192.168.2.200`.

After editing these few lines, you should be able to activate the DHCP daemon with the command `systemctl start dhcpd`. It will be ready for use immediately. Use the command `rcdhcpd check-syntax` to perform a brief syntax check. If you encounter any unexpected problems with your configuration (the server aborts with an error or does not return `done` on start), you should be able to find out what has gone wrong by looking for information either in the main system log that can be queried with the command `journalctl` (see *Chapter 11, `journalctl`: Query the systemd Journal* for more information).

On a default openSUSE Leap system, the DHCP daemon is started in a chroot environment for security reasons. The configuration files must be copied to the chroot environment so the daemon can find them. Normally, there is no need to worry about this because the command `systemctl start dhcpd` automatically copies the files.

20.3.1 Clients with Fixed IP Addresses

DHCP can also be used to assign a predefined, static address to a specific client. Addresses assigned explicitly always take priority over dynamic addresses from the pool. A static address never expires in the way a dynamic address would, for example, if there were not enough addresses available and the server needed to redistribute them among clients.

To identify a client configured with a static address, `dhcpd` uses the hardware address (which is a globally unique, fixed numerical code consisting of six octet pairs) for the identification of all network devices (for example, `00:30:6E:08:EC:80`). If the respective lines, like the ones in

Example 20.2, "Additions to the Configuration File", are added to the configuration file of *Example 20.1, "The Configuration File /etc/dhcpd.conf"*, the DHCP daemon always assigns the same set of data to the corresponding client.

EXAMPLE 20.2: ADDITIONS TO THE CONFIGURATION FILE

```
host jupiter {
hardware ethernet 00:30:6E:08:EC:80;
fixed-address 192.168.2.100;
}
```

The name of the respective client (`host HOSTNAME`, here `jupiter`) is entered in the first line and the MAC address in the second line. On Linux hosts, find the MAC address with the command `ip link show` followed by the network device (for example, `eth0`). The output should contain something like

```
link/ether 00:30:6E:08:EC:80
```

In the preceding example, a client with a network card having the MAC address `00:30:6E:08:EC:80` is assigned the IP address `192.168.2.100` and the host name `jupiter` automatically. The type of hardware to enter is `ethernet` in nearly all cases, although `token-ring`, which is often found on IBM systems, is also supported.

20.3.2 The openSUSE Leap Version

To improve security, the openSUSE Leap version of the ISC's DHCP server comes with the non-root/chroot patch by Ari Edelkind applied. This enables `dhcpd` to run with the user ID `nobody` and run in a chroot environment (`/var/lib/dhcp`). To make this possible, the configuration file `dhcpd.conf` must be located in `/var/lib/dhcp/etc`. The init script automatically copies the file to this directory when starting.

Control the server's behavior regarding this feature by means of entries in the file `/etc/sysconfig/dhcpd`. To run `dhcpd` without the chroot environment, set the variable `DHCPD_RUN_CHROOTED` in `/etc/sysconfig/dhcpd` to "no".

To enable `dhcpd` to resolve host names even from within the chroot environment, some other configuration files must be copied as well:

- `/etc/localtime`
- `/etc/host.conf`

- /etc/hosts
- /var/run/netconfig/resolv.conf

These files are copied to /var/lib/dhcp/etc/ when starting the init script. Take these copies into account for any changes that they require if they are dynamically modified by scripts like /etc/ppp/ip-up. However, there should be no need to worry about this if the configuration file only specifies IP addresses (instead of host names).

If your configuration includes additional files that should be copied into the chroot environment, set these under the variable DHCPD_CONF_INCLUDE_FILES in the file /etc/sysconfig/dhcpd. To ensure that the DHCP logging facility keeps working even after a restart of the syslog daemon, there is an additional entry SYSLOGD_ADDITIONAL_SOCKET_DHCP in the file /etc/sysconfig/syslog.

20.4 For More Information

More information about DHCP is available at the Web site of the *Internet Systems Consortium* (<https://www.isc.org/dhcp/>). Information is also available in the dhcpd, dhcpd.conf, dhcpd.leases, and dhcp-options man pages.

21 Samba

Using Samba, a Unix machine can be configured as a file and print server for macOS, Windows, and OS/2 machines. Samba has developed into a fully-fledged and rather complex product. Configure Samba with YaST, or by editing the configuration file manually.

21.1 Terminology

The following are some terms used in Samba documentation and in the YaST module.

SMB protocol

Samba uses the SMB (server message block) protocol that is based on the NetBIOS services. Microsoft released the protocol so other software manufacturers could establish connections to a Microsoft domain network. With Samba, the SMB protocol works on top of the TCP/IP protocol, so the TCP/IP protocol must be installed on all clients.

CIFS protocol

The CIFS (common Internet file system) protocol, also known as SMB1 is an early version of the SMB protocol. CIFS defines a standard remote file system access protocol for use over the network, enabling groups of users to work together and share documents across the network. In recent versions of Samba, it is disabled by default for security reasons.

NetBIOS

NetBIOS is a software interface (API) designed for communication between machines providing a name service. It enables machines connected to the network to reserve names for themselves. After reservation, these machines can be addressed by name. There is no central process that checks names. Any machine on the network can reserve as many names as it wants as long as the names are not already in use. The NetBIOS interface can be implemented for different network architectures. An implementation that works relatively closely with network hardware is called NetBEUI, but this is often called NetBIOS. Network protocols implemented with NetBIOS are IPX from Novell (NetBIOS via TCP/IP) and TCP/IP.

The NetBIOS names sent via TCP/IP have nothing in common with the names used in `/etc/hosts` or those defined by DNS. NetBIOS uses its own, completely independent naming convention. However, it is recommended to use names that correspond to DNS host names to make administration easier or use DNS natively. This is the default used by Samba.

Samba server

Samba server provides SMB/CIFS services and NetBIOS over IP naming services to clients. For Linux, there are three daemons for Samba server: `smbd` for SMB/CIFS services, `nmbd` for naming services, and `winbind` for authentication.

Samba client

The Samba client is a system that uses Samba services from a Samba server over the SMB protocol. Common operating systems, such as Windows and macOS support the SMB protocol. The TCP/IP protocol must be installed on all computers. Samba provides a client for the different Unix flavors. For Linux, there is a kernel module for SMB that allows the integration of SMB resources on the Linux system level. You do not need to run any daemon for the Samba client.

Shares

SMB servers provide resources to the clients by means of shares. Shares are printers and directories with their subdirectories on the server. It is exported by means of a name and can be accessed by its name. The share name can be set to any name—it does not need to be the name of the export directory. A printer is also assigned a name. Clients can access the printer by its name.

DC

A domain controller (DC) is a server that handles accounts in a domain. For data replication, additional domain controllers are available in one domain.

21.2 Installing a Samba Server

To install a Samba server, start YaST and select *Software > Software Management*. Choose *View > Patterns* and select *File Server*. Confirm the installation of the required packages to finish the installation process.

21.3 Starting and Stopping Samba

You can start or stop the Samba server automatically (during boot) or manually. Starting and stopping policy is a part of the YaST Samba server configuration described in [Section 21.4.1, “Configuring a Samba Server with YaST”](#).

From a command line, stop services required for Samba with `systemctl stop smb nmb` and start them with `systemctl start nmb smb`. The `smb` service cares about `winbind` if needed.



Tip: winbind

`winbind` is an independent service, and as such is also offered as an individual `samba-winbind` package.

21.4 Configuring a Samba Server

A Samba server in openSUSE® Leap can be configured in two different ways: with YaST or manually. Manual configuration offers a higher level of detail, but lacks the convenience of the YaST GUI.

21.4.1 Configuring a Samba Server with YaST

To configure a Samba server, start YaST and select *Network Services > Samba Server*.

21.4.1.1 Initial Samba Configuration

When starting the module for the first time, the *Samba Installation* dialog starts, prompting you to make a few basic decisions concerning administration of the server. At the end of the configuration, it prompts for the Samba administrator password (*Samba Root Password*). For later starts, the *Samba Configuration* dialog appears.

The *Samba Installation* dialog consists of two steps and optional detailed settings:

Workgroup or Domain Name

Select an existing name from *Workgroup or Domain Name* or enter a new one and click *Next*.

Samba Server Type

In the next step, specify whether your server should act as a primary domain controller (PDC), backup domain controller (BDC), or not act as a domain controller. Continue with *Next*.

If you do not want to proceed with a detailed server configuration, confirm with *OK*. Then in the final pop-up box, set the *Samba root Password*.

You can change all settings later in the *Samba Configuration* dialog with the *Start-Up*, *Shares*, *Identity*, *Trusted Domains*, and *LDAP Settings* tabs.

21.4.1.2 Enabling Current Versions of the SMB Protocol on the Server

On clients running current versions of openSUSE Leap or other recent Linux versions, the insecure SMB1/CIFS protocol is disabled by default. However, existing instances of Samba may be configured to only serve shares using the SMB1/CIFS version of the protocol. To interact with such clients, you need to configure Samba to serve shares using at least the SMB 2.1 protocol.

There are setups in which only SMB1 can be used, for example, because they rely on SMB1's/CIFS's Unix extensions. These extensions have not been ported to newer protocol versions. If you are in this situation, consider changing your setup or see [Section 21.5.2, "Mounting SMB1/CIFS Shares on Clients"](#).

To do so, in the configuration file `/etc/samba/smb.conf`, set the global parameter `server max protocol = SMB2_10`. For a list of all possible values, see `man smb.conf`.

21.4.1.3 Advanced Samba Configuration

During the first start of the Samba server module the *Samba Configuration* dialog appears directly after the two initial steps described in [Section 21.4.1.1, "Initial Samba Configuration"](#). Use it to adjust your Samba server configuration.

After editing your configuration, click *OK* to save your settings.

21.4.1.3.1 Starting the Server

In the *Start Up* tab, configure the start of the Samba server. To start the service every time your system boots, select *During Boot*. To activate manual start, choose *Manually*. More information about starting a Samba server is provided in [Section 21.3, "Starting and Stopping Samba"](#).

In this tab, you can also open ports in your firewall. To do so, select *Open Port in Firewall*. If you have multiple network interfaces, select the network interface for Samba services by clicking *Firewall Details*, selecting the interfaces, and clicking *OK*.

21.4.1.3.2 Shares

In the *Shares* tab, determine the Samba shares to activate. There are some predefined shares, like homes and printers. Use *Toggle Status* to switch between *Active* and *Inactive*. Click *Add* to add new shares and *Delete* to delete the selected share.

Allow Users to Share Their Directories enables members of the group in *Permitted Group* to share directories they own with other users. For example, users for a local scope or DOMAIN\Users for a domain scope. The user also must make sure that the file system permissions allow access. With *Maximum Number of Shares*, limit the total amount of shares that may be created. To permit access to user shares without authentication, enable *Allow Guest Access*.

21.4.1.3.3 Identity

In the *Identity* tab, you can determine the domain with which the host is associated (*Base Settings*) and whether to use an alternative host name in the network (*NetBIOS Hostname*). It is also possible to use Microsoft Windows Internet Name Service (WINS) for name resolution. In this case, activate *Use WINS for Hostname Resolution* and decide whether to *Retrieve WINS server via DHCP*. To set expert global settings or set a user authentication source, for example LDAP instead of TDB database, click *Advanced Settings*.

21.4.1.3.4 Trusted Domains

To enable users from other domains to access your domain, make the appropriate settings in the *Trusted Domains* tab. To add a new domain, click *Add*. To remove the selected domain, click *Delete*.

21.4.1.3.5 LDAP Settings

In the tab *LDAP Settings*, you can determine the LDAP server to use for authentication. To test the connection to your LDAP server, click *Test Connection*. To set expert LDAP settings or use default values, click *Advanced Settings*.

For more information about LDAP configuration, see *Book "Security and Hardening Guide", Chapter 6 "LDAP—A Directory Service"*.

21.4.2 Configuring the Server Manually

If you intend to use Samba as a server, install `samba`. The main configuration file for Samba is `/etc/samba/smb.conf`. This file can be divided into two logical parts. The `[global]` section contains the central and global settings. The following default sections contain the individual file and printer shares:

- `[homes]`
- `[profiles]`
- `[users]`
- `[groups]`
- `[printers]`
- `[print$]`

Using this approach, options of the shares can be set differently or globally in the `[global]` section, which makes the configuration file easier to understand.

21.4.2.1 The global Section

The following parameters of the `[global]` section should be modified to match the requirements of your network setup, so other machines can access your Samba server via SMB in a Windows environment.

```
workgroup = WORKGROUP
```

This line assigns the Samba server to a workgroup. Replace `WORKGROUP` with an appropriate workgroup of your networking environment. Your Samba server appears under its DNS name unless this name has been assigned to some other machine in the network. If the DNS name is not available, set the server name using `netbiosname=MYNAME`. For more details about this parameter, see the `smb.conf` man page.

```
os level = 20
```

This parameter triggers whether your Samba server tries to become LMB (local master browser) for its workgroup. Choose a very low value such as 2 to spare the existing Windows network from any interruptions caused by a misconfigured Samba server. More information about this topic can be found in the Network Browsing chapter of the Samba 3 Howto; for more information on the Samba 3 Howto, see [Section 21.9, "For More Information"](#). If no other SMB server is in your network (such as a Windows 2000 server) and you want the Samba server to keep a list of all systems present in the local environment, set the os level to a higher value (for example, 65). Your Samba server is then chosen as LMB for your local network.

When changing this setting, consider carefully how this could affect an existing Windows network environment. First test the changes in an isolated network or at a noncritical time of day.

wins support and wins server

To integrate your Samba server into an existing Windows network with an active WINS server, enable the wins server option and set its value to the IP address of that WINS server.

If your Windows machines are connected to separate subnets and need to still be aware of each other, you have to set up a WINS server. To turn a Samba server into such a WINS server, set the option wins support = Yes. Make sure that only one Samba server of the network has this setting enabled. The options wins server and wins support must never be enabled at the same time in your smb.conf file.

21.4.2.2 Shares

The following examples illustrate how a CD-ROM drive and the user directories (homes) are made available to the SMB clients.

[cdrom]

To avoid having the CD-ROM drive accidentally made available, these lines are deactivated with comment marks (semicolons in this case). Remove the semicolons in the first column to share the CD-ROM drive with Samba.

EXAMPLE 21.1: A CD-ROM SHARE

```
[cdrom]
;comment = Linux CD-ROM
;path = /media/cdrom
```

```
locking = No
```

[cdrom] and comment

The [cdrom] section entry is the name of the share that can be seen by all SMB clients on the network. An additional comment can be added to further describe the share.

path = /media/cdrom

path exports the directory /media/cdrom.

By means of a very restrictive default configuration, this kind of share is only made available to the users present on this system. If this share should be made available to everybody, add a line guest ok = yes to the configuration. This setting gives read permissions to anyone on the network. It is recommended to handle this parameter with great care. This applies even more to the use of this parameter in the [global] section.

[homes]

The [homes] share is of special importance here. If the user has a valid account and password for the Linux file server and their own home directory, they can be connected to it.

EXAMPLE 21.2: [HOMES] SHARE

```
[homes]
comment = Home Directories
valid users = %S
browseable = No
read only = No
inherit acls = Yes
```

[homes]

As long as there is no other share using the share name of the user connecting to the SMB server, a share is dynamically generated using the [homes] share directives. The resulting name of the share is the user name.

valid users = %S

%S is replaced with the concrete name of the share when a connection has been successfully established. For a [homes] share, this is always the user name. As a consequence, access rights to a user's share are restricted exclusively to that user.

browseable = No

This setting makes the share invisible in the network environment.

read only = No

By default, Samba prohibits write access to any exported share by means of the read only = Yes parameter. To make a share writable, set the value read only = No, which is synonymous with writable = Yes.

create mask = 0640

Systems that are not based on MS Windows NT do not understand the concept of Unix permissions, so they cannot assign permissions when creating a file. The parameter create mask defines the access permissions assigned to newly created files. This only applies to writable shares. In effect, this setting means the owner has read and write permissions and the members of the owner's primary group have read permissions. valid users = %S prevents read access even if the group has read permissions. For the group to have read or write access, deactivate the line valid users = %S.



Warning: Do Not Share NFS Mounts with Samba

Sharing NFS mounts with Samba may result in data loss and is not supported. Install Samba directly on the file server or consider using alternatives such as iSCSI.

21.4.2.3 Security Levels

To improve security, each share access can be protected with a password. SMB offers the following ways of checking permissions:

User Level Security (security = user)

This variant introduces the concept of the user to SMB. Each user must register with the server with their own password. After registration, the server can grant access to individual exported shares dependent on user names.

ADS Level Security (security = ADS)

In this mode, Samba will act as a domain member in an Active Directory environment. To operate in this mode, the machine running Samba needs Kerberos installed and configured. You must join the machine using Samba to the ADS realm. This can be done using the *YaST Windows Domain Membership* module.

Domain Level Security (security = domain)

This mode will only work correctly if the machine has been joined into a Windows NT Domain. Samba will try to validate user name and password by passing it to a Windows NT Primary or Backup Domain Controller. The same way as a Windows NT Server would do. It expects the `encrypted passwords` parameter to be set to yes.

The selection of share, user, server, or domain level security applies to the entire server. It is not possible to offer individual shares of a server configuration with share level security and others with user level security. However, you can run a separate Samba server for each configured IP address on a system.

More information about this subject can be found in the Samba 3 HOWTO. For multiple servers on one system, pay attention to the options interfaces and bind interfaces only.

21.5 Configuring Clients

Clients can only access the Samba server via TCP/IP. NetBEUI and NetBIOS via IPX cannot be used with Samba.

21.5.1 Configuring a Samba Client with YaST

Configure a Samba client to access resources (files or printers) on the Samba or Windows server. Enter the NT or Active Directory domain or workgroup in the dialog *Network Services > Windows Domain Membership*. If you activate *Also Use SMB Information for Linux Authentication*, the user authentication runs over the Samba, NT or Kerberos server.

Click *Expert Settings* for advanced configuration options. For example, use the *Mount Server Directories* table to enable mounting server home directory automatically with authentication. This way users can access their home directories when hosted on CIFS. For details, see the pam_mount man page.

After completing all settings, confirm the dialog to finish the configuration.

21.5.2 Mounting SMB1/CIFS Shares on Clients

The first version of the SMB network protocol, SMB1 or CIFS, is an old and insecure protocol which has been deprecated by its originator, Microsoft. For security reasons, the mount command on openSUSE Leap will only mount SMB shares using newer protocol versions by default, namely SMB 2.1, SMB 3.0, or SMB 3.02.

However, this change only affects **mount** and mounting via `/etc/fstab`. SMB1 is still available by explicitly requiring it. Use the following command:

- The **smbclient** tool.
- The Samba server software shipped with openSUSE.

There are setups in which this default setting will lead to connection failures, because only SMB1 can be used:

- Setups using an SMB server that does not support newer SMB protocol versions. Windows has offered SMB 2.1 support since Windows 7 and Windows Server 2008.
- Setups that rely on SMB1's/CIFS's Unix extensions. These extensions have not been ported to newer protocol versions.



Important: Decreased System Security

Following the instruction below makes it possible to exploit security issues. For more information about the issues, see <https://blogs.technet.microsoft.com/filecab/2016/09/16/stop-using-smb1/>.

As soon as possible, upgrade your server to allow for a more secure SMB version.

For information about enabling suitable protocol versions on openSUSE Leap, see [Section 21.4.1.2, "Enabling Current Versions of the SMB Protocol on the Server"](#).

If you need to enable SMB1 shares on the current openSUSE Leap kernel, add the option `vers=1.0` to the **mount** command line you use:

```
root # mount -t cifs //HOST/SHARE /MOUNT_POINT -o username=USER_ID,vers=1.0
```

Alternatively, you can enable SMB1 shares globally within your openSUSE Leap installation. To do so, add the following to `/etc/samba/smb.conf` under the section `[global]`:

```
client min protocol = CORE
```


21.6 Samba as Login Server

In enterprise settings, it is often desirable to allow access only to users registered on a central instance. In a Windows-based network, this task is handled by a primary domain controller (PDC). You can use a Windows NT server configured as PDC, but this task can also be done with a Samba server. The entries that must be made in the `[global]` section of `smb.conf` are shown in *Example 21.3, "Global Section in smb.conf"*.

EXAMPLE 21.3: GLOBAL SECTION IN SMB.CONF

```
[global]
  workgroup = WORKGROUP
  domain logons = Yes
  domain master = Yes
```

It is necessary to prepare user accounts and passwords in an encryption format that conforms with Windows. Do this with the command `smbpasswd -a name`. Create the domain account for the computers, required by the Windows domain concept, with the following commands:

```
useradd hostname
smbpasswd -a -m hostname
```

With the `useradd` command, a dollar sign is added. The command `smbpasswd` inserts this automatically when the parameter `-m` is used. The commented configuration example (`/usr/share/doc/packages/samba/examples/smb.conf.SUSE`) contains settings that automate this task.

```
add machine script = /usr/sbin/useradd -g nogroup -c "NT Machine Account" \
-s /bin/false %m
```

To make sure that Samba can execute this script correctly, choose a Samba user with the required administrator permissions and add it to the `ntadmin` group. Then all users belonging to this Linux group can be assigned `Domain Admin` status with the command:

```
net groupmap add ntgroup="Domain Admins" unixgroup=ntadmin
```

21.7 Samba Server in the Network with Active Directory

If you run Linux servers and Windows servers together, you can build two independent authentication systems and networks or connect servers to one network with one central authentication system. Because Samba can cooperate with an active directory domain, you can join your openSUSE Leap server with an Active Directory (AD) domain.

To join an AD domain proceed as follows:

1. Log in as `root` and start YaST.
2. Start *Network Services > Windows Domain Membership*.
3. Enter the domain to join at *Domain or Workgroup* in the *Windows Domain Membership* screen.

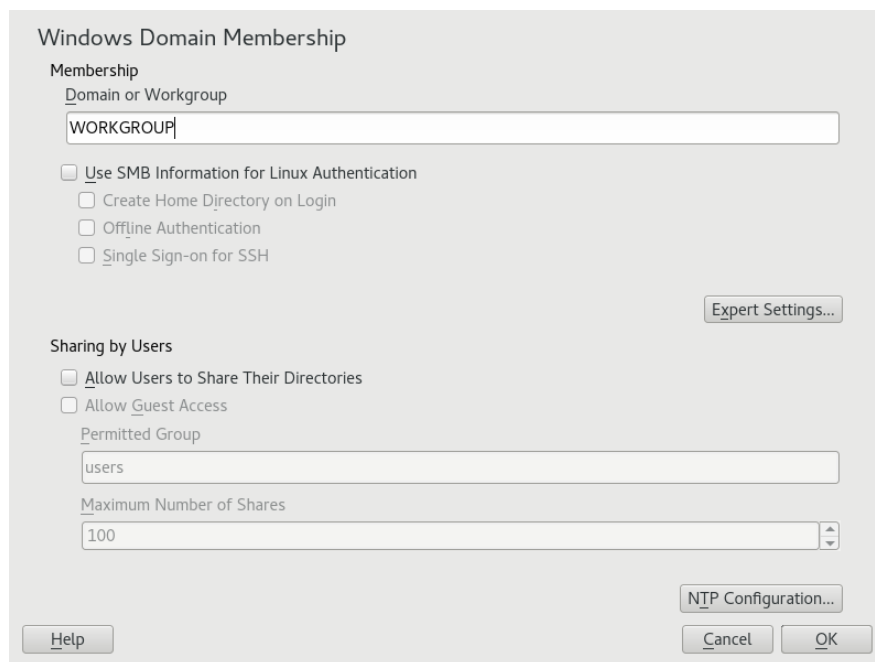


FIGURE 21.1: DETERMINING WINDOWS DOMAIN MEMBERSHIP

4. Check *Also Use SMB Information for Linux Authentication* to use the SMB source for Linux authentication on your server.
5. Click *OK* and confirm the domain join when prompted for it.
6. Provide the password for the Windows Administrator on the AD server and click *OK*.

Your server is now set up to pull in all authentication data from the Active Directory domain controller.



Tip: Identity Mapping

In an environment with more than one Samba server, UIDs and GIDs will not be created consistently. The UIDs that get assigned to users will be dependent on the order in which they first log in, which results in UID conflicts across servers. To fix this, you need to use identity mapping. See <https://www.samba.org/samba/docs/man/Samba-HOWTO-Collection/idmapper.html> for more details.

21.8 Advanced Topics

This section introduces more advanced techniques to manage both the client and server part of the Samba suite.

21.8.1 Transparent File Compression on Btrfs

Samba allows clients to remotely manipulate file and directory compression flags for shares placed on the Btrfs file system. Windows Explorer provides the ability to flag files/directories for transparent compression via the *File > Properties > Advanced* dialog:

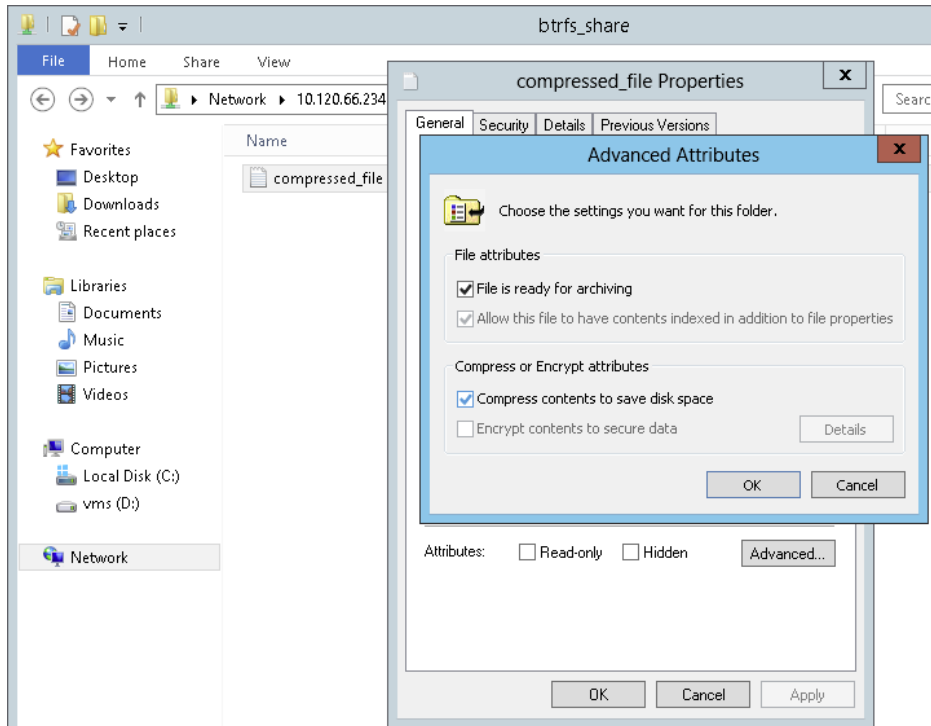


FIGURE 21.2: WINDOWS EXPLORER ADVANCED ATTRIBUTES DIALOG

Files flagged for compression are transparently compressed and decompressed by the underlying file system when accessed or modified. This normally results in storage capacity savings at the expense of extra CPU overhead when accessing the file. New files and directories inherit the compression flag from the parent directory, unless created with the `FILE_NO_COMPRESSION` option.

Windows Explorer presents compressed files and directories visually differently to those that are not compressed:

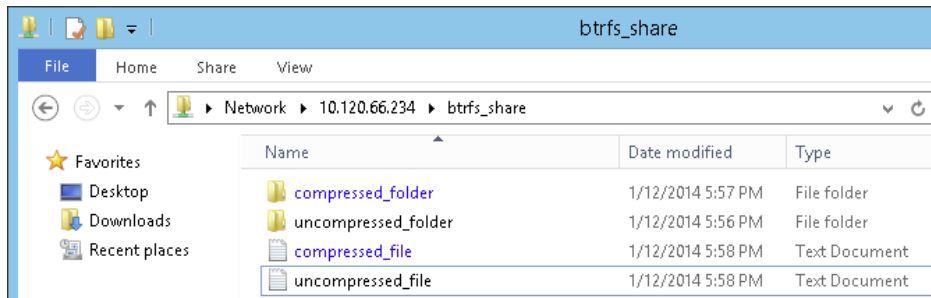


FIGURE 21.3: WINDOWS EXPLORER DIRECTORY LISTING WITH COMPRESSED FILES

You can enable Samba share compression either manually by adding

```
vfs objects = btrfs
```

to the share configuration in `/etc/samba/smb.conf`, or using YaST: *Network Services > Samba Server > Add*, and checking *Utilize Btrfs Features*.

21.8.2 Snapshots

Snapshots, also called Shadow Copies, are copies of the state of a file system subvolume at a certain point of time. Snapper is the tool to manage these snapshots in Linux. Snapshots are supported on the Btrfs file system or thinly-provisioned LVM volumes. The Samba suite supports managing remote snapshots through the FSRVP protocol on both the server and client side.

21.8.2.1 Previous Versions

Snapshots on a Samba server can be exposed to remote Windows clients as file or directory previous versions.

To enable snapshots on a Samba server, the following conditions must be fulfilled:

- The SMB network share resides on a Btrfs subvolume.
- The SMB network share path has a related snapper configuration file. You can create the snapper file with

```
tux > sudo snapper -c <cfg_name> create-config /path/to/share
```

For more information on snapper, see *Chapter 3, System Recovery and Snapshot Management with Snapper*.

- The snapshot directory tree must allow access for relevant users. For more information, see the PERMISSIONS section of the `vfs_snapper` manual page ([man 8 vfs_snapper](#)).

To support remote snapshots, you need to modify the `/etc/samba/smb.conf` file. You can do it either with *YaST > Network Services > Samba Server*, or manually by enhancing the relevant share section with

```
vfs objects = snapper
```

Note that you need to restart the Samba service for manual `smb.conf` changes to take effect:

```
tux > sudo systemctl restart nmb smb
```

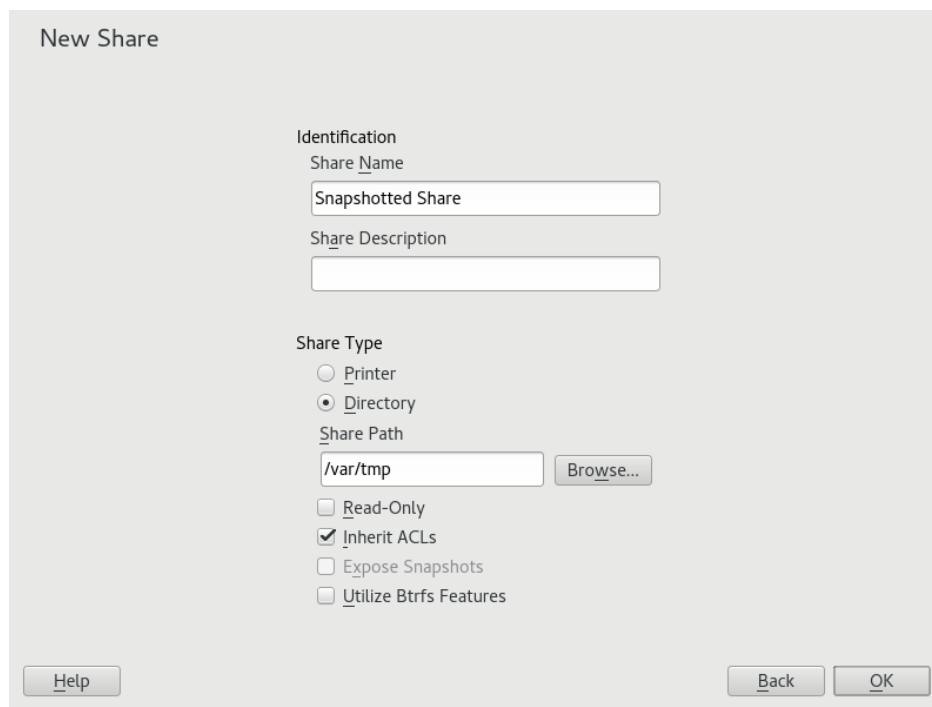


FIGURE 21.4: ADDING A NEW SAMBA SHARE WITH SNAPSHOTTING ENABLED

After being configured, snapshots created by snapper for the Samba share path can be accessed from Windows Explorer from a file or directory's *Previous Versions* tab.

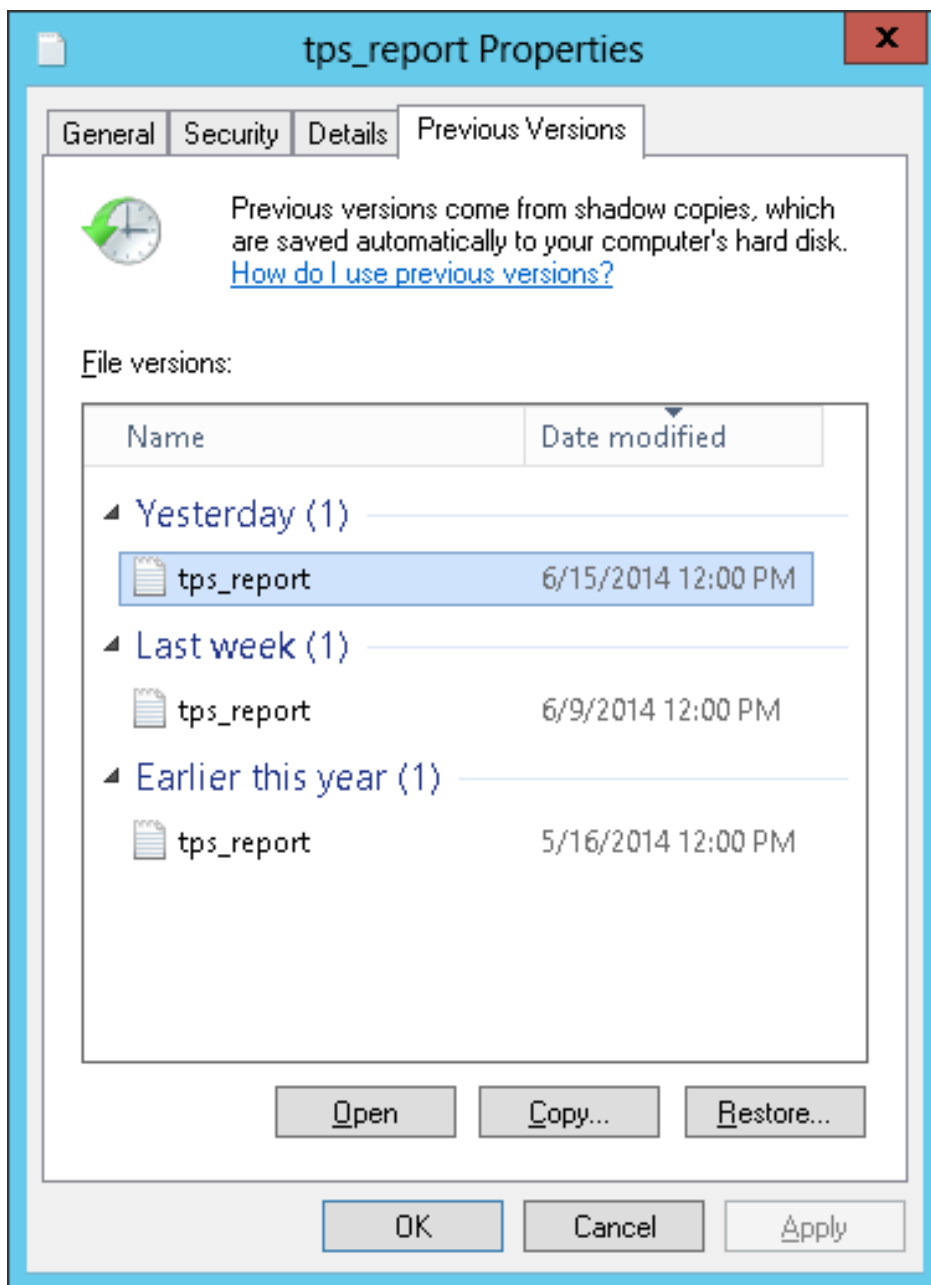


FIGURE 21.5: THE *PREVIOUS VERSIONS* TAB IN WINDOWS EXPLORER

21.8.2.2 Remote Share Snapshots

By default, snapshots can only be created and deleted on the Samba server locally, via the `snapper` command line utility, or using `snapper`'s time line feature.

Samba can be configured to process share snapshot creation and deletion requests from remote hosts using the File Server Remote VSS Protocol (FSRVP).

In addition to the configuration and prerequisites documented in [Section 21.8.2.1, “Previous Versions”](#), the following global configuration is required in `/etc/samba/smb.conf`:

```
[global]
rpc_daemon:fssd = fork
registry shares = yes
include = registry
```

FSRVP clients, including Samba's `rpcclient` and Windows Server 2012 `DiskShadow.exe`, can then instruct Samba to create or delete a snapshot for a given share, and expose the snapshot as a new share.

21.8.2.3 Managing Snapshots Remotely from Linux with `rpcclient`

The `samba-client` package contains an FSRVP client that can remotely request a Windows/Samba server to create and expose a snapshot of a given share. You can then use existing tools in openSUSE Leap to mount the exposed share and back up its files. Requests to the server are sent using the `rpcclient` binary.

EXAMPLE 21.4: USING `rpcclient` TO REQUEST A WINDOWS SERVER 2012 SHARE SNAPSHOT

Connect to `win-server.example.com` server as an administrator in an `EXAMPLE` domain:

```
root # rpcclient -U 'EXAMPLE\Administrator' ncacn_np:win-
server.example.com[ndr64,sign]
Enter EXAMPLE/Administrator's password:
```

Check that the SMB share is visible for `rpcclient`:

```
root # rpcclient $> netshareenum
netname: windows_server_2012_share
remark:
path: C:\Shares\windows_server_2012_share
password: (null)
```

Check that the SMB share supports snapshot creation:

```
root # rpcclient $> fss_is_path_sup windows_server_2012_share \
UNC \\WIN-SERVER\windows_server_2012_share\ supports shadow copy requests
```

Request the creation of a share snapshot:

```
root # rpcclient $> fss_create_expose backup ro windows_server_2012_share
13fe880e-e232-493d-87e9-402f21019fb6: shadow-copy set created
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
```



```
\\WIN-SERVER\windows_server_2012_share\ shadow-copy added to set
13fe880e-e232-493d-87e9-402f21019fb6: prepare completed in 0 secs
13fe880e-e232-493d-87e9-402f21019fb6: commit completed in 1 secs
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
share windows_server_2012_share@{1C26544E-8251-445F-BE89-D1E0A3938777} \
exposed as a snapshot of \\WIN-SERVER\windows_server_2012_share\
```

Confirm that the snapshot share is exposed by the server:

```
root # rpcclient $> netshareenum
netname: windows_server_2012_share
remark:
path: C:\Shares\windows_server_2012_share
password: (null)

netname: windows_server_2012_share@{1C26544E-8251-445F-BE89-D1E0A3938777}
remark: (null)
path: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy{F6E6507E-F537-11E3-9404-
B8AC6F927453}\Shares\windows_server_2012_share\
password: (null)
```

Attempt to delete the snapshot share:

```
root # rpcclient $> fss_delete windows_server_2012_share \
13fe880e-e232-493d-87e9-402f21019fb6 1c26544e-8251-445f-be89-d1e0a3938777
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
\\WIN-SERVER\windows_server_2012_share\ shadow-copy deleted
```

Confirm that the snapshot share has been removed by the server:

```
root # rpcclient $> netshareenum
netname: windows_server_2012_share
remark:
path: C:\Shares\windows_server_2012_share
password: (null)
```

21.8.2.4 Managing Snapshots Remotely from Windows with **DiskShadow.exe**

You can manage snapshots of SMB shares on the Linux Samba server from the Windows environment acting as a client as well. Windows Server 2012 includes the **DiskShadow.exe** utility that can manage remote shares similar to the **rpcclient** described in [Section 21.8.2.3](#), *“Managing Snapshots Remotely from Linux with **rpcclient**”*. Note that you need to carefully set up the Samba server first.

Following is an example procedure to set up the Samba server so that the Windows Server client can manage its share's snapshots. Note that *EXAMPLE* is the Active Directory domain used in the testing environment, `fsrvp-server.example.com` is the host name of the Samba server, and `/srv/smb` is the path to the SMB share.

PROCEDURE 21.1: DETAILED SAMBA SERVER CONFIGURATION

1. Join Active Directory domain via YaST. For more information, [Section 21.7, "Samba Server in the Network with Active Directory"](#).

2. Ensure that the Active Domain DNS entry was correct:

```
fsrvp-server:~ # net -U 'Administrator' ads dns register \  
fsrvp-server.example.com <IP address>  
Successfully registered hostname with DNS
```

3. Create Btrfs subvolume at `/srv/smb`

```
fsrvp-server:~ # btrfs subvolume create /srv/smb
```

4. Create snapper configuration file for path `/srv/smb`

```
fsrvp-server:~ # snapper -c <snapper_config> create-config /srv/smb
```

5. Create new share with path `/srv/smb`, and YaST *Expose Snapshots* check box enabled. Make sure to add the following snippets to the global section of `/etc/samba/smb.conf` as mentioned in [Section 21.8.2.2, "Remote Share Snapshots"](#):

```
[global]  
rpc_daemon:fssd = fork  
registry shares = yes  
include = registry
```

6. Restart Samba with `systemctl restart nmb smb`

7. Configure snapper permissions:

```
fsrvp-server:~ # snapper -c <snapper_config> set-config \  
ALLOW_USERS="EXAMPLE\\\\Administrator EXAMPLE\\\\win-client$"
```

Ensure that any `ALLOW_USERS` are also permitted traversal of the `.snapshots` subdirectory.

```
fsrvp-server:~ # snapper -c <snapper_config> set-config SYNC_ACL=yes
```

! Important: Path Escaping

Be careful about the `\` escapes! Escape twice to ensure that the value stored in `/etc/snapper/configs/<snapper_config>` is escaped once.

"EXAMPLE\win-client\$" corresponds to the Windows client computer account. Windows issues initial FSRVP requests while authenticated with this account.

8. Grant Windows client account necessary privileges:

```
fsrvp-server:~ # net -U 'Administrator' rpc rights grant \  
"EXAMPLE\\win-client$" SeBackupPrivilege  
Successfully granted rights.
```

The previous command is not needed for the "EXAMPLE\Administrator" user, which has privileges already granted.

PROCEDURE 21.2: WINDOWS CLIENT SETUP AND `DiskShadow.exe` IN ACTION

1. Boot Windows Server 2012 (example host name WIN-CLIENT).
2. Join the same Active Directory domain EXAMPLE as with the openSUSE Leap.
3. Reboot.
4. Open Powershell.
5. Start **`DiskShadow.exe`** and begin the backup procedure:

```
PS C:\Users\Administrator.EXAMPLE> diskshadow.exe  
Microsoft DiskShadow version 1.0  
Copyright (C) 2012 Microsoft Corporation  
On computer: WIN-CLIENT, 6/17/2014 3:53:54 PM  
  
DISKSHADOW> begin backup
```

6. Specify that shadow copy persists across program exit, reset or reboot:

```
DISKSHADOW> set context PERSISTENT
```

7. Check whether the specified share supports snapshots, and create one:

```
DISKSHADOW> add volume \\fsrvp-server\sles_snapper
```

```

DISKSHADOW> create
Alias VSS_SHADOW_1 for shadow ID {de4ddca4-4978-4805-8776-cdf82d190a4a} set as \
environment variable.
Alias VSS_SHADOW_SET for shadow set ID {c58e1452-c554-400e-a266-d11d5c837cb1} \
set as environment variable.

Querying all shadow copies with the shadow copy set ID \
{c58e1452-c554-400e-a266-d11d5c837cb1}

* Shadow copy ID = {de4ddca4-4978-4805-8776-cdf82d190a4a}      %VSS_SHADOW_1%
  - Shadow copy set: {c58e1452-c554-400e-a266-d11d5c837cb1} %VSS_SHADOW_SET%
  - Original count of shadow copies = 1
  - Original volume name: \\FSRVP-SERVER\SLES_SNAPPER\ \
    [volume not on this machine]
  - Creation time: 6/17/2014 3:54:43 PM
  - Shadow copy device name:
    \\FSRVP-SERVER\SLES_SNAPPER@{31afd84a-44a7-41be-b9b0-751898756faa}
  - Originating machine: FSRVP-SERVER
  - Service machine: win-client.example.com
  - Not exposed
  - Provider ID: {89300202-3cec-4981-9171-19f59559e0f2}
  - Attributes: No_Auto_Release Persistent FileShare

Number of shadow copies listed: 1

```

8. Finish the backup procedure:

```
DISKSHADOW> end backup
```

9. After the snapshot was created, try to delete it and verify the deletion:

```

DISKSHADOW> delete shadows volume \\FSRVP-SERVER\SLES_SNAPPER\
Deleting shadow copy {de4ddca4-4978-4805-8776-cdf82d190a4a} on volume \
\\FSRVP-SERVER\SLES_SNAPPER\ from provider \
{89300202-3cec-4981-9171-19f59559e0f2} [Attributes: 0x04000009]...

Number of shadow copies deleted: 1

DISKSHADOW> list shadows all

Querying all shadow copies on the computer ...
No shadow copies found in system.

```

21.9 For More Information

- **Man Pages:** To see a list of all man pages installed with the package `samba`, run `apropos samba`. Open any of the man pages with `man NAME_OF_MAN_PAGE`.
- **SUSE-specific README file:** The package `samba-client` contains the file `/usr/share/doc/packages/samba/README.SUSE`.
- **Additional Packaged Documentation:** Install the package `samba-doc` with `zypper install samba-doc`.
This documentation installs into `/usr/share/doc/packages/samba`. It contains an HTML version of the man pages and a library of example configurations (such as `smb.conf.SUSE`).
- **Online Documentation:** The Samba wiki contains extensive *User Documentation* at https://wiki.samba.org/index.php/User_Documentation.

22 Sharing File Systems with NFS

The *Network File System (NFS)* is a protocol that allows access to files on a server in a manner similar to accessing local files.

openSUSE Leap installs NFS v4.2, which introduces support for sparse files, file pre-allocation, server-side clone and copy, application data block (ADB), and labeled NFS for mandatory access control (MAC) (requires MAC on both client and server).

22.1 Overview

The *Network File System (NFS)* is a standardized, well-proven and widely supported network protocol that allows files to be shared between separate hosts.

The *Network Information Service (NIS)* can be used to have a centralized user management in the network. Combining NFS and NIS allows using file and directory permissions for access control in the network. NFS with NIS makes a network transparent to the user.

In the default configuration, NFS completely trusts the network and thus any machine that is connected to a trusted network. Any user with administrator privileges on any computer with physical access to any network the NFS server trusts can access any files that the server makes available.

Often, this level of security is perfectly satisfactory, such as when the network that is trusted is truly private, often localized to a single cabinet or machine room, and no unauthorized access is possible. In other cases the need to trust a whole subnet as a unit is restrictive and there is a need for more fine-grained trust. To meet the need in these cases, NFS supports various security levels using the *Kerberos* infrastructure. Kerberos requires NFSv4, which is used by default. For details, see *Book "Security and Hardening Guide", Chapter 7 "Network Authentication with Kerberos"*.

The following are terms used in the YaST module.

Exports

A directory *exported* by an NFS server, which clients can integrate into their systems.

NFS Client

The NFS client is a system that uses NFS services from an NFS server over the Network File System protocol. The TCP/IP protocol is already integrated into the Linux kernel; there is no need to install any additional software.

NFS Server

The NFS server provides NFS services to clients. A running server depends on the following daemons: `nfsd` (worker), `idmapd` (ID-to-name mapping for NFSv4, needed for certain scenarios only), `statd` (file locking), and `mountd` (mount requests).

NFSv3

NFSv3 is the version 3 implementation, the “old” stateless NFS that supports client authentication.

NFSv4

NFSv4 is the new version 4 implementation that supports secure user authentication via Kerberos. NFSv4 requires one single port only and thus is better suited for environments behind a firewall than NFSv3.

The protocol is specified as <http://tools.ietf.org/html/rfc3530>.

pNFS

Parallel NFS, a protocol extension of NFSv4. Any pNFS clients can directly access the data on an NFS server.



Important: Need for DNS

In principle, all exports can be made using IP addresses only. To avoid time-outs, you need a working DNS system. DNS is necessary at least for logging purposes, because the `mountd` daemon does reverse lookups.

22.2 Installing NFS Server

The NFS server is not part of the default installation. To install the NFS server using YaST, choose *Software > Software Management*, select *Patterns*, and enable the *File Server* option in the *Server Functions* section. Click *Accept* to install the required packages.

Like NIS, NFS is a client/server system. However, a machine can be both—it can supply file systems over the network (export) and mount file systems from other hosts (import).



Note: Mounting NFS Volumes Locally on the Exporting Server

Mounting NFS volumes locally on the exporting server is not supported on openSUSE Leap.

22.3 Configuring NFS Server

Configuring an NFS server can be done either through YaST or manually. For authentication, NFS can also be combined with Kerberos.

22.3.1 Exporting File Systems with YaST

With YaST, turn a host in your network into an NFS server—a server that exports directories and files to all hosts granted access to it or to all members of a group. Thus, the server can also provide applications without installing the applications locally on every host.

To set up such a server, proceed as follows:

PROCEDURE 22.1: SETTING UP AN NFS SERVER

1. Start YaST and select *Network Services > NFS Server*; see *Figure 22.1, “NFS Server Configuration Tool”*. You may be prompted to install additional software.

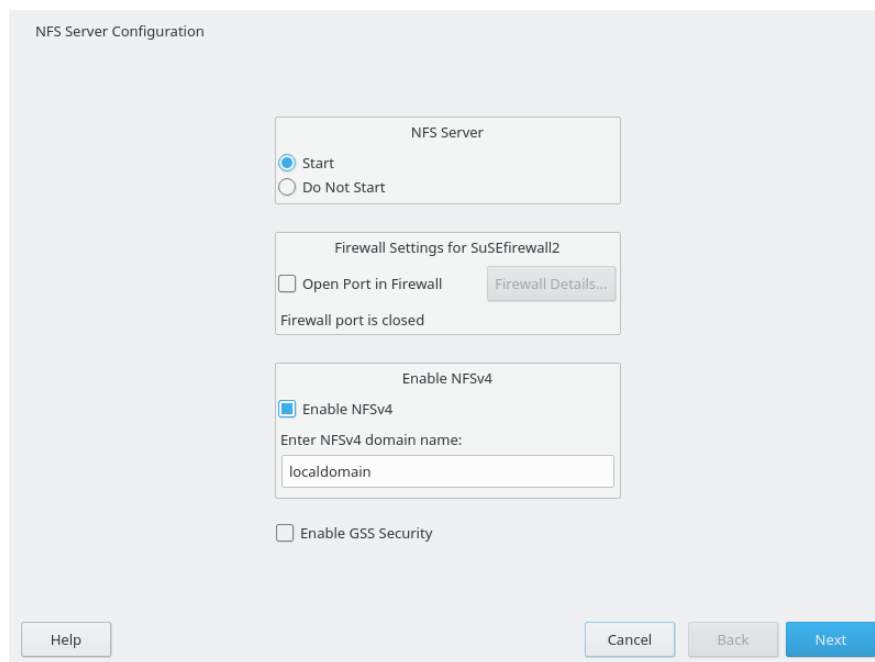


FIGURE 22.1: NFS SERVER CONFIGURATION TOOL

2. Click the *Start* radio button.

3. If `firewalld` is active on your system, configure it separately for NFS (see Book “Security and Hardening Guide”, Chapter 25 “Masquerading and Firewalls”, Section 25.4 “firewalld”). YaST does not yet have complete support for `firewalld`, so ignore the "Firewall not configurable" message and continue.
4. Check whether you want to *Enable NFSv4*. If you deactivate NFSv4, YaST will only support NFSv3. For information about enabling NFSv2, see *Note: NFSv2*.
 - If NFSv4 is selected, additionally enter the appropriate NFSv4 domain name. This parameter is used by the `idmapd` daemon that is required for Kerberos setups or if clients cannot work with numeric user names. Leave it as `localdomain` (the default) if you do not run `idmapd` or do not have any special requirements. For more information on the `idmapd` daemon see `/etc/idmapd.conf`.
5. Click *Enable GSS Security* if you need secure access to the server. A prerequisite for this is to have Kerberos installed on your domain and to have both the server and the clients kerberized. Click *Next* to proceed with the next configuration dialog.
6. Click *Add Directory* in the upper half of the dialog to export your directory.
7. If you have not configured the allowed hosts already, another dialog for entering the client information and options pops up automatically. Enter the host wild card (usually you can leave the default settings as they are).

There are four possible types of host wild cards that can be set for each host: a single host (name or IP address), netgroups, wild cards (such as `*` indicating all machines can access the server), and IP networks.

For more information about these options, see the `exports` man page.
8. Click *Finish* to complete the configuration.

22.3.2 Exporting File Systems Manually

The configuration files for the NFS export service are `/etc/exports` and `/etc/sysconfig/nfs`. In addition to these files, `/etc/idmapd.conf` is needed for the NFSv4 server configuration with kerberized NFS or if the clients cannot work with numeric user names.

To start or restart the services, run the command `systemctl restart nfsserver`. This also restarts the RPC portmapper that is required by the NFS server.

To make sure the NFS server always starts at boot time, run `sudo systemctl enable nfsserver`.



Note: NFSv4

NFSv4 is the latest version of NFS protocol available on openSUSE Leap. Configuring directories for export with NFSv4 is now the same as with NFSv3.

On openSUSE prior to Leap, the bind mount in `/etc/exports` was mandatory. It is still supported, but now deprecated.

`/etc/exports`

The `/etc/exports` file contains a list of entries. Each entry indicates a directory that is shared and how it is shared. A typical entry in `/etc/exports` consists of:

```
/SHARED/DIRECTORY HOST(OPTION_LIST)
```

For example:

```
/export/data 192.168.1.2(rw, sync)
```

Here the IP address `192.168.1.2` is used to identify the allowed client. You can also use the name of the host, a wild card indicating a set of hosts (`*.abc.com`, `*`, etc.), or netgroups (`@my-hosts`).

For a detailed explanation of all options and their meaning, refer to the man page of `/etc/exports` (**man exports**).

In case you have modified `/etc/exports` while the NFS server was running, you need to restart it for the changes to become active: **`sudo systemctl restart nfsserver`**.

`/etc/sysconfig/nfs`

The `/etc/sysconfig/nfs` file contains a few parameters that determine NFSv4 server daemon behavior. It is important to set the parameter `NFS4_SUPPORT` to `yes` (default). `NFS4_SUPPORT` determines whether the NFS server supports NFSv4 exports and clients.

In case you have modified `/etc/sysconfig/nfs` while the NFS server was running, you need to restart it for the changes to become active: **`sudo systemctl restart nfsserver`**.



Tip: Mount Options

On openSUSE prior to Leap, the `--bind` mount in `/etc/exports` was mandatory. It is still supported, but now deprecated. Configuring directories for export with NFSv4 is now the same as with NFSv3.



Note: NFSv2

If NFS clients still depend on NFSv2, enable it on the server in `/etc/sysconfig/nfs` by setting:

```
NFSD_OPTIONS="-V2"
MOUNTD_OPTIONS="-V2"
```

After restarting the service, check whether version 2 is available with the command:

```
tux > cat /proc/fs/nfsd/versions
+2 +3 +4 +4.1 +4.2
```

/etc/idmapd.conf

The `idmapd` daemon is only required if Kerberos authentication is used, or if clients cannot work with numeric user names. Linux clients can work with numeric user names since Linux kernel 2.6.39. The `idmapd` daemon does the name-to-ID mapping for NFSv4 requests to the server and replies to the client.

If required, `idmapd` needs to run on the NFSv4 server. Name-to-ID mapping on the client will be done by `nfsidmap` provided by the package `nfs-client`.

Make sure that there is a uniform way in which user names and IDs (UIDs) are assigned to users across machines that might probably be sharing file systems using NFS. This can be achieved by using NIS, LDAP, or any uniform domain authentication mechanism in your domain.

The parameter `Domain` must be set the same for both, client and server in the `/etc/idmapd.conf` file. If you are not sure, leave the domain as `localdomain` in the server and client files. A sample configuration file looks like the following:

```
[General]
Verbosity = 0
Pipefs-Directory = /var/lib/nfs/rpc_pipefs
Domain = localdomain

[Mapping]
Nobody-User = nobody
Nobody-Group = nobody
```

To start the `idmapd` daemon, run `systemctl start nfs-idmapd`. In case you have modified `/etc/idmapd.conf` while the daemon was running, you need to restart it for the changes to become active: `systemctl start nfs-idmapd`.

For more information, see the man pages of [idmapd](#) and [idmapd.conf](#) ([man idmapd](#) and [man idmapd.conf](#)).

22.3.3 NFS with Kerberos

To use Kerberos authentication for NFS, Generic Security Services (GSS) must be enabled. Select *Enable GSS Security* in the initial YaST NFS Server dialog. You must have a working Kerberos server to use this feature. YaST does not set up the server but only uses the provided functionality. To use Kerberos authentication in addition to the YaST configuration, complete at least the following steps before running the NFS configuration:

1. Make sure that both the server and the client are in the same Kerberos domain. They must access the same KDC (Key Distribution Center) server and share their [krb5.keytab](#) file (the default location on any machine is [/etc/krb5.keytab](#)). For more information about Kerberos, see *Book "Security and Hardening Guide", Chapter 7 "Network Authentication with Kerberos"*.
2. Start the gssd service on the client with `systemctl start rpc-gssd.service`.
3. Start the svcgssd service on the server with `systemctl start rpc-svcgssd.service`.

Kerberos authentication also requires the [idmapd](#) daemon to run on the server. For more information refer to [/etc/idmapd.conf](#).

For more information about configuring kerberized NFS, refer to the links in [Section 22.5, "For More Information"](#).

22.4 Configuring Clients

To configure your host as an NFS client, you do not need to install additional software. All needed packages are installed by default.

22.4.1 Importing File Systems with YaST

Authorized users can mount NFS directories from an NFS server into the local file tree using the YaST NFS client module. Proceed as follows:

PROCEDURE 22.2: IMPORTING NFS DIRECTORIES

1. Start the YaST NFS client module.
2. Click *Add* in the *NFS Shares* tab. Enter the host name of the NFS server, the directory to import, and the mount point at which to mount this directory locally.
3. When using NFSv4, select *Enable NFSv4* in the *NFS Settings* tab. Additionally, the *NFSv4 Domain Name* must contain the same value as used by the NFSv4 server. The default domain is `localdomain`.
4. To use Kerberos authentication for NFS, GSS security must be enabled. Select *Enable GSS Security*.
5. Enable *Open Port in Firewall* in the *NFS Settings* tab if you use a Firewall and want to allow access to the service from remote computers. The firewall status is displayed next to the check box.
6. Click *OK* to save your changes.

The configuration is written to `/etc/fstab` and the specified file systems are mounted. When you start the YaST configuration client at a later time, it also reads the existing configuration from this file.



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 13.4.1.2.5, "Activating the Network Device"](#) and choose *On NFSroot* in the *Device Activation* pane.

22.4.2 Importing File Systems Manually

The prerequisite for importing file systems manually from an NFS server is a running RPC port mapper. The `nfs` service takes care to start it properly; thus, start it by entering `systemctl start nfs` as `root`. Then remote file systems can be mounted in the file system like local partitions using `mount`:

```
tux > sudo mount HOST:REMOTE-PATHLOCAL-PATH
```

To import user directories from the `nfs.example.com` machine, for example, use:

```
tux > sudo mount nfs.example.com:/home /home
```

22.4.2.1 Using the Automount Service

The `autofs` daemon can be used to mount remote file systems automatically. Add the following entry to the `/etc/auto.master` file:

```
/nfsmounts /etc/auto.nfs
```

Now the `/nfsmounts` directory acts as the root for all the NFS mounts on the client if the `auto.nfs` file is filled appropriately. The name `auto.nfs` is chosen for the sake of convenience—you can choose any name. In `auto.nfs` add entries for all the NFS mounts as follows:

```
localdata -fstype=nfs server1:/data  
nfs4mount -fstype=nfs4 server2:/
```

Activate the settings with `systemctl start autofs` as `root`. In this example, `/nfsmounts/localdata`, the `/data` directory of `server1`, is mounted with NFS and `/nfsmounts/nfs4mount` from `server2` is mounted with NFSv4.

If the `/etc/auto.master` file is edited while the service `autofs` is running, the automounter must be restarted for the changes to take effect with `systemctl restart autofs`.

22.4.2.2 Manually Editing `/etc/fstab`

A typical NFSv3 mount entry in `/etc/fstab` looks like this:

```
nfs.example.com:/data /local/path nfs rw,noauto 0 0
```

For NFSv4 mounts, use `nfs4` instead of `nfs` in the third column:

```
nfs.example.com:/data /local/pathv4 nfs4 rw,noauto 0 0
```

The `noauto` option prevents the file system from being mounted automatically at start-up. If you want to mount the respective file system manually, it is possible to shorten the mount command specifying the mount point only:

```
tux > sudo mount /local/path
```



Note: Mounting at Start-Up

If you do not enter the `noauto` option, the init scripts of the system will handle the mount of those file systems at start-up.

22.4.3 Parallel NFS (pNFS)

NFS is one of the oldest protocols, developed in the '80s. As such, NFS is usually sufficient if you want to share small files. However, when you want to transfer big files or many clients want to access data, an NFS server becomes a bottleneck and has a significant impact on the system performance. This is because of files quickly getting bigger, whereas the relative speed of your Ethernet has not fully kept up.

When you request a file from a regular NFS server, the server looks up the file metadata, collects all the data and transfers it over the network to your client. However, the performance bottleneck becomes apparent no matter how small or big the files are:

- With small files most of the time is spent collecting the metadata.
- With big files most of the time is spent on transferring the data from server to client.

pNFS, or parallel NFS, overcomes this limitation as it separates the file system metadata from the location of the data. As such, pNFS requires two types of servers:

- A *metadata or control server* that handles all the non-data traffic
- One or more *storage server(s)* that hold(s) the data

The metadata and the storage servers form a single, logical NFS server. When a client wants to read or write, the metadata server tells the NFSv4 client which storage server to use to access the file chunks. The client can access the data directly on the server.

openSUSE Leap supports pNFS on the client side only.

22.4.3.1 Configuring pNFS Client With YaST

Proceed as described in *Procedure 22.2, "Importing NFS Directories"*, but click the *pNFS (v4.2)* check box and optionally *NFSv4 share*. YaST will do all the necessary steps and will write all the required options in the file `/etc/exports`.

22.4.3.2 Configuring pNFS Client Manually

Refer to *Section 22.4.2, "Importing File Systems Manually"* to start. Most of the configuration is done by the NFSv4 server. For pNFS, the only difference is to add the `minorversion` option and the metadata server `MDS_SERVER` to your `mount` command:

```
tux > sudo mount -t nfs4 -o minorversion=1 MDS_SERVER MOUNTPOINT
```

To help with debugging, change the value in the `/proc` file system:

```
tux > sudo echo 32767 > /proc/sys/sunrpc/nfsd_debug
tux > sudo echo 32767 > /proc/sys/sunrpc/nfs_debug
```

22.5 For More Information

In addition to the man pages of `exports`, `nfs`, and `mount`, information about configuring an NFS server and client is available in `/usr/share/doc/packages/nfsidmap/README`. For further documentation online refer to the following Web sites:

- Find the detailed technical documentation online at SourceForge (<http://nfs.sourceforge.net/>) ↗.
- For instructions for setting up kerberized NFS, refer to NFS Version 4 Open Source Reference Implementation (<http://www.citi.umich.edu/projects/nfsv4/linux/krb5-setup.html>) ↗.
- If you have questions on NFSv4, refer to the Linux NFSv4 FAQ (<http://www.citi.umich.edu/projects/nfsv4/linux/faq/>) ↗.

23 On-Demand Mounting with Autofs

`autofs` is a program that automatically mounts specified directories on an on-demand basis. It is based on a kernel module for high efficiency, and can manage both local directories and network shares. These automatic mount points are mounted only when they are accessed, and unmounted after a certain period of inactivity. This on-demand behavior saves bandwidth and results in better performance than static mounts managed by `/etc/fstab`. While `autofs` is a control script, `automount` is the command (daemon) that does the actual auto-mounting.

23.1 Installation

`autofs` is not installed on openSUSE Leap by default. To use its auto-mounting capabilities, first install it with

```
tux > sudo zypper install autofs
```

23.2 Configuration

You need to configure `autofs` manually by editing its configuration files with a text editor, such as `vim`. There are two basic steps to configure `autofs`—the *master* map file, and specific map files.

23.2.1 The Master Map File

The default master configuration file for `autofs` is `/etc/auto.master`. You can change its location by changing the value of the `DEFAULT_MASTER_MAP_NAME` option in `/etc/sysconfig/autofs`. Here is the content of the default one for openSUSE Leap:

```
#  
# Sample auto.master file  
# This is an automounter map and it has the following format
```

```

# key [ -mount-options-separated-by-comma ] location
# For details of the format look at autofs(5). ❶
#
#/misc /etc/auto.misc ❷
#/net -hosts
#
# Include /etc/auto.master.d/*.autofs ❸
#
#+dir:/etc/auto.master.d
#
# Include central master map if it can be found using
# nsswitch sources.
#
# Note that if there are entries for /net or /misc (as
# above) in the included master map any keys that are the
# same will not be seen as the first read key seen takes
# precedence.
#
+auto.master ❹

```

- ❶ The [autofs](#) manual page ([man 5 autofs](#)) offers a lot of valuable information on the format of the automounter maps.
- ❷ Although commented out (`#`) by default, this is an example of a simple automounter mapping syntax.
- ❸ In case you need to split the master map into several files, uncomment the line, and put the mappings (suffixed with `.autofs`) in the `/etc/auto.master.d/` directory.
- ❹ `+auto.master` ensures that those using NIS (see *Book "Security and Hardening Guide", Chapter 4 "Using NIS", Section 4.1 "Configuring NIS Servers"* for more information on NIS) will still find their master map.

Entries in `auto.master` have three fields with the following syntax:

| mount point | map name | options |
|-------------|----------|---------|
|-------------|----------|---------|

mount point

The base location where to mount the `autofs` file system, such as `/home`.

map name

The name of a map source to use for mounting. For the syntax of the maps files, see [Section 23.2.2, "Map Files"](#).

options

These options (if specified) will apply as defaults to all entries in the given map.



Tip: For More Information

For more detailed information on the specific values of the optional `map-type`, `format`, and `options`, see the `auto.master` manual page (`man 5 auto.master`).

The following entry in `auto.master` tells `autofs` to look in `/etc/auto.smb`, and create mount points in the `/smb` directory.

```
/smb /etc/auto.smb
```

23.2.1.1 Direct Mounts

Direct mounts create a mount point at the path specified inside the relevant map file. Instead of specifying the mount point in `auto.master`, replace the mount point field with `/-`. For example, the following line tells `autofs` to create a mount point at the place specified in `auto.smb`:

```
/- /etc/auto.smb
```



Tip: Maps without Full Path

If the map file is not specified with its full local or network path, it is located using the Name Service Switch (NSS) configuration:

```
/- auto.smb
```

23.2.2 Map Files



Important: Other Types of Maps

Although *files* are the most common types of maps for auto-mounting with `autofs`, there are other types as well. A map specification can be the output of a command, or a result of a query in LDAP or database. For more detailed information on map types, see the manual page `man 5 auto.master`.

Map files specify the (local or network) source location, and the mount point where to mount the source locally. The general format of maps is similar to the master map. The difference is that the *options* appear between the mount point and the location instead of at the end of the entry:

```
mount point      options      location
```

Make sure that map files are not marked as executable. You can remove the executable bits by executing `chmod -x MAP_FILE`.

mount point

Specifies where to mount the source location. This can be either a single directory name (so-called *indirect* mount) to be added to the base mount point specified in `auto.master`, or the full path of the mount point (direct mount, see [Section 23.2.1.1, "Direct Mounts"](#)).

options

Specifies optional comma-separated list of mount options for the relevant entries. If `auto.master` contains options for this map file as well, these are appended.

location

Specifies from where the file system is to be mounted. It is usually an NFS or SMB volume in the usual notation `host_name:path_name`. If the file system to be mounted begins with a '/' (such as local `/dev` entries or smbfs shares), a colon symbol ':' needs to be prefixed, such as `:/dev/sda1`.

23.3 Operation and Debugging

This section introduces information on how to control the `autofs` service operation, and how to view more debugging information when tuning the automounter operation.

23.3.1 Controlling the autofs Service

The operation of the `autofs` service is controlled by `systemd`. The general syntax of the `systemctl` command for `autofs` is

```
tux > sudo systemctl SUB_COMMAND autofs
```

where `SUB_COMMAND` is one of:

enable

Starts the automounter daemon at boot.

start

Starts the automounter daemon.

stop

Stops the automounter daemon. Automatic mount points are not accessible.

status

Prints the current status of the `autofs` service together with a part of a relevant log file.

restart

Stops and starts the automounter, terminating all running daemons and starting new ones.

reload

Checks the current `auto.master` map, restarts those daemons whose entries have changed, and starts new ones for new entries.

23.3.2 Debugging the Automounter Problems

If you experience problems when mounting directories with `autofs`, it is useful to run the `automount` daemon manually and watch its output messages:

1. Stop `autofs`.

```
tux > sudo systemctl stop autofs
```

2. From one terminal, run `automount` manually in the foreground, producing verbose output.

```
tux > sudo automount -f -v
```

3. From another terminal, try to mount the auto-mounting file systems by accessing the mount points (for example by `cd` or `ls`).
4. Check the output of `automount` from the first terminal for more information why the mount failed, or why it was not even attempted.

23.4 Auto-Mounting an NFS Share

The following procedure illustrates how to configure `autofs` to auto-mount an NFS share available on your network. It uses the information mentioned above, and assumes you are familiar with NFS exports. For more information on NFS, see [Chapter 22, Sharing File Systems with NFS](#).

1. Edit the master map file `/etc/auto.master`:

```
tux > sudo vim /etc/auto.master
```

Add a new entry for the new NFS mount at the end of `/etc/auto.master`:

```
/nfs      /etc/auto.nfs      --timeout=10
```

It tells `autofs` that the base mount point is `/nfs`, the NFS shares are specified in the `/etc/auto.nfs` map, and that all shares in this map will be automatically unmounted after 10 seconds of inactivity.

2. Create a new map file for NFS shares:

```
tux > sudo vim /etc/auto.nfs
```

`/etc/auto.nfs` normally contains a separate line for each NFS share. Its format is described in [Section 23.2.2, "Map Files"](#). Add the line describing the mount point and the NFS share network address:

```
export      jupiter.com:/home/geeko/doc/export
```

The above line means that the `/home/geeko/doc/export` directory on the `jupiter.com` host will be auto-mounted to the `/nfs/export` directory on the local host (`/nfs` is taken from the `auto.master` map) when requested. The `/nfs/export` directory will be created automatically by `autofs`.

3. Optionally comment out the related line in `/etc/fstab` if you previously mounted the same NFS share statically. The line should look similar to this:

```
#jupiter.com:/home/geeko/doc/export /nfs/export nfs defaults 0 0
```

4. Reload `autofs` and check if it works:

```
tux > sudo systemctl restart autofs
```

```
# ls -l /nfs/export
total 20
drwxr-xr-x  5 1001 users 4096 Jan 14  2017 .images/
drwxr-xr-x 10 1001 users 4096 Aug 16  2017 .profiled/
drwxr-xr-x  3 1001 users 4096 Aug 30  2017 .tmp/
drwxr-xr-x  4 1001 users 4096 Apr 25 08:56 manual/
```

If you can see the list of files on the remote share, then autofs is functioning.

23.5 Advanced Topics

This section describes topics that are beyond the basic introduction to autofs—auto-mounting of NFS shares that are available on your network, using wild cards in map files, and information specific to the CIFS file system.

23.5.1 /net Mount Point

This helper mount point is useful if you use a lot of NFS shares. /net auto-mounts all NFS shares on your local network on demand. The entry is already present in the auto.master file, so all you need to do is uncomment it and restart autofs:

```
/net    -hosts
```

```
tux > sudo systemctl restart autofs
```

For example, if you have a server named jupiter with an NFS share called /export, you can mount it by typing

```
tux > sudo cd /net/jupiter/export
```

on the command line.

23.5.2 Using Wild Cards to Auto-Mount Subdirectories

If you have a directory with subdirectories that you need to auto-mount individually—the typical case is the /home directory with individual users' home directories inside—autofs offers a clever solution for that.

In case of home directories, add the following line in `auto.master`:

```
/home      /etc/auto.home
```

Now you need to add the correct mapping to the `/etc/auto.home` file, so that the users' home directories are mounted automatically. One solution is to create separate entries for each directory:

```
wilber      jupiter.com:/home/wilber
penguin     jupiter.com:/home/penguin
tux         jupiter.com:/home/tux
[...]
```

This is very awkward as you need to manage the list of users inside `auto.home`. You can use the asterisk `*` instead of the mount point, and the ampersand `&` instead of the directory to be mounted:

```
*          jupiter:/home/&
```

23.5.3 Auto-Mounting CIFS File System

If you want to auto-mount an SMB/CIFS share (see [Chapter 21, Samba](#) for more information on the SMB/CIFS protocol), you need to modify the syntax of the map file. Add `-fstype=cifs` in the option field, and prefix the share location with a colon `!`.

```
mount point  -fstype=cifs      ://jupiter.com/export
```


24 The Apache HTTP Server

According to the survey from <http://www.netcraft.com/>, the Apache HTTP Server (Apache) is the world's most widely-used Web server. Developed by the Apache Software Foundation (<http://www.apache.org/>), it is available for most operating systems. openSUSE® Leap includes Apache version 2.4. In this chapter, learn how to install, configure and set up a Web server; how to use SSL, CGI, and additional modules; and how to troubleshoot Apache.

24.1 Quick Start

With this section, quickly set up and start Apache. You must be root to install and configure Apache.

24.1.1 Requirements

Make sure the following requirements are met before trying to set up the Apache Web server:

1. The machine's network is configured properly. For more information about this topic, refer to *Chapter 13, Basic Networking*.
2. The machine's exact system time is maintained by synchronizing with a time server. This is necessary because parts of the HTTP protocol depend on the correct time. See *Chapter 18, Time Synchronization with NTP* to learn more about this topic.
3. The latest security updates are installed. If in doubt, run a YaST Online Update.
4. The default Web server port (80) is opened in the firewall. For this, configure `firewalld` to allow the service `http` in the public zone. See *Book "Security and Hardening Guide", Chapter 25 "Masquerading and Firewalls", Section 25.4.1 "Configuring the Firewall on the Command Line"* for details.

24.1.2 Installation

Apache on openSUSE Leap is not installed by default. To install it with a standard, predefined configuration that runs “out of the box”, proceed as follows:

PROCEDURE 24.1: INSTALLING APACHE WITH THE DEFAULT CONFIGURATION

1. Start YaST and select *Software > Software Management*.
2. Choose *View > Patterns* and select *Web and LAMP Server*.
3. Confirm the installation of the dependent packages to finish the installation process.

24.1.3 Start

You can start Apache automatically at boot time or start it manually.

To make sure that Apache is automatically started during boot in the targets `multi-user.target` and `graphical.target`, execute the following command:

```
tux > sudo systemctl enable apache2
```

For more information about the `systemd` targets in openSUSE Leap and a description of the YaST *Services Manager*, refer to [Section 10.4, “Managing Services with YaST”](#).

To manually start Apache using the shell, run `systemctl start apache2`.

PROCEDURE 24.2: CHECKING IF APACHE IS RUNNING

If you do not receive error messages when starting Apache, this usually indicates that the Web server is running. To test this:

1. Start a browser and open <http://localhost/>.
If Apache is up and running, you get a test page stating “It works!”.
2. If you do not see this page, refer to [Section 24.9, “Troubleshooting”](#).

Now that the Web server is running, you can add your own documents, adjust the configuration according to your needs, or add functionality by installing modules.

24.2 Configuring Apache

openSUSE Leap offers two configuration options:

- *Configuring Apache Manually*
- *Configuring Apache with YaST*

Manual configuration offers a higher level of detail, but lacks the convenience of the YaST GUI.

Important: Reload or Restart Apache after Configuration Changes

Most configuration changes require a reload (some also a restart) of Apache to take effect. Manually reload Apache with `systemctl reload apache2` or use one of the restart options as described in *Section 24.3, “Starting and Stopping Apache”*.

If you configure Apache with YaST, this can be taken care of automatically if you set *HTTP Service* to *Enabled* as described in *Section 24.2.3.2, “HTTP Server Configuration”*.

24.2.1 Apache Configuration Files

This section gives an overview of the Apache configuration files. If you use YaST for configuration, you do not need to touch these files—however, the information might be useful for you to switch to manual configuration later on.

Apache configuration files can be found in two different locations:

- `/etc/sysconfig/apache2`
- `/etc/apache2/`

24.2.1.1 `/etc/sysconfig/apache2`

`/etc/sysconfig/apache2` controls some global settings of Apache, like modules to load, additional configuration files to include, flags with which the server should be started, and flags that should be added to the command line. Every configuration option in this file is extensively documented and therefore not mentioned here. For a general-purpose Web server, the settings in `/etc/sysconfig/apache2` should be sufficient for any configuration needs.

24.2.1.2 `/etc/apache2/`

`/etc/apache2/` hosts all configuration files for Apache. In the following, the purpose of each file is explained. Each file includes several configuration options (also called *directives*). Every configuration option in these files is extensively documented and therefore not mentioned here.

The Apache configuration files are organized as follows:

```
/etc/apache2/
|
|- charset.conv
|- conf.d/
| |
| |- *.conf
|
|- default-server.conf
|- errors.conf
|- httpd.conf
|- listen.conf
|- magic
|- mime.types
|- mod_*.conf
|- server-tuning.conf
|- ssl.*
|- ssl-global.conf
|- sysconfig.d
| |
| |- global.conf
| |- include.conf
| |- loadmodule.conf . .
|
|- uid.conf
|- vhosts.d
| |- *.conf
```

APACHE CONFIGURATION FILES IN /ETC/APACHE2/

charset.conv

Specifies which character sets to use for different languages. Do not edit this file.

conf.d/*.conf

Configuration files added by other modules. These configuration files can be included into your virtual host configuration where needed. See vhosts.d/vhost.template for examples. By doing so, you can provide different module sets for different virtual hosts.

default-server.conf

Global configuration for all virtual hosts with reasonable defaults. Instead of changing the values, overwrite them with a virtual host configuration.

errors.conf

Defines how Apache responds to errors. To customize these messages for all virtual hosts, edit this file. Otherwise overwrite these directives in your virtual host configurations.

httpd.conf

The main Apache server configuration file. Avoid changing this file. It primarily contains include statements and global settings. Overwrite global settings in the pertinent configuration files listed here. Change host-specific settings (such as document root) in your virtual host configuration.

listen.conf

Binds Apache to specific IP addresses and ports. Name-based virtual hosting is also configured here. For details, see [Section 24.2.2.1.1, "Name-Based Virtual Hosts"](#).

magic

Data for the mime_magic module that helps Apache automatically determine the MIME type of an unknown file. Do not change this file.

mime.types

MIME types known by the system (this actually is a link to /etc/mime.types). Do not edit this file. If you need to add MIME types not listed here, add them to mod_mime-defaults.conf.

mod_*.conf

Configuration files for the modules that are installed by default. Refer to [Section 24.4, "Installing, Activating, and Configuring Modules"](#) for details. Note that configuration files for optional modules reside in the directory conf.d.

server-tuning.conf

Contains configuration directives for the different MPMs (see [Section 24.4.4, "Multiprocessing Modules"](#)) and general configuration options that control Apache's performance. Properly test your Web server when making changes here.

ssl-global.conf and ssl.*

Global SSL configuration and SSL certificate data. Refer to [Section 24.6, "Setting Up a Secure Web Server with SSL"](#) for details.

sysconfig.d/*.conf

Configuration files automatically generated from `/etc/sysconfig/apache2`. Do not change any of these files—edit `/etc/sysconfig/apache2` instead. Do not put other configuration files in this directory.

uid.conf

Specifies under which user and group ID Apache runs. Do not change this file.

vhosts.d/*.conf

Your virtual host configuration should be located here. The directory contains template files for virtual hosts with and without SSL. Every file in this directory ending with `.conf` is automatically included in the Apache configuration. Refer to [Section 24.2.2.1, “Virtual Host Configuration”](#) for details.

24.2.2 Configuring Apache Manually

Configuring Apache manually involves editing plain text configuration files as user `root`.

24.2.2.1 Virtual Host Configuration

The term *virtual host* refers to Apache's ability to serve multiple universal resource identifiers (URIs) from the same physical machine. This means that several domains, such as `www.example.com` and `www.example.net`, are run by a single Web server on one physical machine.

It is common practice to use virtual hosts to save administrative effort (only a single Web server needs to be maintained) and hardware expenses (each domain does not require a dedicated server). Virtual hosts can be name based, IP based, or port based.

To list all existing virtual hosts, use the command `apache2ctl -S`. This outputs a list showing the default server and all virtual hosts together with their IP addresses and listening ports. Furthermore, the list also contains an entry for each virtual host showing its location in the configuration files.

Virtual hosts can be configured via YaST as described in [Section 24.2.3.1.4, “Virtual Hosts”](#) or by manually editing a configuration file. By default, Apache in openSUSE Leap is prepared for one configuration file per virtual host in `/etc/apache2/vhosts.d/`. All files in this directory with the extension `.conf` are automatically included to the configuration. A basic template for a virtual host is provided in this directory (`vhost.template` or `vhost-ssl.template` for a virtual host with SSL support).



Tip: Always Create a Virtual Host Configuration

It is recommended to always create a virtual host configuration file, even if your Web server only hosts one domain. By doing so, you not only have the domain-specific configuration in one file, but you can always fall back to a working basic configuration by simply moving, deleting, or renaming the configuration file for the virtual host. For the same reason, you should also create separate configuration files for each virtual host. When using name-based virtual hosts it is recommended to set up a default configuration that will be used when a domain name does not match a virtual host configuration. The default virtual host is the one whose configuration is loaded first. Since the order of the configuration files is determined by file name, start the file name of the default virtual host configuration with an underscore character (_) to make sure it is loaded first (for example: _default_vhost.conf).

The `<VirtualHost>` `</VirtualHost>` block holds the information that applies to a particular domain. When Apache receives a client request for a defined virtual host, it uses the directives enclosed in this section. Almost all directives can be used in a virtual host context. See <http://httpd.apache.org/docs/2.4/mod/quickreference.html> for further information about Apache's configuration directives.

24.2.2.1.1 Name-Based Virtual Hosts

With name-based virtual hosts, more than one Web site is served per IP address. Apache uses the host field in the HTTP header that is sent by the client to connect the request to a matching `ServerName` entry of one of the virtual host declarations. If no matching `ServerName` is found, the first specified virtual host is used as a default.

The first step is to create a `<VirtualHost>` block for each different name-based host that you want to serve. Inside each `<VirtualHost>` block, you will need at minimum a `ServerName` directive to designate which host is served and a `DocumentRoot` directive to show where in the file system the content for that host resides.

EXAMPLE 24.1: BASIC EXAMPLES OF NAME-BASED `VirtualHost` ENTRIES

```
<VirtualHost *:80>
# This first-listed virtual host is also the default for *:80
ServerName www.example.com
ServerAlias example.com
DocumentRoot /srv/www/htdocs/domain
```

```
</VirtualHost>

<VirtualHost *:80>
ServerName other.example.com
DocumentRoot /srv/www/htdocs/otherdomain
</VirtualHost>
```

The opening `VirtualHost` tag takes the IP address (or fully qualified domain name) as an argument in a name-based virtual host configuration. A port number directive is optional.

The wild card `*` is also allowed as a substitute for the IP address. When using IPv6 addresses, the address must be included in square brackets.

EXAMPLE 24.2: NAME-BASED `VirtualHost` DIRECTIVES

```
<VirtualHost 192.168.3.100:80>
...
</VirtualHost>

<VirtualHost 192.168.3.100>
...
</VirtualHost>

<VirtualHost *:80>
...
</VirtualHost>

<VirtualHost *>
...
</VirtualHost>

<VirtualHost [2002:c0a8:364::]>
...
</VirtualHost>
```

24.2.2.1.2 IP-Based Virtual Hosts

This alternative virtual host configuration requires the setup of multiple IP addresses for a machine. One instance of Apache hosts several domains, each of which is assigned a different IP. The physical server must have one IP address for each IP-based virtual host. If the machine does not have multiple network cards, virtual network interfaces (IP aliasing) can also be used.

The following example shows Apache running on a machine with the IP `192.168.3.100`, hosting two domains on the additional IP addresses `192.168.3.101` and `192.168.3.102`. A separate `VirtualHost` block is needed for every virtual server.

EXAMPLE 24.3: IP-BASED VirtualHost DIRECTIVES

```
<VirtualHost 192.168.3.101>
...
</VirtualHost>

<VirtualHost 192.168.3.102>
...
</VirtualHost>
```

Here, `VirtualHost` directives are only specified for interfaces other than `192.168.3.100`. When a `Listen` directive is also configured for `192.168.3.100`, a separate IP-based virtual host must be created to answer HTTP requests to that interface—otherwise the directives found in the default server configuration (`/etc/apache2/default-server.conf`) are applied.

24.2.2.1.3 Basic Virtual Host Configuration

At least the following directives should be in each virtual host configuration to set up a virtual host. See `/etc/apache2/vhosts.d/vhost.template` for more options.

ServerName

The fully qualified domain name under which the host should be addressed.

DocumentRoot

Path to the directory from which Apache should serve files for this host. For security reasons, access to the entire file system is forbidden by default, so you must explicitly unlock this directory within a `Directory` container.

ServerAdmin

E-mail address of the server administrator. This address is, for example, shown on error pages Apache creates.

ErrorLog

The error log file for this virtual host. Although it is not necessary to create separate error log files for each virtual host, it is common practice to do so, because it makes the debugging of errors much easier. `/var/log/apache2/` is the default directory for Apache's log files.

CustomLog

The access log file for this virtual host. Although it is not necessary to create separate access log files for each virtual host, it is common practice to do so, because it allows the separate analysis of access statistics for each host. `/var/log/apache2/` is the default directory for Apache's log files.

As mentioned above, access to the whole file system is forbidden by default for security reasons. Therefore, explicitly unlock the directories in which you have placed the files Apache should serve—for example the `DocumentRoot` :

```
<Directory "/srv/www/www.example.com/htdocs">
  Require all granted
</Directory>
```



Note: Require all granted

In previous versions of Apache, the statement `Require all granted` was expressed as:

```
Order allow,deny
Allow from all
```

This old syntax is still supported by the `mod_access_compat` module.

The complete configuration file looks like this:

EXAMPLE 24.4: BASIC VirtualHost CONFIGURATION

```
<VirtualHost 192.168.3.100>
  ServerName www.example.com
  DocumentRoot /srv/www/www.example.com/htdocs
  ServerAdmin webmaster@example.com
  ErrorLog /var/log/apache2/www.example.com_log
  CustomLog /var/log/apache2/www.example.com-access_log common
  <Directory "/srv/www/www.example.com/htdocs">
    Require all granted
  </Directory>
</VirtualHost>
```

24.2.3 Configuring Apache with YaST

To configure your Web server with YaST, start YaST and select *Network Services > HTTP Server*. When starting the module for the first time, the *HTTP Server Wizard* starts, prompting you to make a few basic decisions concerning administration of the server. After having finished the wizard, the *HTTP Server Configuration* dialog starts each time you call the *HTTP Server* module. For more information, see [Section 24.2.3.2, “HTTP Server Configuration”](#).

24.2.3.1 HTTP Server Wizard

The HTTP Server Wizard consists of five steps. In the last step of the dialog, you may enter the expert configuration mode to make even more specific settings.

24.2.3.1.1 Network Device Selection

Here, specify the network interfaces and ports Apache uses to listen for incoming requests. You can select any combination of existing network interfaces and their respective IP addresses. Ports from all three ranges (well-known ports, registered ports, and dynamic or private ports) that are not reserved by other services can be used. The default setting is to listen on all network interfaces (IP addresses) on port 80.

Check *Open Port In Firewall* to open the ports in the firewall that the Web server listens on. This is necessary to make the Web server available on the network, which can be a LAN, WAN, or the public Internet. Keeping the port closed is only useful in test situations where no external access to the Web server is necessary. If you have multiple network interfaces, click *Firewall Details* to specify on which interface(s) the port(s) should be opened.

Click *Next* to continue with the configuration.

24.2.3.1.2 Modules

The *Modules* configuration option allows for the activation or deactivation of the script languages that the Web server should support. For the activation or deactivation of other modules, refer to [Section 24.2.3.2.2, “Server Modules”](#). Click *Next* to advance to the next dialog.

24.2.3.1.3 Default Host

This option pertains to the default Web server. As explained in [Section 24.2.2.1, "Virtual Host Configuration"](#), Apache can serve multiple virtual hosts from a single physical machine. The first declared virtual host in the configuration file is commonly called the *default host*. Each virtual host inherits the default host's configuration.

To edit the host settings (also called *directives*), select the appropriate entry in the table then click *Edit*. To add new directives, click *Add*. To delete a directive, select it and click *Delete*.

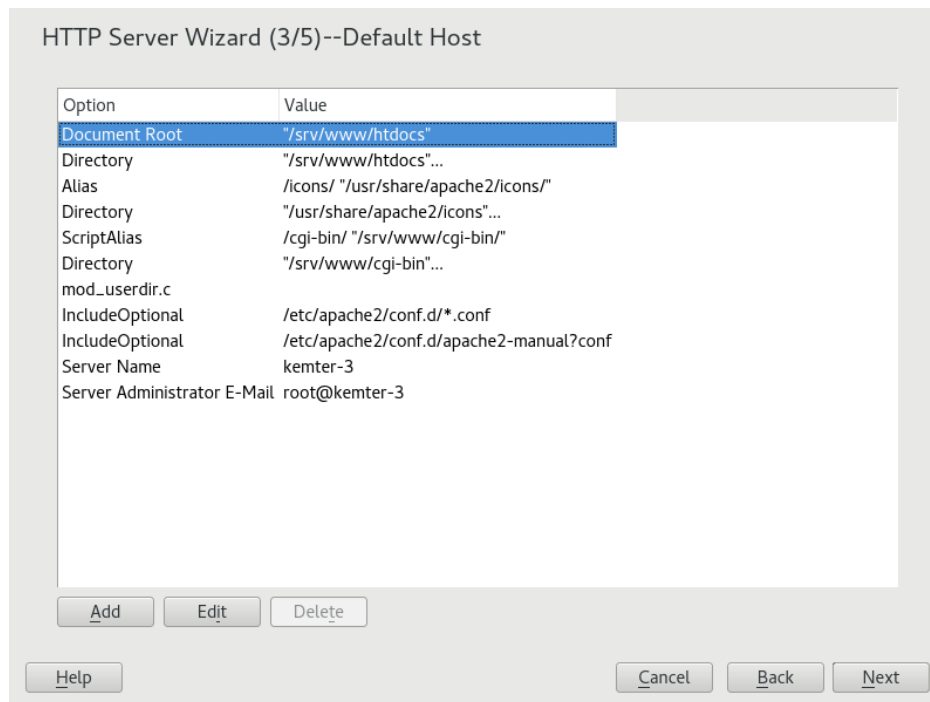


FIGURE 24.1: HTTP SERVER WIZARD: DEFAULT HOST

Here is list of the default settings of the server:

Document Root

Path to the directory from which Apache serves files for this host. `/srv/www/htdocs` is the default location.

Alias

Using Alias directives, URLs can be mapped to physical file system locations. This means that a certain path even outside the Document Root in the file system can be accessed via a URL aliasing that path.

The default openSUSE Leap Alias `/icons` points to `/usr/share/apache2/icons` for the Apache icons displayed in the directory index view.

ScriptAlias

Similar to the Alias directive, the ScriptAlias directive maps a URL to a file system location. The difference is that ScriptAlias designates the target directory as a CGI location, meaning that CGI scripts should be executed in that location.

Directory

With Directory settings, you can enclose a group of configuration options that will only apply to the specified directory.

Access and display options for the directories /srv/www/htdocs, /usr/share/apache2/icons and /srv/www/cgi-bin are configured here. It should not be necessary to change the defaults.

Include

With include, additional configuration files can be specified. Two Include directives are already preconfigured: /etc/apache2/conf.d/ is the directory containing the configuration files that come with external modules. With this directive, all files in this directory ending in .conf are included. With the second directive, /etc/apache2/conf.d/apache2-manual.conf, the apache2-manual configuration file is included.

Server Name

This specifies the default URL used by clients to contact the Web server. Use a fully qualified domain name (FQDN) to reach the Web server at http://FQDN/ or its IP address. You cannot choose an arbitrary name here—the server must be “known” under this name.

Server Administrator E-Mail

E-mail address of the server administrator. This address is, for example, shown on error pages Apache creates.

After finishing with the *Default Host* step, click *Next* to continue with the configuration.

24.2.3.1.4 Virtual Hosts

In this step, the wizard displays a list of already configured virtual hosts (see [Section 24.2.2.1, “Virtual Host Configuration”](#)). If you have not made manual changes prior to starting the YaST HTTP wizard, no virtual host is present.

To add a host, click *Add* to open a dialog in which to enter basic information about the host, such as *Server Name*, *Server Contents Root* (DocumentRoot), and the *Administrator E-Mail*. *Server Resolution* is used to determine how a host is identified (name based or IP based). Specify the name or IP address with *Change Virtual Host ID*

Clicking *Next* advances to the second part of the virtual host configuration dialog.

In part two of the virtual host configuration you can specify whether to enable CGI scripts and which directory to use for these scripts. It is also possible to enable SSL. If you do so, you must specify the path to the certificate as well. See [Section 24.6.2, “Configuring Apache with SSL”](#) for details on SSL and certificates. With the *Directory Index* option, you can specify which file to display when the client requests a directory (by default, `index.html`). Add one or more file names (space-separated) to change this. With *Enable Public HTML*, the content of the users public directories (`~USER/public_html/`) is made available on the server under `http://www.example.com/~USER`.

Important: Creating Virtual Hosts

It is not possible to add virtual hosts at will. If using name-based virtual hosts, each host name must be resolved on the network. If using IP-based virtual hosts, you can assign only one host to each IP address available.

24.2.3.1.5 Summary

This is the final step of the wizard. Here, determine how and when the Apache server is started: when booting or manually. Also see a short summary of the configuration made so far. If you are satisfied with your settings, click *Finish* to complete configuration. To change something, click *Back* until you have reached the desired dialog. Clicking *HTTP Server Expert Configuration* opens the dialog described in [Section 24.2.3.2, “HTTP Server Configuration”](#).

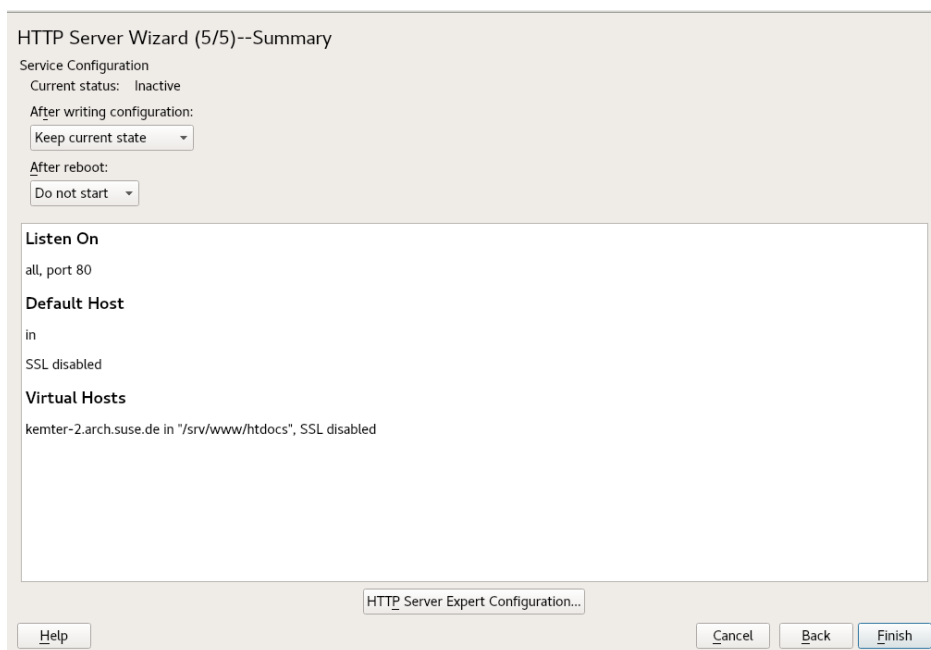


FIGURE 24.2: HTTP SERVER WIZARD: SUMMARY

24.2.3.2 HTTP Server Configuration

The *HTTP Server Configuration* dialog also lets you make even more adjustments to the configuration than the wizard (which only runs if you configure your Web server for the first time). It consists of four tabs described in the following. No configuration option you change here is effective immediately—you always must confirm your changes with *Finish* to make them effective. Clicking *Abort* leaves the configuration module and discards your changes.

24.2.3.2.1 Listen Ports and Addresses

In *HTTP Service*, select whether Apache should be running (*Enabled*) or stopped (*Disabled*). In *Listen on Ports*, *Add*, *Edit*, or *Delete* addresses and ports on which the server should be available. The default is to listen on all interfaces on port 80. You should always check *Open Port In Firewall*, because otherwise the Web server is not reachable from outside. Keeping the port closed is only useful in test situations where no external access to the Web server is necessary. If you have multiple network interfaces, click *Firewall Details* to specify on which interface(s) the port(s) should be opened.

With *Log Files*, watch either the access log file or the error log file. This is useful if you want to test your configuration. The log file opens in a separate window from which you can also restart or reload the Web server. For details, see [Section 24.3, “Starting and Stopping Apache”](#). These commands are effective immediately and their log messages are also displayed immediately.

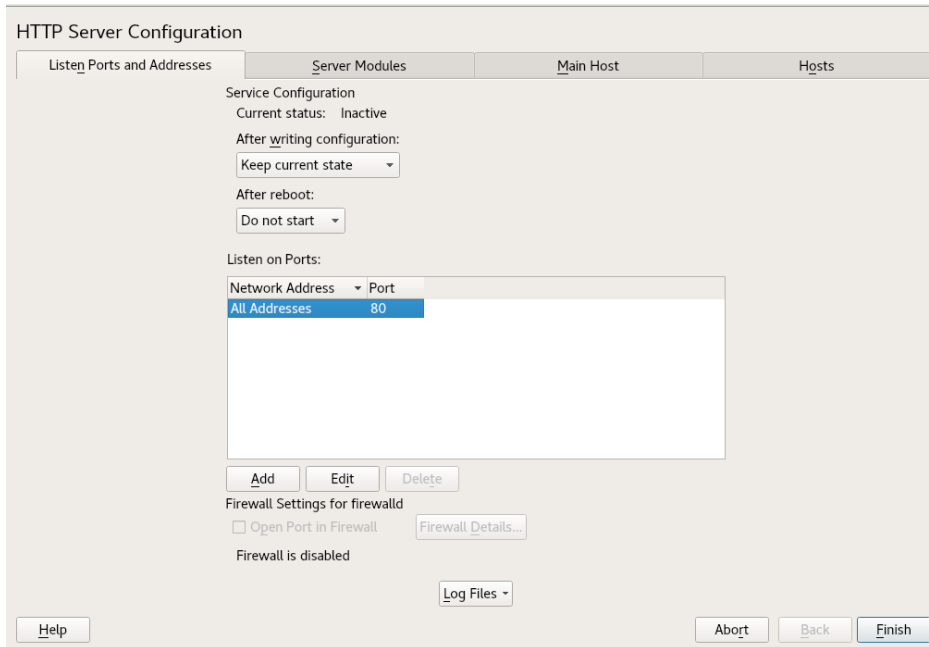


FIGURE 24.3: HTTP SERVER CONFIGURATION: LISTEN PORTS AND ADDRESSES

24.2.3.2.2 Server Modules

You can change the status (enabled or disabled) of Apache2 modules by clicking *Toggle Status*. Click *Add Module* to add a new module that is already installed but not yet listed. Learn more about modules in [Section 24.4, “Installing, Activating, and Configuring Modules”](#).

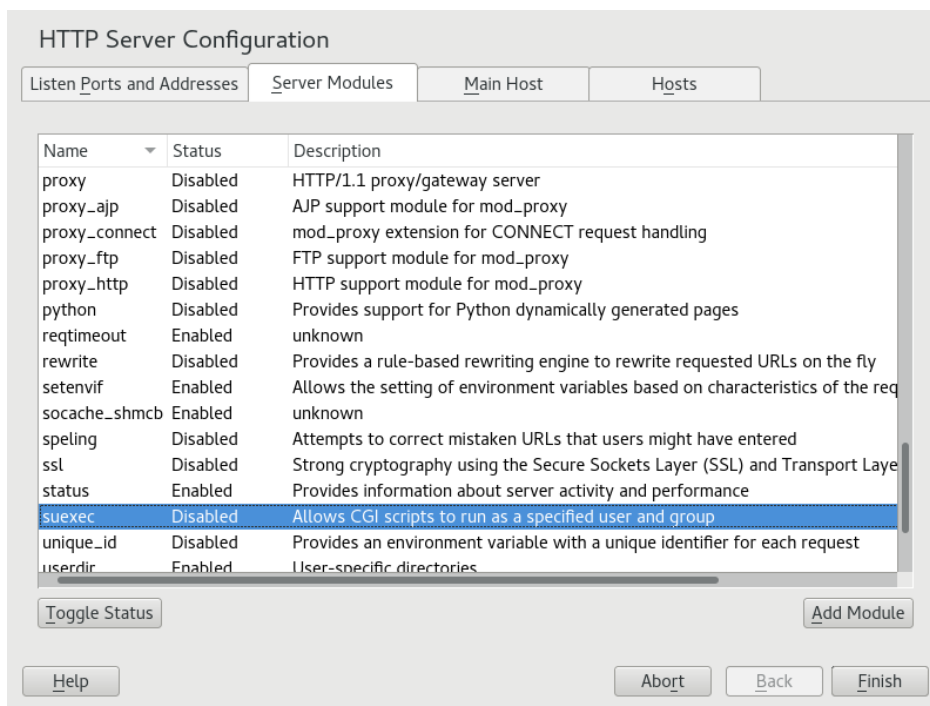


FIGURE 24.4: HTTP SERVER CONFIGURATION: SERVER MODULES

24.2.3.2.3 Main Host or Hosts

These dialogs are identical to the ones already described. Refer to *Section 24.2.3.1.3, “Default Host”* and *Section 24.2.3.1.4, “Virtual Hosts”*.

24.3 Starting and Stopping Apache

If configured with YaST as described in *Section 24.2.3, “Configuring Apache with YaST”*, Apache is started at boot time in the `multi-user.target` and `graphical.target`. You can change this behavior using YaST's *Services Manager* or with the `systemctl` command line tool (`systemctl enable` or `systemctl disable`).

To start, stop, or manipulate Apache on a running system, use either the `systemctl` or the `apachectl` commands as described below.

For general information about `systemctl` commands, refer to *Section 10.2.1, “Managing Services in a Running System”*.

`systemctl status apache2`

Checks if Apache is started.

systemctl start apache2

Starts Apache if it is not already running.

systemctl stop apache2

Stops Apache by terminating the parent process.

systemctl restart apache2

Stops and then restarts Apache. Starts the Web server if it was not running before.

systemctl try-restart apache2

Stops then restarts Apache only if it is already running.

systemctl reload apache2

Stops the Web server by advising all forked Apache processes to first finish their requests before shutting down. As each process dies, it is replaced by a newly started one, resulting in a complete “restart” of Apache.



Tip: Restarting Apache in Production Environments

This command allows activating changes in the Apache configuration without causing connection break-offs.

systemctl stop apache2

Stops the Web server after a defined period of time configured with GracefulShutdownTimeout to ensure that existing requests can be finished.

apachectl configtest

Checks the syntax of the configuration files without affecting a running Web server. Because this check is forced every time the server is started, reloaded, or restarted, it is usually not necessary to run the test explicitly (if a configuration error is found, the Web server is not started, reloaded, or restarted).

apachectl status and apachectl fullstatus

Dumps a short or full status screen, respectively. Requires the module mod_status to be enabled and a text-based browser (such as links or w3m) to be installed. In addition to that, STATUS must be added to APACHE_SERVER_FLAGS in the file /etc/sysconfig/apache2.



Tip: Additional Flags

If you specify additional flags to the commands, these are passed through to the Web server.

24.4 Installing, Activating, and Configuring Modules

The Apache software is built in a modular fashion: all functionality except some core tasks are handled by modules. This has progressed so far that even HTTP is processed by a module (`http_core`).

Apache modules can be compiled into the Apache binary at build time or be dynamically loaded at runtime. Refer to *Section 24.4.2, "Activation and Deactivation"* for details of how to load modules dynamically.

Apache modules can be divided into four different categories:

Base Modules

Base modules are compiled into Apache by default. Apache in openSUSE Leap has only `mod_so` (needed to load other modules) and `http_core` compiled in. All others are available as shared objects: rather than being included in the server binary itself, they can be included at runtime.

Extension Modules

In general, modules labeled as extensions are included in the Apache software package, but are usually not compiled into the server statically. In openSUSE Leap, they are available as shared objects that can be loaded into Apache at runtime.

External Modules

Modules labeled external are not included in the official Apache distribution. However, openSUSE Leap provides several of them.

Multiprocessing Modules (MPMs)

MPMs are responsible for accepting and handling requests to the Web server, representing the core of the Web server software.

24.4.1 Module Installation

If you have done a default installation as described in [Section 24.1.2, “Installation”](#), the following modules are already installed: all base and extension modules, the multiprocessing module Prefork MPM, and the external module `mod_python`.

You can install additional external modules by starting YaST and choosing *Software > Software Management*. Now choose *View > Search* and search for `apache`. Among other packages, the results list contains all available external Apache modules.

24.4.2 Activation and Deactivation

Activate or deactivate particular modules either manually or with YaST. In YaST, script language modules (PHP 5 and Python) need to be enabled or disabled with the module configuration described in [Section 24.2.3.1, “HTTP Server Wizard”](#). All other modules can be enabled or disabled as described in [Section 24.2.3.2.2, “Server Modules”](#).

If you prefer to activate or deactivate the modules manually, use the commands `a2enmod MODULE` or `a2dismod MODULE`, respectively. `a2enmod -l` outputs a list of all currently active modules.



Important: Including Configuration Files for External Modules

If you have activated external modules manually, make sure to load their configuration files in all virtual host configurations. Configuration files for external modules are located under `/etc/apache2/conf.d/` and are loaded in `/etc/apache2/default-server.conf` by default. For more fine-grained control you can comment out the inclusion in `/etc/apache2/default-server.conf` and add it to specific virtual hosts only. See `/etc/apache2/vhosts.d/vhost.template` for examples.

24.4.3 Base and Extension Modules

All base and extension modules are described in detail in the Apache documentation. Only a brief description of the most important modules is available here. Refer to <http://httpd.apache.org/docs/2.4/mod/> to learn details about each module.

`mod_actions`

Provides methods to execute a script whenever a certain MIME type (such as application/pdf), a file with a specific extension (like .rpm), or a certain request method (such as GET) is requested. This module is enabled by default.

mod_alias

Provides Alias and Redirect directives with which you can map a URL to a specific directory (Alias) or redirect a requested URL to another location. This module is enabled by default.

mod_auth*

The authentication modules provide different authentication methods: basic authentication with mod_auth_basic or digest authentication with mod_auth_digest. mod_auth_basic and mod_auth_digest must be combined with an authentication provider module, mod_authn_* (for example, mod_authn_file for text file-based authentication) and with an authorization module mod_authz_* (for example, mod_authz_user for user authorization).

More information about this topic is available in the *Authentication HOWTO* at <http://httpd.apache.org/docs/2.4/howto/auth.html>.

mod_auth_openidc

mod_auth_openidc the only certified way to use OpenID Connect with the Apache HTTP server. (See <https://openid.net/developers/certified/>.)

mod_autoindex

Autoindex generates directory listings when no index file (for example, index.html) is present. The look and feel of these indexes is configurable. This module is enabled by default. However, directory listings are disabled by default via the Options directive—overwrite this setting in your virtual host configuration. The default configuration file for this module is located at /etc/apache2/mod_autoindex-defaults.conf.

mod_cgi

mod_cgi is needed to execute CGI scripts. This module is enabled by default.

mod_deflate

Using this module, Apache can be configured to compress given file types on the fly before delivering them.

mod_dir

mod_dir provides the DirectoryIndex directive with which you can configure which files are automatically delivered when a directory is requested (index.html by default). It also provides an automatic redirect to the correct URL when a directory request does not contain a trailing slash. This module is enabled by default.

mod_env

Controls the environment that is passed to CGI scripts or SSI pages. Environment variables can be set or unset or passed from the shell that invoked the httpd process. This module is enabled by default.

mod_expires

With mod_expires, you can control how often proxy and browser caches refresh your documents by sending an Expires header. This module is enabled by default.

mod_http2

With mod_http2, Apache gains support for the HTTP/2 protocol. It can be enabled by specifying Protocols h2 http/1.1 in a VirtualHost.

mod_include

mod_include lets you use Server Side Includes (SSI), which provide a basic functionality to generate HTML pages dynamically. This module is enabled by default.

mod_info

Provides a comprehensive overview of the server configuration under http://localhost/server-info/. For security reasons, you should always limit access to this URL. By default only localhost is allowed to access this URL. mod_info is configured at /etc/apache2/mod_info.conf.

mod_log_config

With this module, you can configure the look of the Apache log files. This module is enabled by default.

mod_mime

The mime module ensures that a file is delivered with the correct MIME header based on the file name's extension (for example text/html for HTML documents). This module is enabled by default.

mod_negotiation

Necessary for content negotiation. See <http://httpd.apache.org/docs/2.4/content-negotiation.html> for more information. This module is enabled by default.

mod_rewrite

Provides the functionality of mod_alias, but offers more features and flexibility. With mod_rewrite, you can redirect URLs based on multiple rules, request headers, and more.

mod_setenvif

Sets environment variables based on details of the client's request, such as the browser string the client sends, or the client's IP address. This module is enabled by default.

mod_spelling

mod_spelling attempts to automatically correct typographical errors in URLs, such as capitalization errors.

mod_ssl

Enables encrypted connections between Web server and clients. See *Section 24.6, "Setting Up a Secure Web Server with SSL"* for details. This module is enabled by default.

mod_status

Provides information on server activity and performance under `http://localhost/server-status/`. For security reasons, you should always limit access to this URL. By default, only localhost is allowed to access this URL. mod_status is configured at /etc/apache2/mod_status.conf.

mod_suexec

mod_suexec lets you run CGI scripts under a different user and group. This module is enabled by default.

mod_userdir

Enables user-specific directories available under ~USER/. The UserDir directive must be specified in the configuration. This module is enabled by default.

24.4.4 Multiprocessing Modules

openSUSE Leap provides two different multiprocessing modules (MPMs) for use with Apache:

- *Prefork MPM*
- *Worker MPM*

24.4.4.1 Prefork MPM

The prefork MPM implements a non-threaded, preforking Web server. It makes the Web server behave similarly to Apache version 1.x. In this version it isolates each request and handles it by forking a separate child process. Thus problematic requests cannot affect others, avoiding a lockup of the Web server.

While providing stability with this process-based approach, the prefork MPM consumes more system resources than its counterpart, the worker MPM. The prefork MPM is considered the default MPM for Unix-based operating systems.



Important: MPMs in This Document

This document assumes Apache is used with the prefork MPM.

24.4.4.2 Worker MPM

The worker MPM provides a multi-threaded Web server. A thread is a “lighter” form of a process. The advantage of a thread over a process is its lower resource consumption. Instead of only forking child processes, the worker MPM serves requests by using threads with server processes. The preforked child processes are multi-threaded. This approach makes Apache perform better by consuming fewer system resources than the prefork MPM.

One major disadvantage is the stability of the worker MPM: if a thread becomes corrupt, all threads of a process can be affected. In the worst case, this may result in a server crash. Especially when using the Common Gateway Interface (CGI) with Apache under heavy load, internal server errors might occur because of threads being unable to communicate with system resources. Another argument against using the worker MPM with Apache is that not all available Apache modules are thread-safe and thus cannot be used with the worker MPM.



Warning: Using PHP Modules with MPMs

Not all available PHP modules are thread-safe. Using the worker MPM with `mod_php` is strongly discouraged.

24.4.5 External Modules

Find a list of all external modules shipped with openSUSE Leap here. Find the module's documentation in the listed directory.

mod_apparmor

Adds support to Apache to provide AppArmor confinement to individual CGI scripts handled by modules like mod_php5.

Package Name: apache2-mod_apparmor

More Information: *Book "Security and Hardening Guide"*

mod_php5

PHP is a server-side, cross-platform HTML embedded scripting language.

Package Name: apache2-mod_php5

Configuration File: /etc/apache2/conf.d/php5.conf

More Information: /usr/share/doc/packages/apache2-mod_php5

mod_python

mod_python allows embedding Python within the Apache HTTP server for a considerable boost in performance and added flexibility in designing Web-based applications.

Package Name: apache2-mod_python

More Information: /usr/share/doc/packages/apache2-mod_python

mod_security

mod_security provides a Web application firewall to protect Web applications from a range of attacks. It also enables HTTP traffic monitoring and real-time analysis.

Package Name: apache2-mod_security2

Configuration File: /etc/apache2/conf.d/mod_security2.conf

More Information: /usr/share/doc/packages/apache2-mod_security2

Documentation: <http://modsecurity.org/documentation/> ↗

24.4.6 Compilation

Apache can be extended by advanced users by writing custom modules. To develop modules for Apache or compile third-party modules, the package `apache2-devel` is required along with the corresponding development tools. `apache2-devel` also contains the `apxs2` tools, which are necessary for compiling additional modules for Apache.

`apxs2` enables the compilation and installation of modules from source code (including the required changes to the configuration files), which creates *dynamic shared objects* (DSOs) that can be loaded into Apache at runtime.

The `apxs2` binaries are located under `/usr/sbin`:

- `/usr/sbin/apxs2`—suitable for building an extension module that works with any MPM. The installation location is `/usr/lib64/apache2`.
- `/usr/sbin/apxs2-prefork`—suitable for prefork MPM modules. The installation location is `/usr/lib64/apache2-prefork`.
- `/usr/sbin/apxs2-worker`—suitable for worker MPM modules. The installation location is `/usr/lib64/apache2-worker`.

Install and activate a module from source code with the following commands:

```
tux > sudo cd /path/to/module/source
tux > sudo apxs2 -cia MODULE.c
```

where `-c` compiles the module, `-i` installs it, and `-a` activates it. Other options of `apxs2` are described in the `apxs2(1)` man page.

24.5 Enabling CGI Scripts

Apache's Common Gateway Interface (CGI) lets you create dynamic content with programs or scripts usually called CGI scripts. CGI scripts can be written in any programming language. Usually, script languages such as PHP are used.

To enable Apache to deliver content created by CGI scripts, `mod_cgi` needs to be activated. `mod_alias` is also needed. Both modules are enabled by default. Refer to [Section 24.4.2, “Activation and Deactivation”](#) for details on activating modules.



Warning: CGI Security

Allowing the server to execute CGI scripts is a potential security hole. Refer to [Section 24.8, “Avoiding Security Problems”](#) for additional information.

24.5.1 Apache Configuration

In openSUSE Leap, the execution of CGI scripts is only allowed in the directory `/srv/www/cgi-bin/`. This location is already configured to execute CGI scripts. If you have created a virtual host configuration (see [Section 24.2.2.1, “Virtual Host Configuration”](#)) and want to place your scripts in a host-specific directory, you must unlock and configure this directory.

EXAMPLE 24.5: VIRTUALHOST CGI CONFIGURATION

```
ScriptAlias /cgi-bin/ "/srv/www/www.example.com/cgi-bin/" ❶

<Directory "/srv/www/www.example.com/cgi-bin/">
  Options +ExecCGI ❷
  AddHandler cgi-script .cgi .pl ❸
  Require all granted ❹
</Directory>
```

- ❶ Tells Apache to handle all files within this directory as CGI scripts.
- ❷ Enables CGI script execution
- ❸ Tells the server to treat files with the extensions `.pl` and `.cgi` as CGI scripts. Adjust according to your needs.
- ❹ The `Require` directive controls the default access state. In this case, access is granted to the specified directory without limitation. For more information on authentication and authorization, see <http://httpd.apache.org/docs/2.4/howto/auth.html>.

24.5.2 Running an Example Script

CGI programming differs from "regular" programming in that the CGI programs and scripts must be preceded by a MIME-Type header such as `Content-type: text/html`. This header is sent to the client, so it understands what kind of content it receives. Secondly, the script's output must be something the client, usually a Web browser, understands—HTML usually, or plain text or images, for example.

A simple test script available under `/usr/share/doc/packages/apache2/test-cgi` is part of the Apache package. It outputs the content of some environment variables as plain text. Copy this script to either `/srv/www/cgi-bin/` or the script directory of your virtual host (`/srv/www/www.example.com/cgi-bin/`) and name it `test.cgi`. Edit the file to have `#!/bin/sh` as the first line.

Files accessible by the Web server should be owned by the user `root`. For additional information see [Section 24.8, “Avoiding Security Problems”](#). Because the Web server runs with a different user, the CGI scripts must be world-executable and world-readable. Change into the CGI directory and use the command `chmod 755 test.cgi` to apply the proper permissions.

Now call `http://localhost/cgi-bin/test.cgi` or `http://www.example.com/cgi-bin/test.cgi`. You should see the “CGI/1.0 test script report”.

24.5.3 CGI Troubleshooting

If you do not see the output of the test program but an error message instead, check the following:

CGI TROUBLESHOOTING

- *Have you reloaded the server after having changed the configuration?* If not, reload with `systemctl reload apache2`.
- *If you have configured your custom CGI directory, is it configured properly?* If in doubt, try the script within the default CGI directory `/srv/www/cgi-bin/` and call it with `http://localhost/cgi-bin/test.cgi`.
- *Are the file permissions correct?* Change into the CGI directory and execute `ls -l test.cgi`. The output should start with

```
-rwxr-xr-x 1 root root
```

- Make sure that the script does not contain programming errors. If you have not changed `test.cgi`, this should not be the case, but if you are using your own programs, always make sure that they do not contain programming errors.

24.6 Setting Up a Secure Web Server with SSL

Whenever sensitive data, such as credit card information, is transferred between Web server and client, it is desirable to have a secure, encrypted connection with authentication. `mod_ssl` provides strong encryption using the secure sockets layer (SSL) and transport layer security (TLS) protocols for HTTP communication between a client and the Web server. Using TLS/SSL, a private connection between Web server and client is established. Data integrity is ensured and client and server can authenticate each other.

For this purpose, the server sends an SSL certificate that holds information proving the server's valid identity before any request to a URL is answered. In turn, this guarantees that the server is the uniquely correct end point for the communication. Additionally, the certificate generates an encrypted connection between client and server that can transport information without the risk of exposing sensitive, plain-text content.

`mod_ssl` does not implement the TLS/SSL protocols itself, but acts as an interface between Apache and an SSL library. In openSUSE Leap, the OpenSSL library is used. OpenSSL is automatically installed with Apache.

The most visible effect of using `mod_ssl` with Apache is that URLs are prefixed with `https://` instead of `http://`.

24.6.1 Creating an SSL Certificate

To use TLS/SSL with the Web server, you need to create an SSL certificate. This certificate is needed for the authorization between Web server and client, so that each party can clearly identify the other party. To ensure the integrity of the certificate, it must be signed by a party every user trusts.

There are three types of certificates you can create: a “dummy” certificate for testing purposes only, a self-signed certificate for a defined circle of users that trust you, and a certificate signed by an independent, publicly-known certificate authority (CA).

Creating a certificate is a two step process. First, a private key for the certificate authority is generated then the server certificate is signed with this key.



Tip: For More Information

To learn more about concepts and definitions of TLS/SSL, refer to http://httpd.apache.org/docs/2.4/ssl/ssl_intro.html ↗.

24.6.1.1 Creating a “Dummy” Certificate

To generate a dummy certificate, call the script `/usr/bin/gensslcert`. It creates or overwrites the files listed below. Use `gensslcert`'s optional switches to fine-tune the certificate. Call `/usr/bin/gensslcert -h` for more information.

- `/etc/apache2/ssl.crt/ca.crt`
- `/etc/apache2/ssl.crt/server.crt`
- `/etc/apache2/ssl.key/server.key`
- `/etc/apache2/ssl.csr/server.csr`

A copy of `ca.crt` is also placed at `/srv/www/htdocs/CA.crt` for download.



Important: For Testing Purposes Only

A dummy certificate should never be used on a production system. Only use it for testing purposes.

24.6.1.2 Creating a Self-Signed Certificate

If you are setting up a secure Web server for an intranet or for a defined circle of users, it is probably sufficient if you sign a certificate with your own certificate authority (CA). Note that visitors to such a site will see a warning like “this is an untrusted site”, as Web browsers do not recognize self-signed certificates.



Important: Self-Signed Certificates

Only use a self-signed certificate on a Web server that is accessed by people who know and trust you as a certificate authority. It is not recommended to use such a certificate for a public shop, for example.

First you need to generate a certificate signing request (CSR). You are going to use `openssl`, with `PEM` as the certificate format. During this step, you will be asked for a passphrase, and to answer several questions. Remember the passphrase you enter as you will need it in the future.

```
tux > sudo openssl req -new > new.cert.csr
Generating a 1024 bit RSA private key
..++++++
```

```

.....+++++
writing new private key to 'privkey.pem'
Enter PEM pass phrase: ①
Verifying - Enter PEM pass phrase: ②
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]: ③
State or Province Name (full name) [Some-State]: ④
Locality Name (eg, city) []: ⑤
Organization Name (eg, company) [Internet Widgits Pty Ltd]: ⑥
Organizational Unit Name (eg, section) []: ⑦
Common Name (for example server FQDN, or YOUR name) []: ⑧
Email Address []: ⑨

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: ⑩
An optional company name []: ⑪

```

- ① Fill in your passphrase,
- ② ...fill it in once more (and remember it).
- ③ Fill in your 2 letter country code, such as GB or CZ.
- ④ Fill in the name of the state where you live.
- ⑤ Fill in the city name, such as Prague.
- ⑥ Fill in the name of the organization you work for.
- ⑦ Fill in your organization unit, or leave blank if you have none.
- ⑧ Fill in either the domain name of the server, or your first and last name.
- ⑨ Fill in your work e-mail address.
- ⑩ Leave the challenge password empty, otherwise you will need to enter it every time you restart the Apache Web server.
- ⑪ Fill in the optional company name, or leave blank.

Now you can generate the certificate. You are going to use openssl again, and the format of the certificate is the default PEM.

PROCEDURE 24.3: GENERATING THE CERTIFICATE

1. Export the private part of the key to `new.cert.key`. You will be prompted for the passphrase you entered when creating the certificate signing request (CSR).

```
tux > sudo openssl rsa -in privkey.pem -out new.cert.key
```

2. Generate the public part of the certificate according to the information you filled out in the signing request. The `-days` option specifies the length of time before the certificate expires. You can revoke a certificate, or replace one before it expires.

```
tux > sudo openssl x509 -in new.cert.csr -out new.cert.cert -req \  
-signkey new.cert.key -days 365
```

3. Copy the certificate files to the relevant directories, so that the Apache server can read them. Make sure that the private key `/etc/apache2/ssl.key/server.key` is not world-readable, while the public PEM certificate `/etc/apache2/ssl.crt/server.crt` is.

```
tux > sudo cp new.cert.cert /etc/apache2/ssl.crt/server.crt  
tux > sudo cp new.cert.key /etc/apache2/ssl.key/server.key
```



Tip: Public Certificate Location

The last step is to copy the public certificate file from `/etc/apache2/ssl.crt/server.crt` to a location where your users can access it to incorporate it into the list of known and trusted CAs in their Web browsers. Otherwise a browser complains that the certificate was issued by an unknown authority.

24.6.1.3 Getting an Officially Signed Certificate

There are several official certificate authorities that sign your certificates. The certificate is signed by a trustworthy third party, so can be fully trusted. Publicly operating secure Web servers usually have an officially signed certificate. A list of the most used Certificate Authorities (CAs) is available at https://en.wikipedia.org/wiki/Certificate_authority#Providers.

When requesting an officially signed certificate, you do not send a certificate to the CA. Instead, issue a Certificate Signing Request (CSR). To create a CSR, run the following command:

```
tux > openssl req -new -newkey rsa:2048 -nodes -keyout newkey.pem -out newreq.pem
```


You are asked to enter a distinguished name. This requires you to answer a few questions, such as country name or organization name. Enter valid data—everything you enter here later shows up in the certificate and is checked. You do not need to answer every question. If one does not apply to you or you want to leave it blank, use “.”. Common name is the name of the CA itself—choose a significant name, such as *My company* CA. Last, a challenge password and an alternative company name must be entered.

Find the CSR in the directory from which you called the script. The file is named `newreq.pem`.

24.6.2 Configuring Apache with SSL

The default port for TLS/SSL requests on the Web server side is 443. There is no conflict between a “regular” Apache listening on port 80 and an TLS/SSL-enabled Apache listening on port 443. In fact, HTTP and HTTPS can be run with the same Apache instance. Usually separate virtual hosts are used to dispatch requests to port 80 and port 443 to separate virtual servers.

Important: Firewall Configuration

Do not forget to open the firewall for SSL-enabled Apache on port 443. This can be done with `firewalld` as described in *Book “Security and Hardening Guide”, Chapter 25 “Masquerading and Firewalls”, Section 25.4.1 “Configuring the Firewall on the Command Line”*.

The SSL module is enabled by default in the global server configuration. In case it has been disabled on your host, activate it with the following command: `a2enmod ssl`. To finally enable SSL, the server needs to be started with the flag “SSL”. To do so, call `a2enflag SSL` (case-sensitive!). If you have chosen to encrypt your server certificate with a password, you should also increase the value for `APACHE_TIMEOUT` in `/etc/sysconfig/apache2`, so you have enough time to enter the passphrase when Apache starts. Restart the server to make these changes active. A reload is not sufficient.

The virtual host configuration directory contains a template `/etc/apache2/vhosts.d/vhost-ssl.template` with SSL-specific directives that are extensively documented. Refer to [Section 24.2.2.1, “Virtual Host Configuration”](#) for the general virtual host configuration.

To get started, copy the template to `/etc/apache2/vhosts.d/MYSSL-HOST.conf` and edit it. Adjusting the values for the following directives should be sufficient:

- DocumentRoot
- ServerName
- ServerAdmin
- ErrorLog
- TransferLog

24.6.2.1 Name-Based Virtual Hosts and SSL

By default it is not possible to run multiple SSL-enabled virtual hosts on a server with only one IP address. Name-based virtual hosting requires that Apache knows which server name has been requested. The problem with SSL connections is, that such a request can only be read after the SSL connection has already been established (by using the default virtual host). As a result, users will receive a warning message stating that the certificate does not match the server name.

openSUSE Leap comes with an extension to the SSL protocol called Server Name Indication (SNI) addresses this issue by sending the name of the virtual domain as part of the SSL negotiation. This enables the server to “switch” to the correct virtual domain early and present the browser the correct certificate.

SNI is enabled by default on openSUSE Leap. To enable Name-Based Virtual Hosts for SSL, configure the server as described in *Section 24.2.2.1.1, “Name-Based Virtual Hosts”* (note that you need to use port `443` rather than port `80` with SSL).

Important: SNI Browser Support

SNI must also be supported on the client side. However, SNI is supported by most browsers, except for certain older browsers. For more information, see https://en.wikipedia.org/wiki/Server_Name_Indication#Support.

To configure handling of non-SNI capable browsers, use the directive `SSLStrictSNIVHostCheck`. When set to `on` in the server configuration, non-SNI capable browser will be rejected for all virtual hosts. When set to `on` within a `VirtualHost` directive, access to this particular host will be rejected.

When set to `off` in the server configuration, the server will behave as if not having SNI support. SSL requests will be handled by the *first* virtual host defined (for port 443).

24.7 Running Multiple Apache Instances on the Same Server

As of openSUSE® Leap 42.1, you can run multiple Apache instances on the same server. This has several advantages over running multiple virtual hosts (see [Section 24.2.2.1, “Virtual Host Configuration”](#)):

- When a virtual host needs to be disabled for some time, you need to change the Web server configuration and restart it so that the change takes effect.
- In case of problems with one virtual host, you need to restart all of them.

You can run the default Apache instance as usual:

```
tux > sudo systemctl start apache2
```

It reads the default `/etc/sysconfig/apache2` file. If the file is not present, or it is present but it does not set the `APACHE_HTTPD_CONF` variable, it reads `/etc/apache2/httpd.conf`.

To activate another Apache instance, run:

```
tux > sudo systemctl start apache2@INSTANCE_NAME
```

For example:

```
tux > sudo systemctl start apache2@example_web.org
```

By default, the instance uses `/etc/apache2@example_web.org/httpd.conf` as a main configuration file, which can be overwritten by setting `APACHE_HTTPD_CONF` in `/etc/sysconfig/apache2@example_web.org`.

An example to set up an additional instance of Apache follows. Note that you need to execute all the commands as `root`.

1. Create a new configuration file based on `/etc/sysconfig/apache2`, for example `/etc/sysconfig/apache2@example_web.org`:

```
tux > sudo cp /etc/sysconfig/apache2 /etc/sysconfig/apache2@example_web.org
```

2. Edit the file `/etc/sysconfig/apache2@example_web.org` and change the line containing

```
APACHE_HTTPD_CONF
```

to

```
APACHE_HTTPD_CONF="/etc/apache2/httpd@example_web.org.conf"
```

3. Create the file `/etc/apache2/httpd@example_web.org.conf` based on `/etc/apache2/httpd.conf`.

```
tux > sudo cp /etc/apache2/httpd.conf /etc/apache2/httpd@example_web.org.conf
```

4. Edit `/etc/apache2/httpd@example_web.org.conf` and change

```
Include /etc/apache2/listen.conf
```

to

```
Include /etc/apache2/listen@example_web.org.conf
```

Review all the directives and change them to fit your needs. You will probably want to change

```
Include /etc/apache2/global.conf
```

and create new `global@example_web.org.conf` for each instance. We suggest to change

```
ErrorLog /var/log/apache2/error_log
```

to

```
ErrorLog /var/log/apache2/error@example_web.org_log
```

to have separate logs for each instance.

5. Create `/etc/apache2/listen@example_web.org.conf` based on `/etc/apache2/listen.conf`.

```
tux > sudo cp /etc/apache2/listen.conf /etc/apache2/listen@example_web.org.conf
```

6. Edit `/etc/apache2/listen@example_web.org.conf` and change

```
Listen 80
```

to the port number you want the new instance to run on, for example 82:

```
Listen 82
```

To run the new Apache instance over a secured protocol (see [Section 24.6, "Setting Up a Secure Web Server with SSL"](#)), change also the line

```
Listen 443
```

for example to

```
Listen 445
```

7. Start the new Apache instance:

```
tux > sudo systemctl start apache2@example_web.org
```

8. Check if the server is running by pointing your Web browser at `http://server_name:82`. If you previously changed the name of the error log file for the new instance, you can check it:

```
tux > sudo tail -f /var/log/apache2/error@example_web.org_log
```

Here are several points to consider when setting up more Apache instances on the same server:

- The file `/etc/sysconfig/apache2@INSTANCE_NAME` can include the same variables as `/etc/sysconfig/apache2`, including module loading and MPM setting.
- The default Apache instance does not need to be running while other instances run.
- The Apache helper utilities `a2enmod`, `a2dismod` and `apachectl` operate on the default Apache instance if not specified otherwise with the `HTTPD_INSTANCE` environment variable. The following example

```
tux > sudo export HTTPD_INSTANCE=example_web.org
tux > sudo a2enmod access_compat
tux > sudo a2enmod status
tux > sudo apachectl start
```

will add `access_compat` and `status` modules to the `APACHE_MODULES` variable of `/etc/sysconfig/apache2@example_web.org`, and then start the `example_web.org` instance.

24.8 Avoiding Security Problems

A Web server exposed to the public Internet requires an ongoing administrative effort. It is inevitable that security issues appear, both related to the software and to accidental misconfiguration. Here are some tips for how to deal with them.

24.8.1 Up-to-Date Software

If there are vulnerabilities found in the Apache software, a security advisory will be issued by SUSE. It contains instructions for fixing the vulnerabilities, which in turn should be applied when possible. The SUSE security announcements are available from the following locations:

- **Web Page.** <https://www.suse.com/support/security/>
- **Mailing List Archive.** <https://lists.opensuse.org/opensuse-security-announce/>
- **List of Security Announcements.** <https://www.suse.com/support/update/>

24.8.2 DocumentRoot Permissions

By default in openSUSE Leap, the `DocumentRoot` directory `/srv/www/htdocs` and the CGI directory `/srv/www/cgi-bin` belong to the user and group `root`. You should not change these permissions. If the directories are writable for all, any user can place files into them. These files might then be executed by Apache with the permissions of `wwwrun`, which may give the user unintended access to file system resources. Use subdirectories of `/srv/www` to place the `DocumentRoot` and CGI directories for your virtual hosts and make sure that directories and files belong to user and group `root`.

24.8.3 File System Access

By default, access to the whole file system is denied in `/etc/apache2/httpd.conf`. You should never overwrite these directives, but specifically enable access to all directories Apache should be able to read. For details, see [Section 24.2.2.1.3, “Basic Virtual Host Configuration”](#). In doing so, ensure that no critical files, such as password or system configuration files, can be read from the outside.

24.8.4 CGI Scripts

Interactive scripts in PHP, SSI, or any other programming language can essentially run arbitrary commands and therefore present a general security issue. Scripts that will be executed from the server should only be installed from sources the server administrator trusts—allowing users to run their own scripts is generally not a good idea. It is also recommended to do security audits for all scripts.

To make the administration of scripts as easy as possible, it is common practice to limit the execution of CGI scripts to specific directories instead of globally allowing them. The directives `ScriptAlias` and `Option ExecCGI` are used for configuration. The openSUSE Leap default configuration does not allow execution of CGI scripts from everywhere.

All CGI scripts run as the same user, so different scripts can potentially conflict with each other. The module `suEXEC` lets you run CGI scripts under a different user and group.

24.8.5 User Directories

When enabling user directories (with `mod_userdir` or `mod_rewrite`) you should strongly consider not allowing `.htaccess` files, which would allow users to overwrite security settings. At least you should limit the user's engagement by using the directive `AllowOverride`. In openSUSE Leap, `.htaccess` files are enabled by default, but the user is not allowed to overwrite any `Option` directives when using `mod_userdir` (see the `/etc/apache2/mod_userdir.conf` configuration file).

24.9 Troubleshooting

If Apache does not start, the Web page is not accessible, or users cannot connect to the Web server, it is important to find the cause of the problem. Here are some typical places to look for error explanations and important things to check:

Output of the `apache2.service` subcommand:

Instead of starting and stopping the Web server with the binary `/usr/sbin/apache2ctl`, rather use the `systemctl` commands instead (described in [Section 24.3, "Starting and Stopping Apache"](#)). `systemctl status apache2` is verbose about errors, and it even provides tips and hints for fixing configuration errors.

Log Files and Verbosity

In case of both fatal and nonfatal errors, check the Apache log files for causes, mainly the error log file located at `/var/log/apache2/error_log` by default. Additionally, you can control the verbosity of the logged messages with the `LogLevel` directive if more detail is needed in the log files.



Tip: A Simple Test

Watch the Apache log messages with the command `tail -F /var/log/apache2/MY_ERROR_LOG`. Then run `systemctl restart apache2`. Now, try to connect with a browser and check the output.

Firewall and Ports

A common mistake is to not open the ports for Apache in the firewall configuration of the server. If you configure Apache with YaST, there is a separate option available to take care of this specific issue (see [Section 24.2.3, "Configuring Apache with YaST"](#)). If you are configuring Apache manually, open firewall ports for HTTP and HTTPS via YaST's firewall module.

If the error cannot be tracked down with any of these, check the online Apache bug database at http://httpd.apache.org/bug_report.html. Additionally, the Apache user community can be reached via a mailing list available at <http://httpd.apache.org/userslist.html>.

24.10 For More Information

The package `apache2-doc` contains the complete Apache manual in various localizations for local installation and reference. It is not installed by default—the quickest way to install it is to use the command `zypper in apache2-doc`. Having been installed, the Apache manual is available at <http://localhost/manual/>. You may also access it on the Web at <http://httpd.apache.org/docs-2.4/>. SUSE-specific configuration hints are available in the directory `/usr/share/doc/packages/apache2/README.*`.

24.10.1 Apache 2.4

For a list of new features in Apache 2.4, refer to http://httpd.apache.org/docs/2.4/new_features_2_4.html. Information about upgrading from version 2.2 to 2.4 is available at <http://httpd.apache.org/docs-2.4/upgrading.html>.

24.10.2 Apache Modules

More information about external Apache modules that are briefly described in *Section 24.4.5, “External Modules”* is available at the following locations:

`mod_apparmor`

<http://en.opensuse.org/SDB:AppArmor>

`mod_php5`

<http://www.php.net/manual/en/install.unix.apache2.php>

`mod_python`

<http://www.modpython.org/>

`mod_security`

<http://modsecurity.org/>

24.10.3 Development

More information about developing Apache modules or about getting involved in the Apache Web server project are available at the following locations:

Apache Developer Information

<http://httpd.apache.org/dev/> ↗

Apache Developer Documentation

<http://httpd.apache.org/docs/2.4/developer/> ↗

25 Setting Up an FTP Server with YaST

Using the YaST *FTP Server* module, you can configure your machine to function as an FTP (File Transfer Protocol) server. Anonymous and/or authenticated users can connect to your machine and download files using the FTP protocol. Depending on the configuration, they can also upload files to the FTP server. YaST uses vsftpd (Very Secure FTP Daemon).

If the YaST FTP Server module is not available in your system, install the `yast2-ftp-server` package. (For managing the FTP server from the command line, see [Section 1.4.3.7, “yast ftp-server”](#).)

To configure the FTP server using YaST, follow these steps:

1. Open the YaST control center and choose *Network Services* > *FTP Server* or run the `yast2 ftp-server` command as `root`.
2. If there is not any FTP server installed in your system, you will be asked which server to install when the YaST FTP Server module starts. Choose the vsftpd server and confirm the dialog.
3. In the *Start-Up* dialog, configure the options for starting of the FTP server. For more information, see [Section 25.1, “Starting the FTP Server”](#).
In the *General* dialog, configure FTP directories, welcome message, file creation masks and other parameters. For more information, see [Section 25.2, “FTP General Settings”](#).
In the *Performance* dialog, set the parameters that affect the load on the FTP server. For more information, see [Section 25.3, “FTP Performance Settings”](#).
In the *Authentication* dialog, set whether the FTP server should be available for anonymous and/or authenticated users. For more information, see [Section 25.4, “Authentication”](#).
In the *Expert Settings* dialog, configure the operation mode of the FTP server, SSL connections and firewall settings. For more information, see [Section 25.5, “Expert Settings”](#).
4. Click *Finish* to save the configurations.

25.1 Starting the FTP Server

In the *Service Start* frame of the *FTP Start-Up* dialog set the way the FTP server is started up. You can choose between starting the server automatically during the system boot and starting it manually. If the FTP server should be started only after an FTP connection request, choose *Via socket*.

The current status of the FTP server is shown in the *Switch On and Off* frame of the *FTP Start-Up* dialog. Start the FTP server by clicking *Start FTP Now*. To stop the server, click *Stop FTP Now*. After having changed the settings of the server click *Save Settings and Restart FTP Now*. Your configurations will be saved by leaving the configuration module with *Finish*.

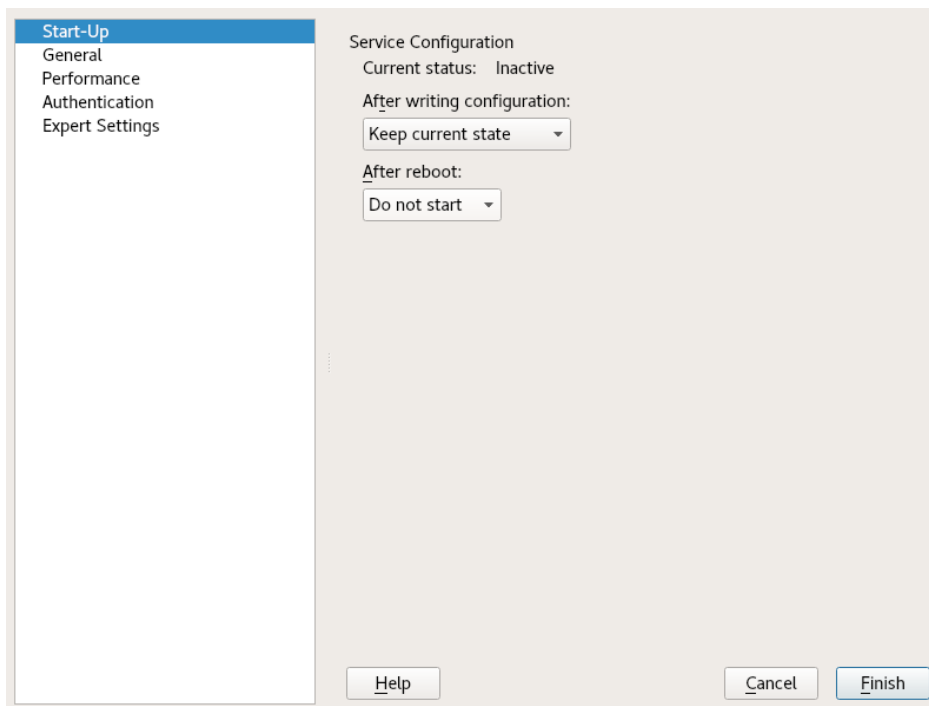


FIGURE 25.1: FTP SERVER CONFIGURATION — START-UP

25.2 FTP General Settings

In the *General Settings* frame of the *FTP General Settings* dialog you can set the *Welcome message* which is shown after connecting to the FTP server.

If you check the *Chroot Everyone* option, all local users will be placed in a chroot jail in their home directory after login. This option has security implications, especially if the users have upload permission or shell access, so be careful enabling this option.

If you check the *Verbose Logging* option, all FTP requests and responses are logged.

You can limit permissions of files created by anonymous and/or authenticated users with `umask`. Set the file creation mask for anonymous users in *Umask for Anonymous* and the file creation mask for authenticated users in *Umask for Authenticated Users*. The masks should be entered as octal numbers with a leading zero. For more information about `umask`, see the `umask` man page (`man 1p umask`).

In the *FTP Directories* frame set the directories used for anonymous and authorized users. With *Browse*, you can select a directory to be used from the local file system. The default FTP directory for anonymous users is `/srv/ftp`. Note that `vsftpd` does not allow this directory to be writable for all users. The subdirectory `upload` with write permissions for anonymous users is created instead.

25.3 FTP Performance Settings

In the *Performance* dialog set the parameters which affect the load on the FTP server. *Max Idle Time* is the maximum time (in minutes) the remote client may spend between FTP commands. In case of longer inactivity, the remote client is disconnected. *Max Clients for One IP* determines the maximum number of clients which can be connected from a single IP address. *Max Clients* determines the maximum number of clients which may be connected. Any additional clients will get an error message.

The maximum data transfer rate (in KB/s) is set in *Local Max Rate* for local authenticated users, and in *Anonymous Max Rate* for anonymous clients respectively. The default value for the rate settings is `0`, which means unlimited data transfer rate.

25.4 Authentication

In the *Enable/Disable Anonymous and Local Users* frame of the *Authentication* dialog, you can set which users are allowed to access your FTP server. You can choose between the following options: granting access to anonymous users only, to authenticated users only (with accounts on the system) or to both types of users.

To allow users to upload files to the FTP server, check *Enable Upload* in the *Uploading* frame of the *Authentication* dialog. Here you can allow uploading or creating directories even for anonymous users by checking the respective box.



Note: vsftp—Allowing File Upload for Anonymous Users

If a vsftpd server is used and you want anonymous users to be able to upload files or create directories, a subdirectory with writing permissions for all users needs to be created in the anonymous FTP directory.

25.5 Expert Settings

An FTP server can run in active or in passive mode. By default the server runs in passive mode. To switch into active mode, deselect the *Enable Passive Mode* option in the *Expert Settings* dialog. You can also change the range of ports on the server used for the data stream by tweaking the *Min Port for Pas. Mode* and *Max Port for Pas. Mode* options.

If you want encrypted communication between clients and the server, you can *Enable SSL*. Check the versions of the protocol to be supported and specify the DSA certificate to be used for SSL encrypted connections.

If your system is protected by a firewall, check *Open Port in Firewall* to enable a connection to the FTP server.

25.6 For More Information

For more information about the FTP server read the manual pages of [vsftpd](#) and [vsftpd.conf](#).

26 Squid Caching Proxy Server

Squid is a widely-used caching proxy server for Linux and Unix platforms. This means that it stores requested Internet objects, such as data on a Web or FTP server, on a machine that is closer to the requesting workstation than the server. It can be set up in multiple hierarchies to assure optimal response times and low bandwidth usage, even in modes that are transparent to end users.

Squid acts as a caching proxy server. It redirects object requests from clients (in this case, from Web browsers) to the server. When the requested objects arrive from the server, it delivers the objects to the client and keeps a copy of them in the hard disk cache. An advantage of caching is that several clients requesting the same object can be served from the hard disk cache. This enables clients to receive the data much faster than from the Internet. This procedure also reduces the network traffic.

Along with actual caching, Squid offers a wide range of features:

- Distributing load over intercommunicating hierarchies of proxy servers
- Defining strict access control lists for all clients accessing the proxy server
- Allowing or denying access to specific Web pages using other applications
- Generating statistics about frequently-visited Web pages for the assessment of surfing habits

Squid is not a generic proxy server. It normally proxies only HTTP connections. It supports the protocols FTP, Gopher, SSL, and WAIS, but it does not support other Internet protocols, such as the news protocol, or video conferencing protocols. Because Squid only supports the UDP protocol to provide communication between different caches, many multimedia programs are not supported.

26.1 Some Facts about Proxy Servers

As a caching proxy server, Squid can be used in several ways. When combined with a firewall, it can help with security. Multiple proxies can be used together. It can also determine what types of objects should be cached and for how long.

26.1.1 Squid and Security

It is possible to use Squid together with a firewall to secure internal networks from the outside. The firewall denies all clients access to external services except Squid. All Web connections must be established by the proxy server. With this configuration, Squid completely controls Web access.

If the firewall configuration includes a demilitarized zone (DMZ), the proxy server should operate within this zone. [Section 26.6, "Configuring a Transparent Proxy"](#) describes how to implement a *transparent* proxy. This simplifies the configuration of the clients, because in this case, they do not need any information about the proxy server.

26.1.2 Multiple Caches

Several instances of Squid can be configured to exchange objects between them. This reduces the total system load and increases the chances of retrieving an object from the local network. It is also possible to configure cache hierarchies, so a cache can forward object requests to sibling caches or to a parent cache—causing it to request objects from another cache in the local network, or directly from the source.

Choosing the appropriate topology for the cache hierarchy is very important, because it is not desirable to increase the overall traffic on the network. For a very large network, it would make sense to configure a proxy server for every subnet and connect them to a parent proxy server, which in turn is connected to the caching proxy server of the ISP.

All this communication is handled by ICP (Internet cache protocol) running on top of the UDP protocol. Data transfers between caches are handled using HTTP (hypertext transmission protocol) based on TCP.

To find the most appropriate server from which to request objects, a cache sends an ICP request to all sibling proxies. The sibling proxies answer these requests via ICP responses. If the object was detected, they use the code HIT, if not, they use MISS.

If multiple HIT responses were found, the proxy server decides from which server to download, depending on factors such as which cache sent the fastest answer or which one is closer. If no satisfactory responses are received, the request is sent to the parent cache.



Note: How Squid Avoids Duplication of Objects

To avoid duplication of objects in different caches in the network, other ICP protocols are used, such as CARP (cache array routing protocol) or HTCP (hypertext cache protocol). The more objects maintained in the network, the greater the possibility of finding the desired object.

26.1.3 Caching Internet Objects

Many objects available in the network are not static, such as dynamically generated pages and TLS/SSL-encrypted content. Objects like these are not cached because they change each time they are accessed.

To determine how long objects should remain in the cache, objects are assigned one of several states. Web and proxy servers find out the status of an object by adding headers to these objects, such as “Last modified” or “Expires” and the corresponding date. Other headers specifying that objects must not be cached can be used as well.

Objects in the cache are normally replaced, because of a lack of free disk space, using algorithms such as LRU (last recently used). This means that the proxy expunges those objects that have not been requested for the longest time.

26.2 System Requirements

System requirements largely depend on the maximum network load that the system must bear. Therefore, examine load peaks, as during those times, load might be more than four times the day's average. When in doubt, slightly overestimate the system's requirements. Having Squid working close to the limit of its capabilities can lead to a severe loss in quality of service. The following sections point to system factors in order of significance:

1. RAM size
2. CPU speed/physical CPU cores
3. Size of the disk cache
4. Hard disks/SSDs and their architecture

26.2.1 RAM

The amount of memory (RAM) required by Squid directly correlates with the number of objects in the cache. Random access memory is much faster than a hard disk/SSD. Therefore, it is very important to have sufficient memory for the Squid process, because system performance is dramatically reduced if the swap disk is used.

Squid also stores cache object references and frequently requested objects in the main memory to speed up retrieval of this data. In addition to that, there is other data that Squid needs to keep in memory, such as a table with all the IP addresses handled, an exact domain name cache, the most frequently requested objects, access control lists, buffers, and more.

26.2.2 CPU

Squid is tuned to work best with lower processor core counts (4–8 physical cores), with each providing high performance. Technologies providing virtual cores such as hyperthreading can hurt performance.

To make the best use of multiple CPU cores, it is necessary to set up multiple worker threads writing to different caching devices. By default, multi-core support is mostly disabled.

26.2.3 Size of the Disk Cache

In a small cache, the probability of a HIT (finding the requested object already located there) is small, because the cache is easily filled and less requested objects are replaced by newer ones. If, for example, 1 GB is available for the cache and the users use up only 10 MB per day surfing, it would take more than one hundred days to fill the cache.

The easiest way to determine the necessary cache size is to consider the maximum transfer rate of the connection. With a 1 Mbit/s connection, the maximum transfer rate is 128 KB/s. If all this traffic ended up in the cache, in one hour it would add up to 460 MB. Assuming that all this traffic is generated in only eight working hours, it would reach 3.6 GB in one day. Because the connection is normally not used to its upper volume limit, it can be assumed that the total data volume handled by the cache is approximately 2 GB. Hence, in this example, 2 GB of disk space is required for Squid to keep one day's worth of browsing data cached.

26.2.4 Hard Disk/SSD Architecture

Speed plays an important role in the caching process, so this factor deserves special attention. For hard disks/SSDs, this parameter is described as *random seek time* or *random read performance*, measured in milliseconds. Because the data blocks that Squid reads from or writes to the hard disk/SSD tend to be small, the seek time/read performance of the hard disk/SSD is more important than its data throughput.

For use as a proxy server, hard disks with high rotation speeds or SSDs are the best choice. When using hard disks, it can be better to use multiple smaller hard disks, each with a single cache directory to avoid excessive read times.

Using a RAID system allows increasing reliability at expense of speed. However, for performance reasons, avoid (software) RAID5 and similar settings.

File system choice is usually not decisive. However, using the mount option `noatime` can improve performance—Squid provides its own time stamps and thus does not need the file system to track access times.

26.3 Basic Usage of Squid

If not already installed, install the package `squid`. `squid` is not among the packages installed by default on openSUSE® Leap.

Squid is already preconfigured in openSUSE Leap, you can start it directly after the installation. To ensure a smooth start-up, the network should be configured in a way that at least one name server and the Internet can be reached. Problems can arise if a dial-up connection is used with a dynamic DNS configuration. In this case, at least the name server should be specified, because Squid does not start if it does not detect a DNS server in `/var/run/netconfig/resolv.conf`.

26.3.1 Starting Squid

To start Squid, use:

```
tux > sudo systemctl start squid
```

If you want Squid to start when the system boots up, enable the service with `systemctl enable squid`.

26.3.2 Checking Whether Squid Is Working

To check whether Squid is running, choose one of the following ways:

- Using **systemctl**:

```
tux > systemctl status squid
```

The output of this command should indicate that Squid is loaded and active (running).

- Using Squid itself:

```
tux > sudo squid -k check | echo $?
```

The output of this command should be 0, but may contain additional warnings or messages.

To test the functionality of Squid on the local system, choose one of the following ways:

- To test, you can use **squidclient**, a command-line tool that can output the response to a Web request, similar to **wget** or **curl**.

Unlike those tools, **squidclient** will automatically connect to the default proxy setup of Squid, localhost:3128. However, if you changed the configuration of Squid, you need to configure **squidclient** to use different settings using command line options. For more information, see **squidclient --help**.

EXAMPLE 26.1: A REQUEST WITH **squidclient**

```
tux > squidclient http://www.example.org
HTTP/1.1 200 OK
Cache-Control: max-age=604800
Content-Type: text/html
Date: Fri, 22 Jun 2016 12:00:00 GMT
Expires: Fri, 29 Jun 2016 12:00:00 GMT
Last-Modified: Fri, 09 Aug 2013 23:54:35 GMT
Server: ECS (iad/182A)
Vary: Accept-Encoding
X-Cache: HIT
x-ec-custom-error: 1
Content-Length: 1270
X-Cache: MISS from moon ①
X-Cache-Lookup: MISS from moon:3128
Via: 1.1 moon (squid/3.5.16) ②
Connection: close
```

```
<!doctype html>
<html>
<head>
  <title>Example Domain</title>
[...]
```

The output shown in *Example 26.1, “A Request With squidclient”* can be split into two parts:

1. The protocol headers of the response: the lines before the blank line.
2. The actual content of the response: the lines after the blank line.

To verify that Squid is used, refer to the selected header lines:

- ① The value of the header X-Cache tells you that the requested document was not in the Squid cache (MISS) of the computer moon.
The example above contains two X-Cache lines. You can ignore the first X-Cache header. It is produced by the internal caching software of the originating Web server.
 - ② The value of the header Via tells you the HTTP version, the name of the computer, and the version of Squid in use.
- Using a browser: Set up localhost as the proxy and 3128 as the port. You can then load a page and check the response headers in the *Network* panel of the browser's *Inspector* or *Developer Tools*. The headers should be reproduced similarly to the way shown in *Example 26.1, “A Request With squidclient”*.

To allow users from the local system and other systems to access Squid and the Internet, change the entry in the configuration files /etc/squid/squid.conf from http_access deny all to http_access allow all. However, in doing so, consider that Squid is made completely accessible to anyone by this action. Therefore, define ACLs (access control lists) that control access to the proxy server. After modifying the configuration file, Squid must be reloaded or restarted. For more information on ACLs, see *Section 26.5.2, “Options for Access Controls”*.

If Squid quits after a short period of time even though it was started successfully, check whether there is a faulty name server entry or whether the /var/run/netconfig/resolv.conf file is missing. Squid logs the cause of a start-up failure in the file /var/log/squid/cache.log.

26.3.3 Stopping, Reloading, and Restarting Squid

To reload Squid, choose one of the following ways:

- Using **systemctl**:

```
tux > sudo systemctl reload squid
```

or

```
tux > sudo systemctl restart squid
```

- Using YaST:

In the Squid module, click the *Save Settings and Restart Squid Now* button.

To stop Squid, choose one of the following ways:

- Using **systemctl**:

```
tux > sudo systemctl stop squid
```

- Using YaST

In the Squid module click the *Stop Squid Now.* button.

Shutting down Squid can take a while, because Squid waits up to half a minute before dropping the connections to the clients and writing its data to the disk (see shutdown_lifetime option in /etc/squid/squid.conf),



Warning: Terminating Squid

Terminating Squid with **kill** or **killall** can damage the cache. To be able to restart Squid, damaged caches must be deleted.

26.3.4 Removing Squid

Removing Squid from the system does not remove the cache hierarchy and log files. To remove these, delete the /var/cache/squid directory manually.

26.3.5 Local DNS Server

Setting up a local DNS server makes sense even if it does not manage its own domain. It then simply acts as a caching-only name server and is also able to resolve DNS requests via the root name servers without requiring any special configuration (see [Section 19.4, "Starting the BIND Name Server"](#)). How this can be done depends on whether you chose dynamic DNS during the configuration of the Internet connection.

Dynamic DNS

Normally, with dynamic DNS, the DNS server is set by the provider during the establishment of the Internet connection and the local `/var/run/netconfig/resolv.conf` file is adjusted automatically. This behavior is controlled in the `/etc/sysconfig/network/config` file with the `NETCONFIG_DNS_POLICY` sysconfig variable. Set `NETCONFIG_DNS_POLICY` to `"` with the YaST sysconfig editor.

Then, add the local DNS server in the `/var/run/netconfig/resolv.conf` file with the IP address `127.0.0.1` for `localhost`. This way, Squid can always find the local name server when it starts.

To make the provider's name server accessible, specify it in the configuration file `/etc/named.conf` under `forwarders` along with its IP address. With dynamic DNS, this can be achieved automatically when establishing the connection by setting the sysconfig variable `NETCONFIG_DNS_POLICY` to `auto`.

Static DNS

With static DNS, no automatic DNS adjustments take place while establishing a connection, so there is no need to change any sysconfig variables. However, you must specify the local DNS server in the file `/var/run/netconfig/resolv.conf` as described in [Dynamic DNS](#). Additionally, the provider's static name server must be specified manually in the `/etc/named.conf` file under `forwarders` along with its IP address.



Tip: DNS and Firewall

If you have a firewall running, make sure DNS requests can pass it.

26.4 The YaST Squid Module

The YaST Squid module contains the following tabs:

Start-Up

Specifies how Squid is started and which Firewall port is open on which interfaces.

HTTP Ports

Define all ports where Squid will listen for HTTP requests from clients.

Refresh Patterns

Defines how Squid treats objects in the cache.

Cache Settings

Defines settings in regard to cache memory, maximum and minimum object size, and more.

Cache Directory

Defines the top-level directory where Squid stores all cache swap files.

Access Control

Controls the access to the Squid server via ACL groups.

Logging and Timeout

Define paths to access, cache, and cache store log files in addition with connection timeouts and client lifetime.

Miscellaneous

Sets language and mail address of administrator.

26.5 The Squid Configuration File

All Squid proxy server settings are made in the `/etc/squid/squid.conf` file. To start Squid for the first time, no changes are necessary in this file, but external clients are initially denied access. The proxy is available for `localhost`. The default port is `3128`. The preinstalled configuration file `/etc/squid/squid.conf` provides detailed information about the options and many examples.

Many entries are commented and therefore begin with the comment character `#`. The relevant specifications can be found at the end of the line. The given values usually correlate with the default values, so removing the comment signs without changing any of the parameters usually

has no effect. If possible, leave the commented lines as they are and insert the options along with the modified values in the line below. This way, the default values may easily be recovered and compared with the changes.



Tip: Adapting the Configuration File After an Update

If you have updated from an earlier Squid version, it is recommended to edit the new `/etc/squid/squid.conf` and only apply the changes made in the previous file.

Sometimes, Squid options are added, removed, or modified. Therefore, if you try to use the old `squid.conf`, Squid might stop working properly.

26.5.1 General Configuration Options

The following is a list of a selection of configuration options for Squid. It is not exhaustive. The Squid package contains a full, lightly documented list of options in `/etc/squid/squid.conf.documented`.

`http_port` *PORT*

This is the port on which Squid listens for client requests. The default port is `3128`, but `8080` is also common.

`cache_peer` *HOST_NAME TYPE PROXY_PORT ICP_PORT*

This option allows creating a network of caches that work together. The cache peer is a computer that also hosts a network cache and stands in a relationship to your own. The type of relationship is specified as the *TYPE*. The type can either be `parent` or `sibling`. As the *HOST_NAME*, specify the name or IP address of the proxy server to use. For *PROXY_PORT*, specify the port number for use in a browser (usually `8080`). Set *ICP_PORT* to `7` or, if the ICP port of the parent is not known and its use is irrelevant to the provider, to `0`.

To make Squid behave like a Web browser instead of like a proxy server, prohibit the use of the ICP protocol. You can do so by appending the options `default` and `no-query`.

`cache_mem` *SIZE*

This option defines the amount of memory Squid can use for very popular replies. The default is `8 MB`. This does not specify the memory usage of Squid and may be exceeded.

`cache_dir` *STORAGE_TYPE CACHE_DIRECTORY CACHE_SIZE LEVEL_1_DIRECTORIES LEVEL_2_DIRECTORIES*

The option `cache_dir` defines the directory for the disk cache. In the default configuration on openSUSE Leap, Squid does not create a disk cache.

The placeholder `STORAGE_TYPE` can be one of the following:

- Directory-based storage types: `ufs`, `aufs` (the default), `diskd`. All three are variations of the storage format `ufs`. However, while `ufs` runs as part of the core Squid thread, `aufs` runs in a separate thread, and `diskd` uses a separate process. This means that the latter two types avoid blocking Squid because of disk I/O.
- Database-based storage systems: `rock`. This storage format relies on a single database file, in which each object takes up one or more memory units of a fixed size (“slots”).

In the following, only the parameters for storage types based on `ufs` will be discussed. `rock` has somewhat different parameters.

The `CACHE_DIRECTORY` is the directory for the disk cache. By default, that is `/var/cache/squid`. `CACHE_SIZE` is the maximum size of that directory in megabytes; by default, this is set to 100 MB. Set it to between 50% and a maximum of 80% of available disk space.

The final two values, `LEVEL_1_DIRECTORIES` and `LEVEL_2_DIRECTORIES` specify how many subdirectories are created in the `CACHE_DIRECTORY`. By default, 16 subdirectories are created at the first level below `CACHE_DIRECTORY` and 256 within each of these. These values should only be increased with caution, because creating too many directories can lead to performance problems.

If you have several disks that share a cache, specify several `cache_dir` lines.

`cache_access_log LOG_FILE,`

`cache_log LOG_FILE,`

`cache_store_log LOG_FILE`

These three options specify the paths where Squid logs all its actions. Normally, nothing needs to be changed here. If Squid is burdened by heavy usage, it might make sense to distribute the cache and the log files over several disks.

`client_netmask NETMASK`

This option allows masking IP addresses of clients in the log files by applying a subnet mask. For example, to set the last digit of the IP address to `0`, specify `255.255.255.0`.

`ftp_user E-MAIL`

This option allows setting the password that Squid should use for anonymous FTP login. Specify a valid e-mail address here, because some FTP servers check these for validity.

`cache_mgr E-MAIL`

If it unexpectedly crashes, Squid sends a message to this e-mail address. The default is *webmaster*.

logfile_rotate VALUE

If you run **squid -k rotate**, **squid** can rotate log files. The files are numbered in this process and, after reaching the specified value, the oldest file is overwritten. The default value is 10 which rotates log files with the numbers 0 to 9.

However, on openSUSE Leap, rotating log files is performed automatically using logrotate and the configuration file /etc/logrotate.d/squid.

append_domain DOMAIN

Use *append_domain* to specify which domain to append automatically when none is given. Usually, your own domain is specified here, so specifying *www* in the browser accesses your own Web server.

forwarded_for STATE

If this option is set to on, it adds a line to the header similar to this:

```
X-Forwarded-For: 192.168.0.1
```

If you set this option to off, Squid removes the IP address and the system name of the client from HTTP requests.

negative_ttl TIME,

negative_dns_ttl TIME

If these options are set, Squid will cache some types of failures, such as 404 responses. It will then refuse to issue new requests, even if the resource would be available then.

By default, negative_ttl is set to 0, negative_dns_ttl is set to 1 minutes. This means that negative responses to Web requests are not cached by default, while negative responses to DNS requests are cached for 1 minute.

never_direct allow ACL_NAME

To prevent Squid from taking requests directly from the Internet, use the option never_direct to force connection to another proxy server. This must have previously been specified in cache_peer. If all is specified as the ACL_NAME, all requests are forwarded directly to the parent. This can be necessary, for example, if you are using a provider that dictates the use of its proxies or denies its firewall direct Internet access.

26.5.2 Options for Access Controls

Squid provides a detailed system for controlling the access to the proxy server. These Access Control Lists (ACL) are lists with rules that are processed sequentially. ACLs must be defined before they can be used. Some default ACLs, such as `all` and `localhost`, already exist. However, the mere definition of an ACL does not mean that it is actually applied. This only happens when there is a corresponding `http_access` rule.

The syntax for the option `acl` is as follows:

```
acl ACL_NAME TYPE DATA
```

The placeholders within this syntax stand for the following:

- The name `ACL_NAME` can be chosen arbitrarily.
- For `TYPE`, select from a variety of different options which can be found in the `ACCESS CONTROLS` section in the `/etc/squid/squid.conf` file.
- The specification for `DATA` depends on the individual ACL type, for example host names, IP addresses, or URLs, and can also be read from a file.

To add rules in the YaST squid module, open the module and click the *Access Control* tab. Click *Add* under the ACL Groups list and enter the name of your rule, the type, and its parameters.

For more information on types of ACL rules, see the Squid documentation at <http://www.squid-cache.org/Versions/v3/3.5/cfgman/acl.html>.

EXAMPLE 26.2: DEFINING ACL RULES

```
acl mysurfers srcdomain .example.com ❶  
acl teachers src 192.168.1.0/255.255.255.0 ❷  
acl students src 192.168.7.0-192.168.9.0/255.255.255.0 ❸  
acl lunch time MTWHF 12:00-15:00 ❹
```

- ❶ This ACL defines `mysurfers` as all users coming from within `.example.com` (as determined by a reverse lookup for the IP).
- ❷ This ACL defines `teachers` as the users of computers with IP addresses starting with `192.168.1.`
- ❸ This ACL defines `students` as the users of the computer with IP addresses starting with `192.168.7.`, `192.168.8.`, or `192.168.9.`

- 4 This ACL defines lunch as a time on the days Monday through Friday between noon and 3 p.m.

http_access allow ACL_NAME

http_access defines who is allowed to use the proxy server and who can access what on the Internet. For this, ACLs must be defined. localhost and all have already been defined above for which you can deny or allow access via deny or allow. A list containing any number of http_access entries can be created, processed from top to bottom. Depending on which occurs first, access is allowed or denied to the respective URL. The last entry should always be http_access deny all. In the following example, localhost has free access to everything while all other hosts are denied access completely:

```
http_access allow localhost
http_access deny all
```

In another example using these rules, the group teachers always has access to the Internet. The group students only has access between Monday and Friday during lunch time:

```
http_access deny localhost
http_access allow teachers
http_access allow students lunch time
http_access deny all
```

For readability, within the configuration file /etc/squid/squid.conf, specify all http_access options as a block.

url_rewrite_program PATH

With this option, specify a URL rewriter.

auth_param basic program PATH

If users must be authenticated on the proxy server, set a corresponding program, such as /usr/sbin/pam_auth. When accessing pam_auth for the first time, the user sees a login window in which they need to specify a user name and a password. In addition, you need an ACL, so only clients with a valid login can use the Internet:

```
acl password proxy_auth REQUIRED

http_access allow password
http_access deny all
```

In the acl proxy_auth option, using REQUIRED means that all valid user names are accepted. REQUIRED can also be replaced with a list of permitted user names.

ident_lookup_access allow ACL_NAME

With this option, have an ident request run to find each user's identity for all clients defined by an ACL of the type src. Alternatively, use this for all clients, apply the predefined ACL all as the ACL_NAME.

All clients covered by ident_lookup_access must run an ident daemon. On Linux, you can use pidentd (package pidentd) as the ident daemon. For other operating systems, free software is usually available. To ensure that only clients with a successful ident lookup are permitted, define a corresponding ACL:

```
acl identhosts ident REQUIRED
http_access allow identhosts
http_access deny all
```

In the acl identhosts ident option, using REQUIRED means that all valid user names are accepted. REQUIRED can also be replaced with a list of permitted user names.

Using ident can slow down access time, because ident lookups are repeated for each request.

26.6 Configuring a Transparent Proxy

A transparent proxy intercepts and answers the requests of the Web browser, so the Web browser receives the requested pages without knowing where they are coming from. As the name indicates, the entire process is transparent to the user.

The usual way of working with proxy servers is as follows: the Web browser sends requests to a certain port of the proxy server and the proxy always provides these required objects, regardless of whether they are in its cache. However, in some cases the transparent proxy mode of Squid makes sense:

- If, for security reasons, it is recommended that all clients use a proxy server to surf the Internet.
- If all clients must use a proxy server, regardless of whether they are aware of it.
- If the proxy server in a network is moved, but the existing clients need to retain their old configuration.

PROCEDURE 26.1: SQUID AS A TRANSPARENT PROXY SERVER (COMMAND LINE)

1. In `/etc/squid/squid.conf`, on the line of the option `http_port` add the parameter `transparent`. You should then have 2 lines:

```
http_port 3128
http_port 3128 transparent
```

2. Restart Squid:

```
tux > sudo systemctl restart squid
```

3. Set up the firewall to redirect HTTP traffic to the port given in `http_proxy`. In the example above it is port 3128. Then reload the firewall configuration. This assumes that the zone `internal` is assigned to your LAN interface.

```
tux > sudo firewall-cmd --permanent --zone=internal \
--add-forward-port=port=80:proto=tcp:toport=3128:toaddr=LAN_IP
tux > sudo firewall-cmd --permanent --zone=internal --add-port=3128/tcp
tux > sudo firewall-cmd --reload
```

Replace `LAN_IP` with the IP address of your LAN interface or the interface Squid is listening on.

4. To verify that everything is working properly, check the Squid log files in `/var/log/squid/access.log`.

26.7 Using the Squid Cache Manager CGI Interface (`cachemgr.cgi`)

The Squid cache manager CGI interface (`cachemgr.cgi`) is a CGI utility for displaying statistics about the memory usage of a running Squid process. It is also a convenient way to manage the cache and view statistics without logging the server.

PROCEDURE 26.2: SETTING UP `cachemgr.cgi`

1. Make sure the Apache Web server is running on your system. Configure Apache as described in *Chapter 24, The Apache HTTP Server*. In particular, see *Section 24.5, "Enabling CGI Scripts"*. To check whether Apache is already running, use:

```
tux > sudo systemctl status apache2
```

If `inactive` is shown, you can start Apache with the openSUSE Leap default settings:

```
tux > sudo systemctl start apache2
```

2. Now enable `cachemgr.cgi` in Apache. To do so, create a configuration file for a `ScriptAlias`.

Create the file in the directory `/etc/apache2/conf.d` and name it `cachemgr.conf`. In it, add the following:

```
ScriptAlias /squid/cgi-bin/ /usr/lib64/squid/  
  
<Directory "/usr/lib64/squid/">  
Options +ExecCGI  
AddHandler cgi-script .cgi  
Require host HOST_NAME  
</Directory>
```

Replace `HOST_NAME` with the host name of the computer you want to access `cachemgr.cgi` from. This allows only your computer to access `cachemgr.cgi`. To allow access from anywhere, use `Require all granted` instead.

3.
 - If Squid and your Apache Web server run on the same computer, there should be no changes that need to be made to `/etc/squid/squid.conf`. However, verify that `/etc/squid/squid.conf` contains the following lines:

```
http_access allow manager localhost  
http_access deny manager
```

These lines allow you to access the manager interface from your own computer (`localhost`) but not from elsewhere.

- If Squid and your Apache Web server run on different computers, you need to add extra rules to allow access from the CGI script to Squid. Define an ACL for your server (replace `WEB_SERVER_IP` with the IP address of your Web server):

```
acl webserver src WEB_SERVER_IP/255.255.255.255
```

Make sure the following rules are in the configuration file. Compared to the default configuration, only the rule in the middle is new. However, the sequence is important.

```
http_access allow manager localhost
```



```
http_access allow manager webserver
http_access deny manager
```

4. (Optional) Optionally, you can configure one or more passwords for `cachemgr.cgi`. This also allows access to more actions such as closing the cache remotely or viewing more information about the cache. For this, configure the options `cache_mgr` and `cachemgr_passwd` with one or more password for the manager and a list of allowed actions.

For example, to explicitly enable viewing the index page, the menu, 60-minute average of counters without authentication, to enable toggling offline mode using the password `secretpassword`, and to completely disable everything else, use the following configuration:

```
cache_mgr user
cachemgr_passwd none index menu 60min
cachemgr_passwd secretpassword offline_toggle
cachemgr_passwd disable all
```

`cache_mgr` defines a user name. `cache_mgr` defines which actions are allowed using which password.

The keywords `none` and `disable` are special: `none` removes the need for a password, `disable` disables functionality outright.

The full list of actions can be best seen after logging in to `cachemgr.cgi`. To find out how the operation needs to be referenced in the configuration file, see the string after `&operation=` in the URL of the action page. `all` is a special keyword meaning all actions.

5. Reload Squid and Apache after the configuration file changes:

```
tux > sudo systemctl reload squid
```

6. To view the statistics, go to the `cachemgr.cgi` page that you set up before. For example, it could be `http://webserver.example.org/squid/cgi-bin/cachemgr.cgi`. Choose the right server, and, if set, specify user name and password. Then click *Continue* and browse through the different statistics.

26.8 Cache Report Generation with Calamaris

Calamaris is a Perl script used to generate reports of cache activity in ASCII or HTML format. It works with native Squid access log files. The Calamaris home page is located at <http://cord.de/calamaris-english>. This tool does not belong to the openSUSE Leap default installation scope—to use it, install the `calamaris` package.

Log in as `root`, then enter:

```
root # cat access1.log [access2.log access3.log] | calamaris OPTIONS > reportfile
```

When using more than one log file, make sure they are chronologically ordered, with older files listed first. This can be achieved by either listing the files one after the other as in the example above, or by using `access{1..3}.log`.

`calamaris` takes the following options:

- `-a`
output all available reports
- `-w`
output as HTML report
- `-l`
include a message or logo in report header

More information about the various options can be found in the program's manual page with `man calamaris`.

A typical example is:

```
root # cat access.log.{10..1} access.log | calamaris -a -w \  
> /usr/local/httpd/htdocs/Squid/squidreport.html
```

This puts the report in the directory of the Web server. Apache is required to view the reports.

26.9 For More Information

Visit the home page of Squid at <http://www.squid-cache.org/>. Here, find the “Squid User Guide” and a very extensive collection of FAQs on Squid.

In addition, mailing lists are available for Squid at <http://www.squid-cache.org/Support/mailling-lists.html>.

IV Mobile Computers

- 27 Mobile Computing with Linux **487**
- 28 Using NetworkManager **498**
- 29 Power Management **508**

27 Mobile Computing with Linux

Mobile computing is mostly associated with laptops, PDAs and cellular phones (and the data exchange between them). Mobile hardware components, such as external hard disks, flash disks, or digital cameras, can be connected to laptops or desktop systems. A number of software components are involved in mobile computing scenarios and some applications are tailor-made for mobile use.

27.1 Laptops

The hardware of laptops differs from that of a normal desktop system. This is because criteria like exchangeability, space requirements and power consumption must be taken into account. The manufacturers of mobile hardware have developed standard interfaces like Mini PCI and Mini PCIe that can be used to extend the hardware of laptops. The standards cover memory cards, network interface cards, and external hard disks.

27.1.1 Power Conservation

The inclusion of energy-optimized system components during laptop manufacturing contributes to their suitability for use without access to the electrical power grid. Their contribution to conservation of power is at least as important as that of the operating system. openSUSE® Leap supports various methods that control the power consumption of a laptop and have varying effects on the operating time under battery power. The following list is in descending order of contribution to power conservation:

- Throttling the CPU speed.
- Switching off the display illumination during pauses.
- Manually adjusting the display illumination.
- Disconnecting unused, hotplug-enabled accessories (USB CD-ROM, external mouse, Wi-Fi, etc.).
- Spinning down the hard disk when idling.

Detailed background information about power management in openSUSE Leap is provided in *Chapter 29, Power Management*.

27.1.2 Integration in Changing Operating Environments

Your system needs to adapt to changing operating environments when used for mobile computing. Many services depend on the environment and the underlying clients must be reconfigured. openSUSE Leap handles this task for you.

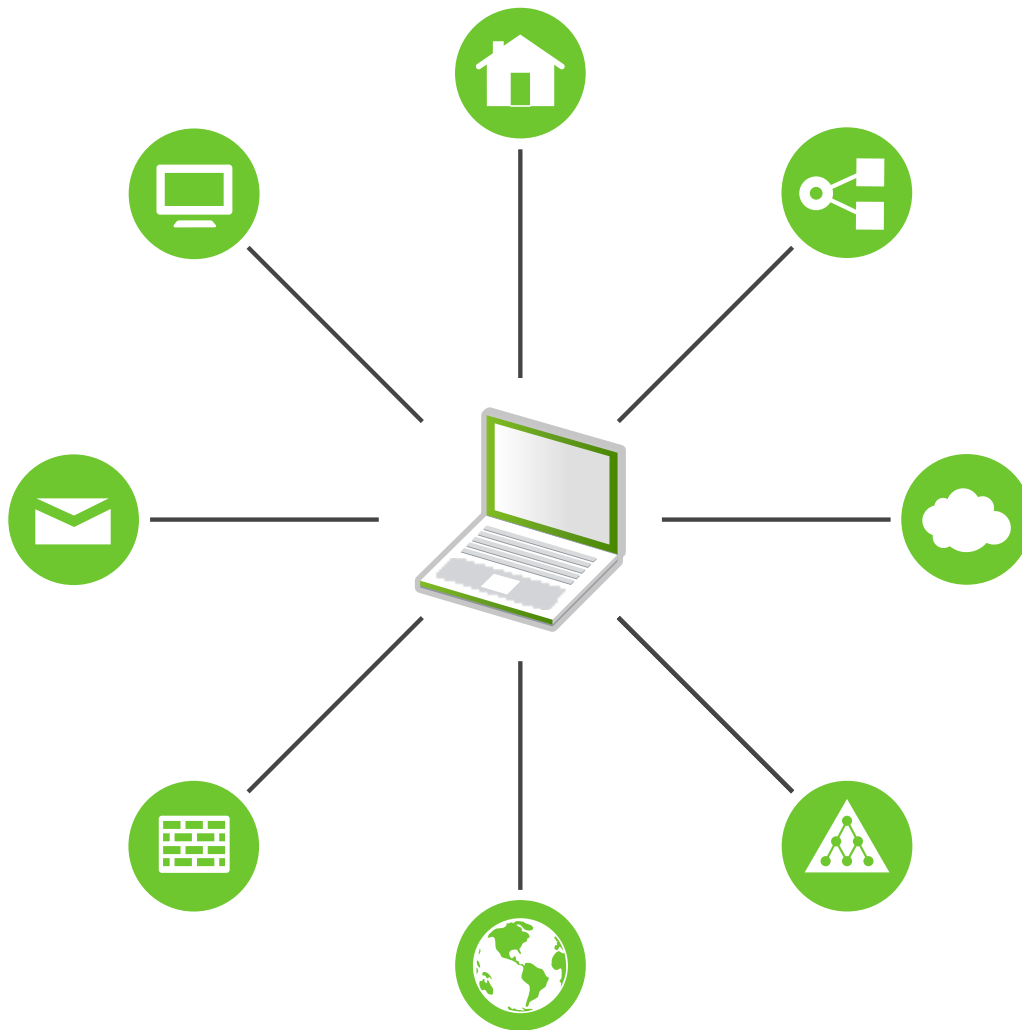


FIGURE 27.1: INTEGRATING A MOBILE COMPUTER IN AN EXISTING ENVIRONMENT

The services affected in the case of a laptop commuting back and forth between a small home network and an office network are:

Network

This includes IP address assignment, name resolution, Internet connectivity and connectivity to other networks.

Printing

A current database of available printers and an available print server must be present, depending on the network.

E-Mail and Proxies

As with printing, the list of the corresponding servers must be current.

X (Graphical Environment)

If your laptop is temporarily connected to a projector or an external monitor, different display configurations must be available.

openSUSE Leap offers several ways of integrating laptops into existing operating environments:

NetworkManager

NetworkManager is especially tailored for mobile networking on laptops. It provides a means to easily and automatically switch between network environments or different types of networks such as mobile broadband (such as GPRS, EDGE, or 3G), wireless LAN, and Ethernet. NetworkManager supports WEP and WPA-PSK encryption in wireless LANs. It also supports dial-up connections. The GNOME desktop includes a front-end for NetworkManager. For more information, see [Section 28.3, "Configuring Network Connections"](#).

TABLE 27.1: USE CASES FOR NETWORKMANAGER

| My computer... | Use NetworkManager |
|---|--------------------|
| is a laptop | Yes |
| is sometimes attached to different networks | Yes |
| provides network services (such as DNS or DHCP) | No |
| only uses a static IP address | No |

Use the YaST tools to configure networking whenever NetworkManager should not handle network configuration.



Tip: DNS Configuration and Various Types of Network Connections

If you travel frequently with your laptop and change different types of network connections, NetworkManager works fine when all DNS addresses are assigned correctly assigned with DHCP. If some connections use static DNS address(es), add it to the `NETCONFIG_DNS_STATIC_SERVERS` option in `/etc/sysconfig/network/config`.

SLP

The service location protocol (SLP) simplifies the connection of a laptop to an existing network. Without SLP, the administrator of a laptop usually requires detailed knowledge of the services available in a network. SLP broadcasts the availability of a certain type of service to all clients in a local network. Applications that support SLP can process the information dispatched by SLP and be configured automatically. SLP can also be used to install a system, minimizing the effort of searching for a suitable installation source. Find detailed information about SLP in [Chapter 17, SLP](#).

27.1.3 Software Options

There are various task areas in mobile use that are covered by dedicated software: system monitoring (especially the battery charge), data synchronization, and wireless communication with peripherals and the Internet. The following sections cover the most important applications that openSUSE Leap provides for each task.

27.1.3.1 System Monitoring

Two system monitoring tools are provided by openSUSE Leap:

Power Management

Power Management is an application that lets you adjust the energy saving related behavior of the GNOME desktop. You can typically access it via *Computer > Control Center > System > Power Management*.

System Monitor

The *System Monitor* gathers measurable system parameters into one monitoring environment. It presents the output information in three tabs by default. *Processes* gives detailed information about currently running processes, such as CPU load, memory usage, or process ID number and priority. The presentation and filtering of the collected data can be customized—to add a new type of process information, left-click the process table header and choose which column to hide or add to the view. It is also possible to monitor different system parameters in various data pages or collect the data of various machines in parallel over the network. The *Resources* tab shows graphs of CPU, memory and network history and the *File System* tab lists all partitions and their usage.

27.1.3.2 Synchronizing Data

When switching between working on a mobile machine disconnected from the network and working at a networked workstation in an office, it is necessary to keep processed data synchronized across all instances. This could include e-mail folders, directories and individual files that need to be present for work on the road and at the office. The solution in both cases is as follows:

Synchronizing E-Mail

Use an IMAP account for storing your e-mails in the office network. Then access the e-mails from the workstation using any disconnected IMAP-enabled e-mail client, like Mozilla Thunderbird or Evolution as described in *Book “GNOME User Guide”*. The e-mail client must be configured so that the same folder is always accessed for Sent messages. This ensures that all messages are available along with their status information after the synchronization process has completed. Use an SMTP server implemented in the mail client for sending messages instead of the system-wide MTA postfix or sendmail to receive reliable feedback about unsent mail.

Synchronizing Files and Directories

There are several utilities suitable for synchronizing data between a laptop and a workstation. One of the most widely used is a command-line tool called rsync. For more information, see its manual page (man 1 rsync).

27.1.3.3 Wireless Communication: Wi-Fi

With the largest range of these wireless technologies, Wi-Fi is the only one suitable for the operation of large and sometimes even spatially separate networks. Single machines can connect with each other to form an independent wireless network or access the Internet. Devices called *access points* act as base stations for Wi-Fi-enabled devices and act as intermediaries for access to the Internet. A mobile user can switch among access points depending on location and which access point is offering the best connection. Like in cellular telephony, a large network is available to Wi-Fi users without binding them to a specific location for accessing it.

Wi-Fi cards communicate using the 802.11 standard, prepared by the IEEE organization. Originally, this standard provided for a maximum transmission rate of 2 Mbit/s. Meanwhile, several supplements have been added to increase the data rate. These supplements define details such as the modulation, transmission output, and transmission rates (see [Table 27.2, "Overview of Various Wi-Fi Standards"](#)). Additionally, many companies implement hardware with proprietary or draft features.

TABLE 27.2: OVERVIEW OF VARIOUS WI-FI STANDARDS

| Name (802.11) | Frequency (GHz) | Maximum Transmission Rate (Mbit/s) | Note |
|---------------|-----------------|------------------------------------|--|
| a | 5 | 54 | Less interference-prone |
| b | 2.4 | 11 | Less common |
| g | 2.4 | 54 | Widespread, backward-compatible with 11b |
| n | 2.4 and/or 5 | 300 | Common |
| ac | 5 | up to ~865 | Expected to be common in 2015 |

| Name (802.11) | Frequency (GHz) | Maximum Transmission Rate (Mbit/s) | Note |
|---------------|-----------------|------------------------------------|--|
| ad | 60 | up to appr. 7000 | Released 2012, currently less common; not supported in openSUSE Leap |

802.11 Legacy cards are not supported by openSUSE® Leap. Most cards using 802.11 a/b/g/n are supported. New cards usually comply with the 802.11n standard, but cards using 802.11g are still available.

27.1.3.3.1 Operating Modes

In wireless networking, various techniques and configurations are used to ensure fast, high-quality, and secure connections. Usually your Wi-Fi card operates in *managed mode*. However, different operating types need different setups. Wireless networks can be classified into four network modes:

Managed Mode (Infrastructure Mode), via Access Point (default mode)

Managed networks have a managing element: the access point. In this mode (also called infrastructure or default mode), all connections of the Wi-Fi stations in the network run through the access point, which may also serve as a connection to an Ethernet. To make sure only authorized stations can connect, various authentication mechanisms (WPA, etc.) are used. This is also the main mode that consumes the least amount of energy.

Ad-hoc Mode (Peer-to-Peer Network)

Ad-hoc networks do not have an access point. The stations communicate directly with each other, therefore an ad-hoc network is usually slower than a managed network. However, the transmission range and number of participating stations are greatly limited in ad-hoc networks. They also do not support WPA authentication. Additionally, not all cards support ad-hoc mode reliably.

Master Mode

In master mode, your Wi-Fi card is used as the access point, assuming your card supports this mode. Find out the details of your Wi-Fi card at <http://linux-wless.passys.nl>.

Mesh Mode

Wireless mesh networks are organized in a *mesh topology*. A wireless mesh network's connection is spread among all wireless mesh *nodes*. Each node belonging to this network is connected to other nodes to share the connection, possibly over a large area.

27.1.3.3.2 Authentication

Because a wireless network is much easier to intercept and compromise than a wired network, the various standards include authentication and encryption methods.

Old Wi-Fi cards support only WEP (Wired Equivalent Privacy). However, because WEP has proven to be insecure, the Wi-Fi industry has defined an extension called WPA, which is supposed to eliminate the weaknesses of WEP. WPA, sometimes synonymous with WPA2, should be the default authentication method.

Usually the user cannot choose the authentication method. For example, when a card operates in managed mode the authentication is set by the access point. NetworkManager shows the authentication method.

27.1.3.3.3 Encryption

There are various encryption methods to ensure that no unauthorized person can read the data packets that are exchanged in a wireless network or gain access to the network:

WEP (defined in IEEE 802.11)

This standard uses the RC4 encryption algorithm, originally with a key length of 40 bits, later also with 104 bits. Often, the length is declared as 64 bits or 128 bits, depending on whether the 24 bits of the initialization vector are included. However, this standard has some weaknesses. Attacks against the keys generated by this system may be successful. Nevertheless, it is better to use WEP than not to encrypt the network.

Some vendors have implemented the non-standard “Dynamic WEP”. It works exactly as WEP and shares the same weaknesses, except that the key is periodically changed by a key management service.

TKIP (defined in WPA/IEEE 802.11i)

This key management protocol defined in the WPA standard uses the same encryption algorithm as WEP, but eliminates its weakness. Because a new key is generated for every data packet, attacks against these keys are fruitless. TKIP is used together with WPA-PSK.

CCMP (defined in IEEE 802.11i)

CCMP describes the key management. Usually, it is used in connection with WPA-EAP, but it can also be used with WPA-PSK. The encryption takes place according to AES and is stronger than the RC4 encryption of the WEP standard.

27.1.3.4 Wireless Communication: Bluetooth

Bluetooth has the broadest application spectrum of all wireless technologies. It can be used for communication between computers (laptops) and PDAs or cellular phones, as can IrDA. It can also be used to connect various computers within range. Bluetooth is also used to connect wireless system components, like a keyboard or a mouse. The range of this technology is, however, not sufficient to connect remote systems to a network. Wi-Fi is the technology of choice for communicating through physical obstacles like walls.

27.1.3.5 Wireless Communication: IrDA

IrDA is the wireless technology with the shortest range. Both communication parties must be within viewing distance of each other. Obstacles like walls cannot be overcome. One possible application of IrDA is the transmission of a file from a laptop to a cellular phone. The short path from the laptop to the cellular phone is then covered using IrDA. Long-range transmission of the file to the recipient is handled by the mobile network. Another application of IrDA is the wireless transmission of printing jobs in the office.

27.1.4 Data Security

Ideally, you protect data on your laptop against unauthorized access in multiple ways. Possible security measures can be taken in the following areas:

Protection against Theft

Always physically secure your system against theft whenever possible. Various securing tools (like chains) are available in retail stores.

Strong Authentication

Use biometric authentication in addition to standard authentication via login and password. openSUSE Leap supports fingerprint authentication.

Securing Data on the System

Important data should not only be encrypted during transmission, but also on the hard disk. This ensures its safety in case of theft. The creation of an encrypted partition with openSUSE Leap is described in *Book "Security and Hardening Guide", Chapter 14 "Encrypting Partitions and Files"*. Another possibility is to create encrypted home directories when adding the user with YaST.

Important: Data Security and Suspend to Disk

Encrypted partitions are not unmounted during a suspend to disk event. Thus, all data on these partitions is available to any party who manages to steal the hardware and issue a resume of the hard disk.

Network Security

Any transfer of data should be secured, no matter how the transfer is done. Find general security issues regarding Linux and networks in *Book "Security and Hardening Guide", Chapter 1 "Security and Confidentiality"*.

27.2 Mobile Hardware

openSUSE Leap supports the automatic detection of mobile storage devices over FireWire (IEEE 1394) or USB. The term *mobile storage device* applies to any kind of FireWire or USB hard disk, flash disk, or digital camera. These devices are automatically detected and configured when they are connected with the system over the corresponding interface. The file manager of GNOME offers flexible handling of mobile hardware items. To unmount any of these media safely, use the *Unmount Volume* (GNOME) feature of the file manager. For more details refer to *Book "GNOME User Guide"*.

External Hard Disks (USB and FireWire)

When an external hard disk is correctly recognized by the system, its icon appears in the file manager. Clicking the icon displays the contents of the drive. It is possible to create directories and files here and edit or delete them. To rename a hard disk, select the corresponding menu item from the right-click contextual menu. This name change is limited to display in the file manager. The descriptor by which the device is mounted in `/media` remains unaffected.

USB Flash Drives

These devices are handled by the system like external hard disks. It is similarly possible to rename the entries in the file manager.

Digital Cameras (USB and FireWire)

Digital cameras recognized by the system also appear as external drives in the overview of the file manager. The images can then be processed using the default image editor. For advanced photo processing use The GIMP. For a short introduction to The GIMP, see *Book "GNOME User Guide", Chapter 17 "GIMP: Manipulating Graphics"*.

27.3 Mobile Devices (Smartphones and Tablets)

A desktop system or a laptop can communicate with mobile devices via Bluetooth, Wi-Fi, or a direct USB connection. Choosing a connection method depends on your mobile device model and your specific needs. Connecting a mobile device to a desktop machine or laptop via USB usually makes it possible to work with the device as conventional external storage. Setting up a Bluetooth or Wi-Fi connection allows you to interact with the mobile device and control its functions directly from your desktop machine or laptop. There are several open-source graphical utilities you can use to control the connected mobile device (notably [KDE Connect \(https://community.kde.org/KDEConnect\)](https://community.kde.org/KDEConnect) and [GSConnect \(https://extensions.gnome.org/extension/1319/gconnect/\)](https://extensions.gnome.org/extension/1319/gconnect/)).

28 Using NetworkManager

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.

28.1 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.

28.2 Enabling or Disabling NetworkManager

On laptop computers, NetworkManager is enabled by default. However, it can be at any time enabled or disabled in the YaST Network Settings module.

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 28.3, “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 IP address.

Find a detailed description of the network configuration with YaST in [Section 13.4, “Configuring a Network Connection with YaST”](#).

28.3 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.



Tip: NetworkManager Connection Editor

In previous openSUSE Leap releases, network connections were configured using an application called *NetworkManager Connection Editor*. This is no longer installed by default, because *GNOME Control Center* has fully replaced its configuration capabilities.

If you still need to use NetworkManager Connection Editor to configure network connections, install the `NetworkManager-connection-editor` package manually:

```
tux > sudo zypper install NetworkManager-connection-editor
```

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 connections. In a secured environment, some options may be locked or require root permission. Ask your system administrator for details.

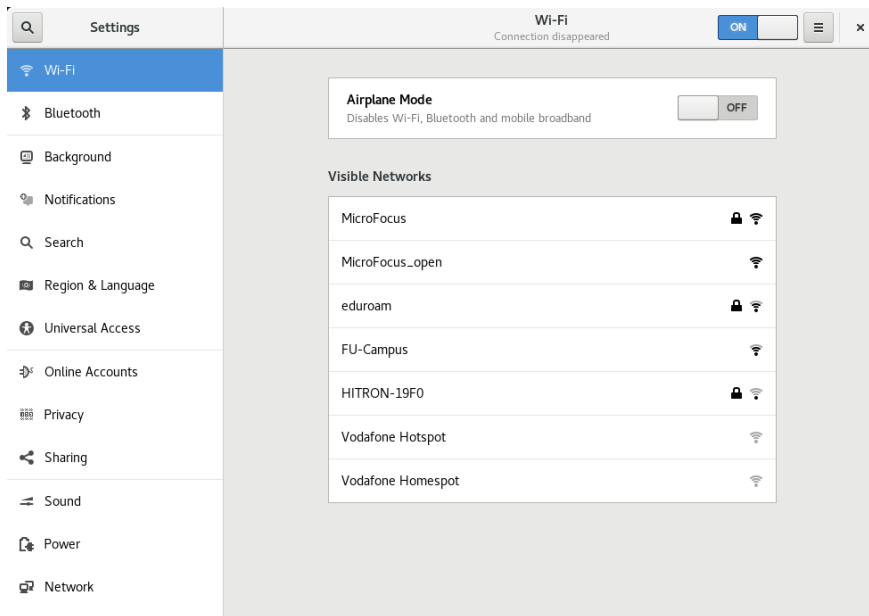


FIGURE 28.1: GNOME NETWORK CONNECTIONS DIALOG

PROCEDURE 28.1: ADDING AND EDITING CONNECTIONS

1. Open the NetworkManager configuration dialog.
2. To add a Connection:
 - a. Click the + icon in the lower left corner.
 - b. Select your preferred connection type and follow the instructions.
 - 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. Select the entry to edit.
 - b. Click the gear icon to open the *Connection Settings* dialog.

- c. Insert your changes and click *Apply* to save them.
- d. To make your connection available as a system connection go to the *Identity* tab and set the check box *Make available to other users*. For more information about user and system connections, see [Section 28.4.1, "User and System Connections"](#).

28.3.1 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 change the connection details or to switch it off.
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.

28.3.2 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 28.2: 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 28.3: 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 will remain 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.

28.3.3 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.

28.3.4 NetworkManager and VPN

NetworkManager supports several Virtual Private Network (VPN) technologies. For each technology, openSUSE Leap comes with a base package providing the generic support for NetworkManager. In addition to 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 28.4: 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:

| | |
|-------------------------|--|
| <i>Gateway</i> | The remote endpoint of the VPN server |
| <i>User name</i> | The user (only available when you have selected <i>Password with Certificates (TLS)</i>) |
| <i>Password</i> | The password for the user (only available when you have selected <i>Password with Certificates (TLS)</i>) |
| <i>User Certificate</i> | <u>/etc/openvpn/client1.crt</u> |
| <i>CA Certificate</i> | <u>/etc/openvpn/ca.crt</u> |

| | |
|--------------------|---------------------------------|
| <i>Private Key</i> | <u>/etc/openvpn/client1.key</u> |
|--------------------|---------------------------------|

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*.

28.4 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.

28.4.1 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 they don't have 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 checkbox, *Make available to other users*. For information on how to configure user or system connections with NetworkManager, refer to [Section 28.3, "Configuring Network Connections"](#).

28.4.2 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.

NetworkManager can also retrieve its certificates for secure connections (for example, encrypted wired, wireless or VPN connections) from the certificate store. For more information, refer to *Book "Security and Hardening Guide", Chapter 21 "Certificate Store"*.

28.5 Frequently Asked Questions

In the following, find some frequently asked questions about configuring special network options with NetworkManager.

Q: 1. *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.

Q: 2. *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 28.3, "Configuring Network Connections"](#). Choose the wireless connection you want to modify and click *Edit*. On the *Wireless* tab, enter the BSSID.

Q: 3. *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 28.3, “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 *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 *DCHP Server* is not already shown as *Allowed Service*, select *DCHP Server* from *Services to Allow* and click *Add*. Confirm your changes in YaST.

Q: 4. *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 28.3, “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.

Q: 5. *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 28.4.1, “User and System Connections”](#).

28.6 Troubleshooting

Connection problems can occur. Some 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 applet 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 28.2, “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 for some reason, 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 28.3.4, “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`.

28.7 For More Information

More information about NetworkManager can be found on the following Web sites and directories:

NetworkManager Project Page

<http://projects.gnome.org/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/](#).

29 Power Management

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.

29.1 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 very 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 prior to the suspend is restored. Some 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.

29.2 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 11, journalctl: Query the systemd Journal](#) for more information on viewing the journal log messages. See [Section 29.2.2, “Troubleshooting”](#) for more information about troubleshooting ACPI problems.

29.2.1 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. Usually, 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, some 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 11 "Power Management"*.

29.2.2 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

Some 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 29.4, “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 will only take the issue seriously if they realize that an adequate number of their customers use Linux.

29.2.2.1 For More Information

- <http://tldp.org/HOWTO/ACPI-HOWTO/>  (detailed ACPI HOWTO, contains DSDT patches)
- <http://www.acpi.info>  (Advanced Configuration & Power Interface Specification)
- <http://acpi.sourceforge.net/dsdt/index.php>  (DSDT patches by Bruno Ducrot)

29.3 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 some of the following methods, using the `hdparm` command.

It can be used to modify various 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, numerous 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: Impairment of the Data Integrity

Changes to the `pdflush` daemon settings endanger the 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 openSUSE Leap these technologies are controlled by `laptop-mode-tools`.

29.4 Troubleshooting

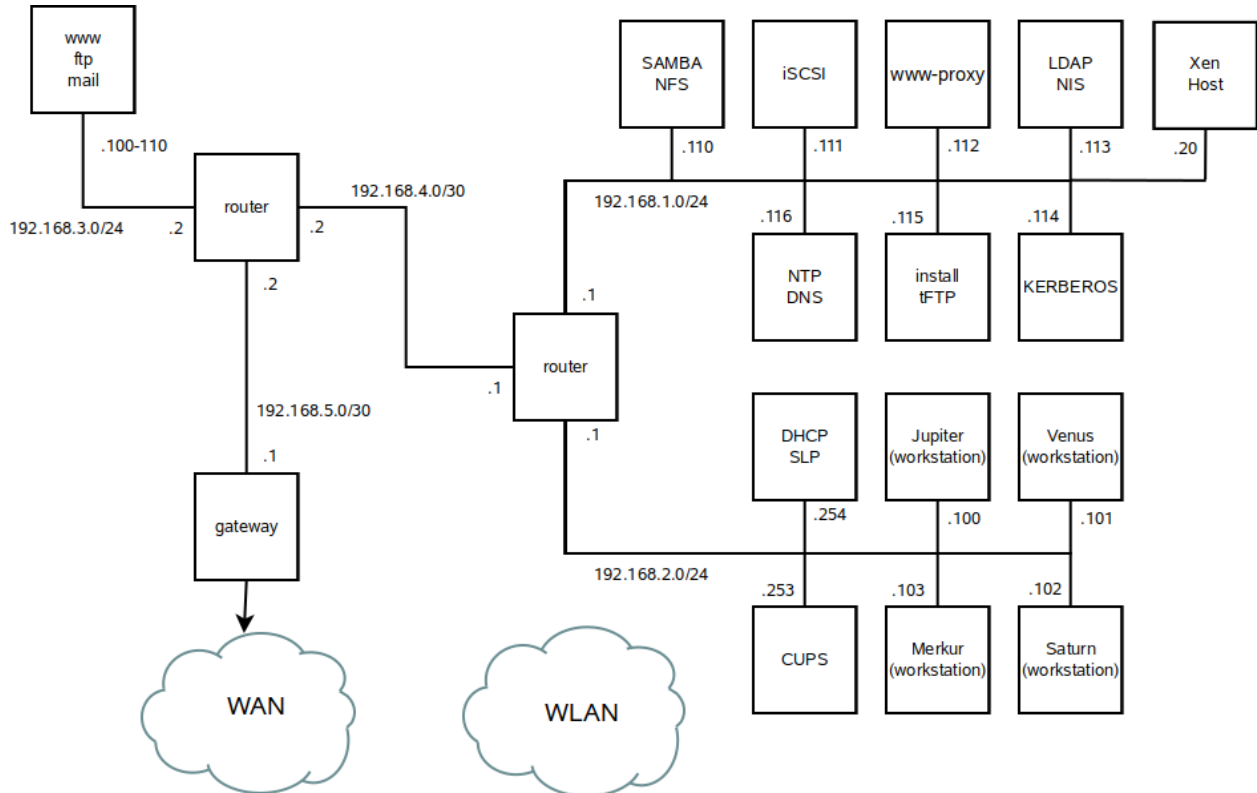
All error messages and alerts are logged in the system journal, which can be queried with the command `journalctl` (see [Chapter 11, journalctl: Query the systemd Journal](#) for more information). The following sections cover the most common problems.

29.4.1 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/>.

A An Example Network

This example network is used across all network-related chapters of the openSUSE® Leap documentation.



B GNU Licenses

This appendix contains the GNU Free Documentation License version 1.2.

GNU Free Documentation License

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

0. PREAMBLE

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

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

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

1. APPLICABILITY AND DEFINITIONS

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

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

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

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

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

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

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or

XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

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

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

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

2. VERBATIM COPYING

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

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

3. COPYING IN QUANTITY

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

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

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

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

4. MODIFICATIONS

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

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

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

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

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

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

5. COMBINING DOCUMENTS

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

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

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

6. COLLECTIONS OF DOCUMENTS

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

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

7. AGGREGATION WITH INDEPENDENT WORKS

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

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

8. TRANSLATION

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

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

9. TERMINATION

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

10. FUTURE REVISIONS OF THIS LICENSE

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

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

ADDENDUM: How to use this License for your documents

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

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being LIST  
THEIR TITLES, with the  
Front-Cover Texts being LIST, and with the  
Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.