



Reference

openSUSE Leap 42.1



Reference

openSUSE Leap 42.1

Publication Date: November 05, 2018

SUSE LLC
10 Canal Park Drive
Suite 200
Cambridge MA 02141
USA

<https://www.suse.com/documentation> 

Copyright © 2006– 2018 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 or Novell trademarks, see the Novell Trademark and Service Mark list <http://www.novell.com/company/legal/trademarks/tmlist.html> . All other third party trademarks are the property of their respective owners. A trademark symbol (®, ™ etc.) denotes a SUSE or Novell trademark; an asterisk (*) denotes a third party trademark. 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 **xvi**

- I **ADVANCED ADMINISTRATION 1**
- 1 YaST in Text Mode 2**
 - 1.1 Navigation in Modules 3
 - 1.2 Restriction of Key Combinations 4
 - 1.3 YaST Command Line Options 5
 - Starting the Individual Modules 5 • Installing Packages from the Command Line 5 • Command Line Parameters of the YaST Modules 6
- 2 Managing Software with Command Line Tools 7**
 - 2.1 Using Zypper 7
 - General Usage 7 • Installing and Removing Software with Zypper 8 • Updating Software with Zypper 12 • Identifying Processes and Services Using Deleted Files 15 • Managing Repositories with Zypper 17 • Querying Repositories and Packages with Zypper 19 • Configuring Zypper 20 • Troubleshooting 20 • Zypper Rollback Feature on Btrfs File System 20 • For More Information 21
 - 2.2 RPM—the Package Manager 21
 - Verifying Package Authenticity 22 • Managing Packages: Install, Update, and Uninstall 22 • Delta RPM Packages 23 • RPM Queries 24 • Installing and Compiling Source Packages 27 • Compiling RPM Packages with build 29 • Tools for RPM Archives and the RPM Database 29

3 System Recovery and Snapshot Management with Snapper 30

- 3.1 Default Setup 30
 - Types of Snapshots 31 • Directories That Are Excluded from Snapshots 32 • Customizing the Setup 33
- 3.2 Using Snapper to Undo Changes 36
 - Undoing YaST and Zypper Changes 37 • Using Snapper to Restore Files 42
- 3.3 System Rollback by Booting from Snapshots 44
 - Accessing and Identifying Snapshot Boot Entries 46 • Limitations 47
- 3.4 Creating and Modifying Snapper Configurations 48
 - Managing Existing Configurations 49
- 3.5 Manually Creating and Managing Snapshots 54
 - Snapshot Metadata 54 • Creating Snapshots 56 • Modifying Snapshot Metadata 57 • Deleting Snapshots 58
- 3.6 Frequently Asked Questions 59

4 Remote Access with VNC 60

- 4.1 One-time VNC Sessions 60
 - Available Configurations 61 • Initiating a One-time VNC Session 61 • Configuring One-time VNC Sessions 61
- 4.2 Persistent VNC Sessions 62
 - Connecting to a Persistent VNC Session 64 • Configuring Persistent VNC Sessions 64

5 Advanced Disk Setup 65

- 5.1 Using the YaST Partitioner 65
 - Partition Types 67 • Creating a Partition 67 • Editing a Partition 71 • Expert Options 73 • Advanced Options 73 • More Partitioning Tips 73 • Partitioning and LVM 76
- 5.2 LVM Configuration 77
 - The Logical Volume Manager 77 • LVM Configuration with YaST 79

- 5.3 Soft RAID Configuration 82
 - Soft RAID Configuration with YaST 84 • Troubleshooting 86 • For More Information 86

6 Installing Multiple Kernel Versions 87

- 6.1 Enabling and Configuring Multiversion Support 87
 - Automatically Deleting Unused Kernels 88
- 6.2 Installing/Removing Multiple Kernel Versions with YaST 89
- 6.3 Installing/Removing Multiple Kernel Versions with Zypper 90

7 GNOME Configuration for Administrators 92

- 7.1 Starting Applications Automatically 92
- 7.2 Automounting and Managing Media Devices 92
- 7.3 Changing Preferred Applications 92
- 7.4 Adding Document Templates 93
- 7.5 For More Information 93

II SYSTEM 94

8 32-Bit and 64-Bit Applications in a 64-Bit System Environment 95

- 8.1 Runtime Support 95
- 8.2 Software Development 96
- 8.3 Software Compilation on Biarch Platforms 97
- 8.4 Kernel Specifications 98

9 Booting a Linux System 100

- 9.1 The Linux Boot Process 100
- 9.2 `initramfs` 101
- 9.3 `init` on `initramfs` 103

10 The systemd Daemon 105

- 10.1 The systemd Concept 105
 - What Is systemd 105 • Unit File 106
- 10.2 Basic Usage 107
 - Managing Services in a Running System 107 • Permanently Enabling/Disabling Services 109
- 10.3 System Start and Target Management 110
 - Targets vs. Runlevels 111 • Debugging System Start-Up 114 • System V Compatibility 117
- 10.4 Managing Services with YaST 118
- 10.5 Customization of systemd 119
 - Customizing Service Files 119 • Creating “Drop-in” Files 120 • Creating Custom Targets 120
- 10.6 Advanced Usage 121
 - Cleaning Temporary Directories 121 • System Log 122 • Snapshots 122 • Loading Kernel Modules 122 • Performing Actions Before Loading A Service 123 • Kernel Control Groups (cgroups) 124 • Terminating Services (Sending Signals) 125 • Debugging Services 125
- 10.7 More information 127

11 journalctl: Query the systemd Journal 128

- 11.1 Making the Journal Persistent 128
- 11.2 **journalctl** Useful Switches 129
- 11.3 Filtering the Journal Output 130
 - Filtering Based on a Boot Number 130 • Filtering Based on Time Interval 130 • Filtering Based on Fields 131
- 11.4 Investigating systemd Errors 132

- 11.5 Journald Configuration 133
 - Changing the Journal Size Limit 133 • Forwarding the Journal to /dev/ttyX 134 • Forwarding the Journal to Syslog Facility 134
- 11.6 Using YaST to Filter the systemd Journal 134
- 12 The Boot Loader GRUB 2 136**
 - 12.1 Main Differences between GRUB Legacy and GRUB 2 136
 - 12.2 Configuration File Structure 136
 - The File /boot/grub2/grub.cfg 137 • The File /etc/default/grub 138 • Scripts in /etc/grub.d 141 • Mapping between BIOS Drives and Linux Devices 142 • Editing Menu Entries during the Boot Procedure 142 • Setting a Boot Password 144
 - 12.3 Configuring the Boot Loader with YaST 145
 - Modifying the Boot Loader Location 146 • Adjusting the Disk Order 146 • Configuring Advanced Options 147
 - 12.4 Differences in Terminal Usage on z Systems 149
 - Limitations 149 • Key Combinations 150
 - 12.5 Helpful GRUB 2 Commands 152
 - 12.6 More Information 153
- 13 Basic Networking 154**
 - 13.1 IP Addresses and Routing 157
 - IP Addresses 157 • Netmasks and Routing 157
 - 13.2 IPv6—The Next Generation Internet 159
 - Advantages 160 • Address Types and Structure 161 • Coexistence of IPv4 and IPv6 165 • Configuring IPv6 166 • For More Information 167
 - 13.3 Name Resolution 168
 - 13.4 Configuring a Network Connection with YaST 169
 - Configuring the Network Card with YaST 169

- 13.5 NetworkManager 180
 - NetworkManager and **wicked** 180 • NetworkManager Functionality and Configuration Files 181 • Controlling and Locking Down NetworkManager Features 181
- 13.6 Configuring a Network Connection Manually 182
 - The **wicked** Network Configuration 182 • Configuration Files 188 • Testing the Configuration 198 • Unit Files and Start-Up Scripts 201
- 13.7 Basic Router Setup 203
- 13.8 Setting Up Bonding Devices 204
 - Hotplugging of Bonding Slaves 206
- 13.9 Software-Defined Networking with Open vSwitch 207
 - Advantages of Open vSwitch 208 • Installing Open vSwitch 208 • Overview of Open vSwitch Daemons and Utilities 209 • Creating a Bridge with Open vSwitch 210 • Using Open vSwitch Directly with KVM 211 • Using Open vSwitch with libvirt 212 • For More Information 213
- 14 UEFI (Unified Extensible Firmware Interface) 214**
- 14.1 Secure Boot 214
 - Implementation on SUSE Linux Enterprise 215 • MOK (Machine Owner Key) 219 • Booting a Custom Kernel 219 • Using Non-Inbox Drivers 221 • Features and Limitations 222
- 14.2 For More Information 223
- 15 Special System Features 224**
- 15.1 Information about Special Software Packages 224
 - The bash Package and /etc/profile 224 • The cron Package 225 • Stopping Cron Status Messages 226 • Log Files: Package logrotate 226 • The locate Command 227 • The ulimit Command 227 • The free Command 229 • Man Pages and Info Pages 229 • Selecting Man Pages Using the **man** Command 229 • Settings for GNU Emacs 230
- 15.2 Virtual Consoles 231

- 15.3 Keyboard Mapping 231
- 15.4 Language and Country-Specific Settings 232
 - Some Examples 233 • Locale Settings in ~/.i18n 234 • Settings for Language Support 234 • For More Information 235
- 16 Dynamic Kernel Device Management with udev 236**
 - 16.1 The /dev Directory 236
 - 16.2 Kernel uevents and udev 236
 - 16.3 Drivers, Kernel Modules and Devices 237
 - 16.4 Booting and Initial Device Setup 237
 - 16.5 Monitoring the Running udev Daemon 238
 - 16.6 Influencing Kernel Device Event Handling with udev Rules 239
 - Using Operators in udev Rules 241 • Using Substitutions in udev Rules 242 • Using udev Match Keys 243 • Using udev Assign Keys 244
 - 16.7 Persistent Device Naming 245
 - 16.8 Files used by udev 246
 - 16.9 For More Information 247
- III SERVICES 248**
- 17 SLP 249**
 - 17.1 The SLP Front-End **slptool** 249
 - 17.2 Providing Services via SLP 250
 - Setting up an SLP Installation Server 252
 - 17.3 For More Information 252
- 18 Time Synchronization with NTP 253**
 - 18.1 Configuring an NTP Client with YaST 253
 - Basic Configuration 253 • Changing Basic Configuration 254
 - 18.2 Manually Configuring NTP in the Network 257

- 18.3 Dynamic Time Synchronization at Runtime 257
- 18.4 Setting Up a Local Reference Clock 258
- 19 The Domain Name System 259**
 - 19.1 DNS Terminology 259
 - 19.2 Installation 260
 - 19.3 Configuration with YaST 260
 - Wizard Configuration 260 • Expert Configuration 263
 - 19.4 Starting the BIND Name Server 271
 - 19.5 The /etc/named.conf Configuration File 273
 - Important Configuration Options 274 • Logging 275 • Zone Entries 276
 - 19.6 Zone Files 277
 - 19.7 Dynamic Update of Zone Data 281
 - 19.8 Secure Transactions 281
 - 19.9 DNS Security 283
 - 19.10 For More Information 283
- 20 DHCP 284**
 - 20.1 Configuring a DHCP Server with YaST 285
 - Initial Configuration (Wizard) 285 • DHCP Server Configuration (Expert) 290
 - 20.2 DHCP Software Packages 295
 - 20.3 The DHCP Server dhcpd 296
 - Clients with Fixed IP Addresses 298 • The openSUSE Leap Version 299
 - 20.4 For More Information 299
- 21 Samba 300**
 - 21.1 Terminology 300
 - 21.2 Installing a Samba Server 301

- 21.3 Starting and Stopping Samba 302
- 21.4 Configuring a Samba Server 302
 - Configuring a Samba Server with YaST 302 • Configuring the Server Manually 305
- 21.5 Configuring Clients 309
 - Configuring a Samba Client with YaST 309
- 21.6 Samba as Login Server 309
- 21.7 Samba Server in the Network with Active Directory 310
- 21.8 Advanced Topics 312
 - Transparent File Compression on Btrfs 312 • Snapshots 313
- 21.9 For More Information 321
- 22 Sharing File Systems with NFS 322**
- 22.1 Terminology 322
- 22.2 Installing NFS Server 323
- 22.3 Configuring NFS Server 323
 - Exporting File Systems with YaST 324 • Exporting File Systems Manually 325 • NFS with Kerberos 327
- 22.4 Configuring Clients 328
 - Importing File Systems with YaST 328 • Importing File Systems Manually 329 • Parallel NFS (pNFS) 330
- 22.5 For More Information 332
- 23 On-Demand Mounting with Autofs 333**
- 23.1 Installation 333
- 23.2 Configuration 333
 - The Master Map File 333 • Map Files 335
- 23.3 Operation and Debugging 336
 - Controlling the autofs Service 336 • Debugging the Automounter Problems 337

- 23.4 Auto-Mounting an NFS Share 338
- 23.5 Advanced Topics 339
 - /net Mount Point 339 • Using Wild Cards to Auto-Mount Subdirectories 339 • Auto-Mounting CIFS File System 340
- 24 The Apache HTTP Server 341**
- 24.1 Quick Start 341
 - Requirements 341 • Installation 342 • Start 342
- 24.2 Configuring Apache 343
 - Apache Configuration Files 343 • Configuring Apache Manually 346 • Configuring Apache with YaST 351
- 24.3 Starting and Stopping Apache 357
- 24.4 Installing, Activating, and Configuring Modules 359
 - Module Installation 360 • Activation and Deactivation 360 • Base and Extension Modules 360 • Multiprocessing Modules 363 • External Modules 364 • Compilation 365
- 24.5 Getting CGI Scripts to Work 366
 - Apache Configuration 367 • Running an Example Script 367 • CGI Troubleshooting 368
- 24.6 Setting Up a Secure Web Server with SSL 368
 - Creating an SSL Certificate 369 • Configuring Apache with SSL 373
- 24.7 Running Multiple Apache Instances on the Same Server 375
- 24.8 Avoiding Security Problems 378
 - Up-to-Date Software 378 • DocumentRoot Permissions 378 • File System Access 379 • CGI Scripts 379 • User Directories 379
- 24.9 Troubleshooting 380
- 24.10 For More Information 381
 - Apache 2.4 381 • Apache Modules 381 • Development 382 • Miscellaneous Sources 382

25 Setting up an FTP Server with YaST 383

- 25.1 Starting the FTP Server 384
- 25.2 FTP General Settings 384
- 25.3 FTP Performance Settings 385
- 25.4 Authentication 386
- 25.5 Expert Settings 386
- 25.6 For More Information 386

26 The Proxy Server Squid 387

- 26.1 Some Facts about Proxy Caches 387
 - Squid and Security 388 • Multiple Caches 388 • Caching Internet Objects 389
- 26.2 System Requirements 389
 - Hard Disks 390 • Size of the Disk Cache 390 • RAM 390 • CPU 391
- 26.3 Starting Squid 391
 - Commands for Starting and Stopping Squid 391 • Local DNS Server 392
- 26.4 The /etc/squid/squid.conf Configuration File 393
 - General Configuration Options (Selection) 394 • Options for Access Controls 396
- 26.5 Configuring a Transparent Proxy 398
 - Configuration Options in /etc/squid/squid.conf 399 • Firewall Configuration with SuSEFirewall2 399
- 26.6 cachemgr.cgi 401
 - Setup 401 • Cache Manager ACLs in /etc/squid/squid.conf 401 • Viewing the Statistics 402
- 26.7 squidGuard 402
- 26.8 Cache Report Generation with Calamaris 403
- 26.9 For More Information 404

IV MOBILE COMPUTERS 405

27 Mobile Computing with Linux 406

27.1 Laptops 406

Power Conservation 406 • Integration in Changing Operating Environments 407 • Software Options 409 • Data Security 414

27.2 Mobile Hardware 415

27.3 Cellular Phones and PDAs 416

27.4 For More Information 416

28 Using NetworkManager 417

28.1 Use Cases for NetworkManager 417

28.2 Enabling or Disabling NetworkManager 417

28.3 Configuring Network Connections 418

Managing Wired Network Connections 420 • Managing Wireless Network Connections 420 • Configuring Your Wi-Fi/Bluetooth Card as an Access Point 421 • NetworkManager and VPN 421

28.4 NetworkManager and Security 422

User and System Connections 422 • Storing Passwords and Credentials 423

28.5 Frequently Asked Questions 423

28.6 Troubleshooting 425

28.7 For More Information 425

29 Power Management 427

29.1 Power Saving Functions 427

29.2 Advanced Configuration and Power Interface (ACPI) 428

Controlling the CPU Performance 429 • Troubleshooting 429

29.3 Rest for the Hard Disk 431

29.4 Troubleshooting 432

CPU Frequency Does Not Work 433

29.5	For More Information	433
A	An Example Network	434
B	GNU Licenses	435
B.1	GNU Free Documentation License	435

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 roll-backs 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.

Many chapters in this manual contain links to additional documentation resources. These include additional documentation that is available on the system, as well as documentation available on the Internet.

For an overview of the documentation available for your product and the latest documentation updates, refer to <http://doc.opensuse.org/> or to the following section.

1 Available Documentation

We provide HTML and PDF versions of our books in different languages. The following manuals for users and administrators are 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”

AutoYaST is a system for installing one or more openSUSE Leap systems automatically and without user intervention, using an AutoYaST profile that contains installation and configuration data. The manual guides you through the basic steps of auto-installation: preparation, installation, and configuration.

Book “Security 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 or the auditing system that reliably collects information about any security-relevant events.

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.

Find HTML versions of most product manuals in your installed system under `/usr/share/doc/manual` or in the help centers of your desktop. Find the latest documentation updates at <http://doc.opensuse.org/>  where you can download PDF or HTML versions of the manuals for your product.

2 Feedback

Several feedback channels are available:

Bugs and Enhancement Requests

For services and support options available for your product, refer to <http://www.suse.com/support/>.

To report bugs for a product component, go to <https://scc.suse.com/support/requests>, log in, and click *Create New*.

User Comments

We want to hear your comments about and suggestions for this manual and the other documentation included with this product. Use the User Comments feature at the bottom of each page in the online documentation or go to <http://www.suse.com/doc/feedback.html> and enter your comments there.

Mail

For feedback on the documentation of this product, you can also send a mail to doc-team@suse.de. Make sure to include the document title, the product version and the publication date of the documentation. To report errors or suggest enhancements, provide a concise description of the problem and refer to the respective section number and page (or URL).

3 Documentation Conventions

The following typographical conventions are used in this manual:

- /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
- 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.

4 About the Making of This Documentation

This documentation is written in SUSEDoc, a subset of DocBook 5 (<http://www.docbook.org> [↗]). The XML source files were validated by **jing** (see <https://code.google.com/p/jing-trang/> [↗]), processed by **xsltproc**, and converted into XSL-FO using a customized version of Norman Walsh's stylesheets. The final PDF is formatted through FOP from Apache Software Foundation (<https://xmlgraphics.apache.org/fop>) [↗]. The open source tools and the environment used to build this documentation are provided by the DocBook Authoring and Publishing Suite (DAPS). The project's home page can be found at <https://github.com/openSUSE/daps> [↗].

The XML source code of this documentation can be found at <https://github.com/SUSE/doc-sle> [↗].

5 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.

6 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 7
- 3 System Recovery and Snapshot Management with Snapper 30
- 4 Remote Access with VNC 60
- 5 Advanced Disk Setup 65
- 6 Installing Multiple Kernel Versions 87
- 7 GNOME Configuration for Administrators 92

1 YaST in Text Mode

This section is intended for system administrators and experts who do not run an X server on their systems and depend on the text-based installation tool. It provides basic information about starting and operating YaST in text mode.

YaST in text mode uses the ncurses library to provide an easy pseudo-graphical user interface. The ncurses library is installed by default. The minimum supported size of the terminal emulator in which to run YaST is 80x25 characters.

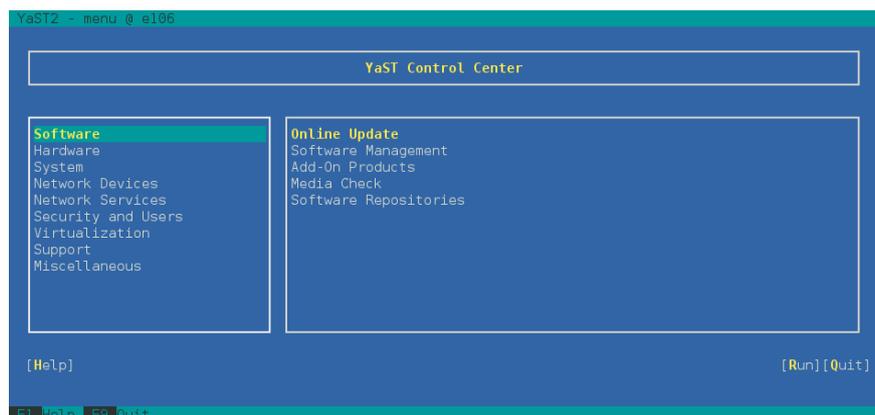


FIGURE 1.1: MAIN WINDOW OF YAST IN TEXT MODE

When you start YaST in text mode, the YaST control center appears (see [Figure 1.1](#)). The main window consists of three areas. The left frame features the categories to which the various modules belong. This frame is active when YaST is started and therefore it is marked by a bold white border. The active category is highlighted. The right frame provides an overview of the modules available in the active category. The bottom frame contains the buttons for *Help* and *Quit*.

When you start the YaST control center, the category *Software* is selected automatically. Use `↓` and `↑` to change the category. To select a module from the category, activate the right frame with `→` and then use `↓` and `↑` to select the module. Keep the arrow keys pressed to scroll through the list of available modules. The selected module is highlighted. Press `Enter` to start the active module.

Various buttons or selection fields in the module contain a highlighted letter (yellow by default). Use `Alt-highlighed_letter` to select a button directly instead of navigating there with `→`. Exit the YaST control center by pressing `Alt-Q` or by selecting *Quit* and pressing `Enter`.



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 1.2, "Restriction of Key Combinations"* for information about possible exceptions.

Navigation among Buttons and Selection Lists

Use **→** to navigate among the buttons and frames containing selection lists. To navigate in reverse order, use **Alt-→** or **Shift-→** combinations.

Navigation in Selection Lists

Use the arrow keys (**↑** and **↓**) to navigate among the individual elements in an active frame containing a selection list. If individual entries within a frame exceed its width, use **Shift-→** or **Shift-←** to scroll horizontally to the right and left. Alternatively, use **Ctrl-E** or **Ctrl-A**. This combination can also be used if using **→** or **←** results in changing the active frame or the current selection list, as in the control center.

Buttons, Radio Buttons, and Check Boxes

To select buttons 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 F keys (**F1** through **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 Navigation Tree in ncurses Mode

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 `Space` to open or close tree items. In ncurses mode, `Enter` must be pressed after a selection in the navigation tree to show the selected dialog. This is an intentional behavior to save time consuming redraws when browsing through the navigation tree.

Selecting Software in the Software Installation Module

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

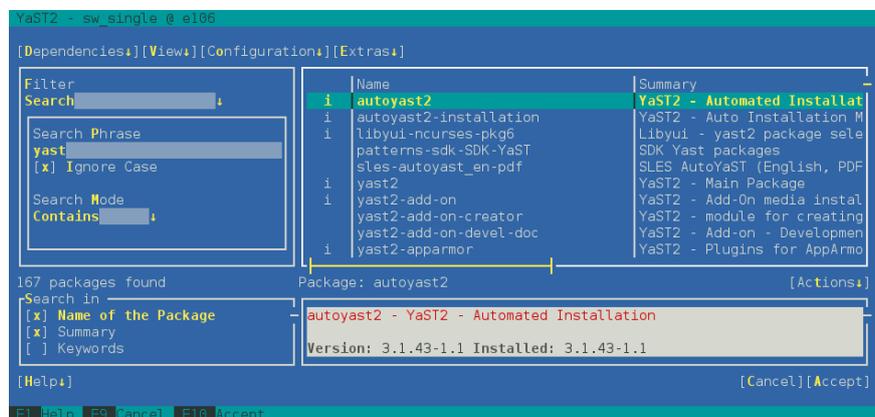


FIGURE 1.2: THE SOFTWARE INSTALLATION MODULE

1.2 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.

Replacing `Alt` with `Esc`

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

Backward and Forward Navigation with `Ctrl-F` and `Ctrl-B`

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

Restriction of Function Keys

The F keys are also used for functions. Certain function keys might be occupied 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 pure text console.

1.3 YaST Command Line Options

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

```
yast -h
```

1.3.1 Starting the Individual Modules

To save time, the individual YaST modules can be started directly. To start a module, enter:

```
yast <module_name>
```

View a list of all module names available on your system with `yast -l` or `yast --list`. Start the network module, for example, with `yast lan`.

1.3.2 Installing Packages from the Command Line

If you know a package name and the package is provided by any of your active installation repositories, you can use the command line option `-i` to install the package:

```
yast -i <package_name>
```

or

```
yast --install <package_name>
```

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

If you need a command line based software management utility with functionality beyond what YaST provides, consider using Zypper. This utility uses the same software management library that is also the foundation for the YaST package manager. The basic usage of Zypper is covered in [Section 2.1, "Using Zypper"](#).

1.3.3 Command Line Parameters of the YaST Modules

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

```
yast <module_name> help
```

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

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

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 9 "Installing or Removing Software", Section 9.1 "Definition of Terms"*.

2.1 Using Zypper

Zypper is a command line package manager for installing, updating and removing packages as well as for managing 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*. 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 type:

```
zypper patch
```

Additionally, you can choose from one or more global options by typing them immediately before the command. For example, --non-interactive means running the command without asking anything (automatically applying the default answers):

```
zypper --non-interactive patch
```

To use the options specific to a particular command, type them right after the command. For example, --auto-agree-with-licenses means applying all needed patches to the system without asking to confirm any licenses (they will automatically be accepted):

```
zypper patch --auto-agree-with-licenses
```

Some commands require one or more arguments. When using the install command, for example, you need to specify which package(s) to install:

```
zypper install mplayer
```

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

```
zypper search -t pattern
```

You can combine all of the above. For example, the following command will install the `aspell-de` and `aspell-fr` packages from the `factory` repository while being verbose:

```
zypper -v install --from factory aspell-de aspell-fr
```

The `--from` option makes sure to keep all repositories enabled (for solving any dependencies) while requesting the package from the specified repository.

Most Zypper commands have a `dry-run` option that does a simulation of the given command. It can be used for test purposes.

```
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.

```
zypper --userdata string patch
```

2.1.2 Installing and Removing Software with Zypper

To install or remove packages use the following commands:

```
zypper install package_name  
zypper remove package_name
```

Zypper knows various ways to address packages for the install and remove commands.

by the exact package name (and version number)

```
zypper install MozillaFirefox
```

or

```
zypper install MozillaFirefox-3.5.3
```

by repository alias and package name

```
zypper install mozilla:MozillaFirefox
```

Where mozilla is the alias of the repository from which to install.

by package name using wild cards

The following command will install all packages that have names starting with “Moz”. Use with care, especially when removing packages.

```
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:

```
zypper remove '*-debuginfo'
```

by capability

For example, if you want to install a Perl module without knowing the name of the package, capabilities come in handy:

```
zypper install firefox
```

by capability and/or architecture and/or version

Together with a capability you can specify an architecture (such as `x86_64`) and/or a version. The version must be preceded by an operator: `<` (lesser than), `<=` (lesser than or equal), `=` (equal), `>=` (greater than or equal), `>` (greater than).

```
zypper install 'firefox.x86_64'  
zypper install 'firefox>=3.5.3'  
zypper install 'firefox.x86_64>=3.5.3'
```

by path to the RPM file

You can also specify a local or remote path to a package:

```
zypper install /tmp/install/MozillaFirefox.rpm
zypper install http://download.opensuse.org/repositories/mozilla/SUSE_Factory/
x86_64/MozillaFirefox-3.5.3-1.3.x86_64.rpm
```

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

```
zypper install emacs -vim
```

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

```
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 `--`:

```
zypper install -emacs +vim      # Wrong
zypper install vim -emacs       # Correct
zypper install -- -emacs +vim   # same as above
zypper remove emacs +vim       # same as above
```

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:

```
zypper rm package_name --clean-deps
```

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 in the following:

```
zypper --non-interactive install package_name
```

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



Warning: Do not Remove Mandatory System Packages

Do not remove packages such as `glibc`, `zypper`, `kernel`, or similar packages. These packages are mandatory for the system and, if removed, may cause the system to become unstable or stop working altogether.

2.1.2.1 Installing or Downloading Source Packages

If you want to install the corresponding source package of a package, use:

```
zypper source-install package_name
```

That command will also install the build dependencies of the specified package. If you do not want this, add the switch `-D`. To install only the build dependencies use `-d`.

```
zypper source-install -D package_name # source package only
zypper source-install -d package_name # build dependencies only
```

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 2.1.5, “Managing Repositories with Zypper”](#) for details on repository management.

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

```
zypper search -t srcpackage
```

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

```
zypper source-download
```

The default download directory is `/var/cache/zypper/source-download`. You can change it using the `--directory` option. To only show missing or extraneous packages without downloading or deleting anything, use the `--status` option. To delete extraneous source packages, use the `--delete` option. To disable deleting, use the `--no-delete` option.

2.1.2.2 Installing Packages from Disabled Repositories

Normally you can only install 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:

```
zypper --plus-content debug install "debuginfo(build-id)=eb844a5c20c70a59fc693cd1061f851fb7d046f4"
```

The `build-id` string is reported by `gdb` for missing debuginfo packages.

2.1.2.3 Utilities

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

```
zypper verify
```

In addition to dependencies that must be fulfilled, some packages “recommend” other packages. These recommended packages are only installed if actually available and installable. In case recommended packages were made available after the recommending package has been installed (by adding additional packages or hardware), use the following command:

```
zypper install-new-recommends
```

This command is very useful after plugging in a webcam 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.

2.1.3 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 12 “Upgrading the System and System Changes”*.

2.1.3.1 Installing Patches

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

```
zypper patch
```

In this case, all patches available in your repositories are checked for relevance and installed, if necessary. After registering your openSUSE Leap installation, an official update repository containing such patches will be added to your system. The above command is all you need to enter to apply them when needed.

If a patch to be installed includes changes that require a system reboot, you will be warned before installing the patch.

Zypper knows three different commands to query for the availability of patches:

zypper patch-check

Lists the number of needed patches (patches, that apply to your system but are not yet installed)

```
tux > sudo zypper patch-check
Loading repository data...
Reading installed packages...
5 patches needed (1 security patch)
```

zypper list-patches

Lists all needed patches (patches, that apply to your system but are not yet installed)

```
tux > sudo zypper list-patches
Loading repository data...
Reading installed packages...

Repository      | Name          | Version | Category | Status | Summary
-----+-----+-----+-----+-----+-----
SLES12-Updates | SUSE-2014-8  | 1       | security | needed | openssl: Update to OpenSSL
1.0.1g
```

zypper patches

Lists all patches available for openSUSE Leap, regardless of whether they are already installed or apply to your installation.

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:

--bugzilla[=number]

Lists all needed patches for Bugzilla issues. Optionally, you can specify a bug number if you only want to list patches for this specific bug.

--cve[=number]

Lists all needed patches for CVE (Common Vulnerabilities and Exposures) issues, or only patches matching a certain CVE number, if specified.

```
zypper list-patches --cve
```

Lists all 'needed' patches with a CVE number assigned.

```
zypper list-patches --all --cve
Issue | No.          | Patch          | Category | Severity | Status
```

```
-----+-----+-----+-----+-----+-----+-----
cve   | CVE-2015-0287 | SUSE-SLE-Module.. | recommended | moderate | needed
cve   | CVE-2014-3566 | SUSE-SLE-SERVER.. | recommended | moderate | not needed
[...]
```

Lists all patches with a CVE number assigned.

```
zypper list-patches --all --cve=CVE-2015-4477,CVE-2014-3639
Issue | No.          | Patch      | Category  | Severity  | Inter.  | Status
-----+-----+-----+-----+-----+-----+-----
cve   | CVE-2014-3639 | 2014-558  | security  | moderate  | reboot  | not needed
cve   | CVE-2014-3639 | 2014-558  | security  | moderate  | reboot  | not needed
cve   | CVE-2015-4477 | 2015-547  | security  | important | ---     | needed
cve   | CVE-2014-3639 | 3240      | recommended | moderate  | ---     | not needed
cve   | CVE-2014-3639 | 3240      | recommended | moderate  | ---     | not needed
```

Lists 'all' patches with CVE-2015-4477 or CVE-2014-3639.

To install a patch for a specific Bugzilla or CVE issue, use the following commands:

```
zypper patch --bugzilla=number
```

or

```
zypper patch --cve=number
```

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

```
zypper patch --cve=CVE-2010-2713
```

2.1.3.2 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:

```
zypper update
```

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

```
zypper update package_name
zypper install package_name
```

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

```
zypper list-updates
```

Note that this command only lists packages that match the following criteria:

- has the same vendor like the already installed package,
- is provided by repositories with at least the same priority than the already installed package,
- is installable (all dependencies are satisfied).

A list of *all* new available packages (regardless whether installable or not) can be obtained with:

```
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.

2.1.3.3 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:

```
zypper packages --orphaned
```

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

2.1.4 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 show a list of 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:

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/libpthreads-> /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 executethe 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.

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](#).

2.1.5 Managing Repositories with Zypper

All installation or patch commands of Zypper rely on a list of known repositories. To list all repositories known to the system, use the command:

```
zypper repos
```

The result will look similar to the following output:

EXAMPLE 2.1: ZYPPER—LIST OF KNOWN REPOSITORIES

#	Alias	Name	Enabled	Refresh
1	SLEHA-12-GE0	SLEHA-12-GE0	Yes	No
2	SLEHA-12	SLEHA-12	Yes	No
3	SLES12	SLES12	Yes	No

When specifying repositories in various commands, an alias, URI or repository number from the **zypper repos** command output can be used. A repository alias is a short version of the repository name for use in repository handling commands. Note that the repository numbers can change after modifying the list of repositories. The alias will never change by itself.

By default, details such as the URI or the priority of the repository are not displayed. Use the following command to list all details:

```
zypper repos -d
```

2.1.5.1 Adding Repositories

To add a repository, run

```
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_URLs 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.

2.1.5.2 Removing Repositories

If you want to remove a repository from the list, use the command **zypper removerepo** together with the alias or number of the repository you want to delete. For example, to remove the repository SLEHA-12-GE0 from *Example 2.1, “Zypper—List of Known Repositories”*, use one of the following commands:

```
zypper removerepo 1
zypper removerepo "SLEHA-12-GE0"
```

2.1.5.3 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:

```
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
- r: 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:

```
zypper renamerepo 'Mozilla Firefox' firefox
```

2.1.6 Querying Repositories and Packages with Zypper

Zypper offers various methods to query repositories or packages. To get lists of all products, patterns, packages or patches available, use the following commands:

```
zypper products
zypper patterns
zypper packages
zypper patches
```

To query all repositories for certain packages, use `search`. It works on package names, or, optionally, on package summaries and descriptions. String wrapped in `/` are interpreted as regular expressions. By default, the search is not case-sensitive.

Simple search for a package name containing `fire`

```
zypper search "fire"
```

Simple search for the exact package `MozillaFirefox`

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

Also search in package descriptions and summaries

```
zypper search -d fire
```

Only display packages not already installed

```
zypper search -u fire
```

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

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

To search for packages which provide a special capability, use the command `what-provides`. For example, if you want to know which package provides the Perl module `SVN::Core`, use the following command:

```
zypper what-provides 'perl(SVN::Core)'
```

To query single packages, use `info` with an exact package name as an argument. It displays detailed information about a package. To also show what is required/recommended by the package, use the options `--requires` and `--recommends`:

```
zypper info --requires MozillaFirefox
```

The `what-provides package` is similar to `rpm -q --whatprovides package`, 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 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.

2.1.8 Troubleshooting

In case you have problems to access packages from configured repositories (for example, Zypper cannot find a certain package though you know that it exists in one the repositories), it can help to refresh the repositories with:

```
zypper refresh
```

If that does not help, try

```
zypper refresh -fdb
```

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

2.1.9 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` (via script installed by `snapper`) when committing changes to the file system to create appropriate file system snapshots. These snapshots can be used for reverting any changes made by Zypper. See [Chapter 3, System Recovery and Snapshot Management with Snapper](#) for more information.

2.1.10 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, including `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.

Essentially, `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.

While fixing issues in the operating system, you might need to install a Problem Temporary Fix (PTF) into a production system. The packages provided by SUSE are signed against a special PTF key. However, in contrast to SUSE Linux Enterprise 11, this key is not imported by default on SUSE Linux Enterprise 12 systems. To manually import the key, use the following command:

```
rpm --import /usr/share/doc/packages/suse-build-key/suse_ptf_key.asc
```

After importing the key, you can install PTF packages on your system.

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, but `-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 (but 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. Afterwards, be sure to 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* just 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:

```
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:

```
applydeltarpm new.delta.rpm new.rpm
```

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

```
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
<code>-f FILE</code>	Query the package that contains the file <code>FILE</code> (the full path must be specified with <code>FILE</code>)
<code>-s</code>	File list with status information (implies <code>-l</code>)
<code>-d</code>	List only documentation files (implies <code>-l</code>)
<code>-c</code>	List only configuration files (implies <code>-l</code>)

<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                               Relocations: (not relocatable)
Version    : 1.11.4                             Vendor: openSUSE
Release    : 1.70                               Build Date: Sat 01 Aug 2009 09:49:48 CEST
Install Date: Thu 06 Aug 2009 14:53:24 CEST    Build Host: build18
Group      : Productivity/Networking/Web/Utilities   Source RPM:
           wget-1.11.4-1.70.src.rpm
Size       : 1525431                             License: GPL v3 or later
Signature  : RSA/8, Sat 01 Aug 2009 09:50:04 CEST, Key ID b88b2fd43dbdc284
Packager   : http://bugs.opensuse.org
URL        : http://www.gnu.org/software/wget/
Summary    : A Tool for Mirroring FTP and HTTP Servers
Description:
Wget enables you to retrieve WWW documents or FTP files from a server.
This can be done in script files or via the command line.
[...]
```

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, the following command

```
rpm -q -f /bin/rpm /usr/bin/wget
```

results in:

```
rpm-4.8.0-4.3.x86_64
wget-1.11.4-11.18.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 help of 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>5</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):

```
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_RPMDB_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-1.11.4.tar.bz2
/usr/src/packages/SOURCES/wgetrc.patch
/usr/src/packages/SPECS/wget.spec
```

rpmbuild -b X /usr/src/packages/SPECS/wget.spec starts the compilation. X is a wild card for various stages of the build process (see the output of --help or the RPM documentation for details). The following is merely a brief explanation:

-bp

Prepare sources in /usr/src/packages/BUILD: unpack and patch.

-bc

Do the same as -bp, but with additional compilation.

-bi

Do the same as -bp, but with additional installation of the built software. Caution: if the package does not support the BuildRoot feature, you might overwrite configuration files.

-bb

Do the same as -bi, but with the additional creation of the binary package. If the compile was successful, the binary should be in /usr/src/packages/RPMS.

-ba

Do the same as -bb, but with the additional creation of the source RPM. If the compilation was successful, the binary should be in /usr/src/packages/SRPMS.

--short-circuit

Skip some steps.

The binary RPM created can now be installed with `rpm -i` or, preferably, with `rpm -U`. Installation with `rpm` makes it appear in the RPM database.

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`:

```
cd /usr/src/packages/SOURCES/  
mv ../SPECS/wget.spec .  
build --rpms /media/dvd/suse/ wget.spec
```

Subsequently, a minimum environment is established at `/var/tmp/build-root`. The package is built in this environment. Upon completion, the resulting packages are located in `/var/tmp/build-root/usr/src/packages/RPMS`.

The `build` script offers several additional options. For example, cause the script to prefer your own RPMs, omit the initialization of the build environment or limit the `rpm` command to one of the above-mentioned stages. Access additional information with `build --help` and by reading the `build` man page.

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 9 "Installing or Removing Software"*.

3 System Recovery and Snapshot Management with Snapper

Being able to do file system snapshots providing the ability to do rollbacks on Linux is a feature that was often requested in the past. Snapper, in conjunction with the Btrfs file system or thin-provisioned LVM volumes now fills that gap.

Btrfs, a new copy-on-write file system for Linux, supports file system snapshots (a copy of the state of a subvolume at a certain point of time) of subvolumes (one or more separately mountable file systems within each physical partition). Snapshots are also supported on thin-provisioned LVM volumes formatted with XFS, Ext4 or Ext3. Snapper lets you create and manage these snapshots. It comes with a command line and a YaST interface. Starting with SUSE Linux Enterprise Server 12 it is also possible to boot from Btrfs snapshots—see *Section 3.3, “System Rollback by Booting from Snapshots”* for more information.

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 snapshots on the fly and manage existing snapshots. See *Section 3.5, “Manually Creating and Managing Snapshots”* for details.

3.1 Default Setup

Snapper on openSUSE Leap is set up to serve 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 (approximately more than 8GB). Taking snapshots on partitions other than / is not enabled by default.

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 “normal” partitions. The exact amount strongly depends on the number of snapshots you keep and the amount of data modifications. As a rule of thumb you should consider using twice the size than you normally would.

3.1.1 Types of Snapshots

Although snapshots themselves do not differ in a technical sense, we distinguish between three types of snapshots, based on the occasion on which they were taken:

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 enabled by default, except for the root partition.

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.

3.1.2 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,,

A rollback of the boot loader configuration is not supported. The directories listed above are architecture-specific. The first two directories are present on x86_64 machines, the latter two on IBM POWER and on IBM z Systems, respectively.

/home

If /home does not reside on a separate partition, it is excluded to avoid data loss on rollbacks.

/opt, /var/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, /var/tmp, /var/crash

All directories containing temporary files 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/lib/named

Contains zone data for the DNS server. Excluded from snapshots to ensure a name server can operate after a rollback.

/var/lib/mailman, /var/spool

Directories containing mails or mail queues are excluded to avoid a loss of mails after a rollback.

/var/lib/pgsql

Contains PostgreSQL data.

/var/log

Log file location. Excluded from snapshots to allow log file analysis after the rollback of a broken system.

3.1.3 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.

3.1.3.1 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.

3.1.3.2 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" ❶ important="true" ❷>kernel-* ❸</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> ❹
10 </solvables>
11 </snapper-zypp-plugin-conf>
```

- ❶ The `match` attribute defines whether the pattern is a Unix shell-style wild card (`w`) or a Python regular expression (`re`).
- ❷ 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.
- ❸ 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-`.
- ❹ This line unconditionally matches all packages.

With this configuration snapshot, pairs are made whenever a package is installed (line 9). When 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>
```

3.1.3.3 Creating and Mounting New Subvolumes

Creating a new subvolume underneath the `/`-hierarchy and permanently mounting it is supported. However, you need to make sure not to create it inside a 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`.

```
mount /dev/sda2 -o subvol=@ /mnt
btrfs subvolume create /mnt/usr/important
umount /mnt
```

The corresponding entry in `/etc/fstab` needs to look like the following:

```
/dev/sda2 /usr/important btrfs subvol=@/usr/important 0 0
```

3.1.3.4 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, the following snapshots are kept:

- the first snapshot of the last ten days, months, and years
- the last ten installation snapshot pairs marked as important
- the last ten installation/administration snapshots

Refer to [Section 3.4.1, "Managing Existing Configurations"](#) for instructions on how to change these values.

3.1.3.5 Using Snapper on Thin-Provisioned LVM Volumes

Apart from snapshots on Btrfs file systems, Snapper also supports taking snapshots on thin-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 thin-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:

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

You can adjust this configuration according to your needs as described in [Section 3.4.1, “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 3.1.3.1, “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 vs. 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 broken 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`.

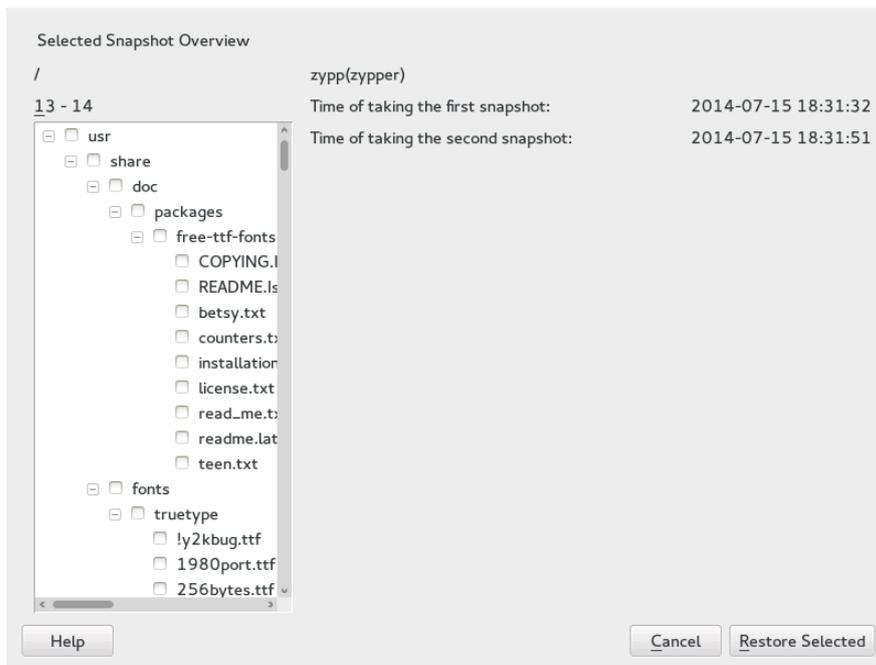
2. Make sure *Current Configuration* is set to *root*. This is always the case unless you have manually added own Snapper configurations.
3. Choose a pair of pre- and post-snapshots from the list. Both, YaST and Zypper snapshot pairs are of the type *Pre & Post*. YaST snapshots are labeled as zypp(y2base) in the *Description column*; Zypper snapshots are labeled zypp(zypper).

Snapshots

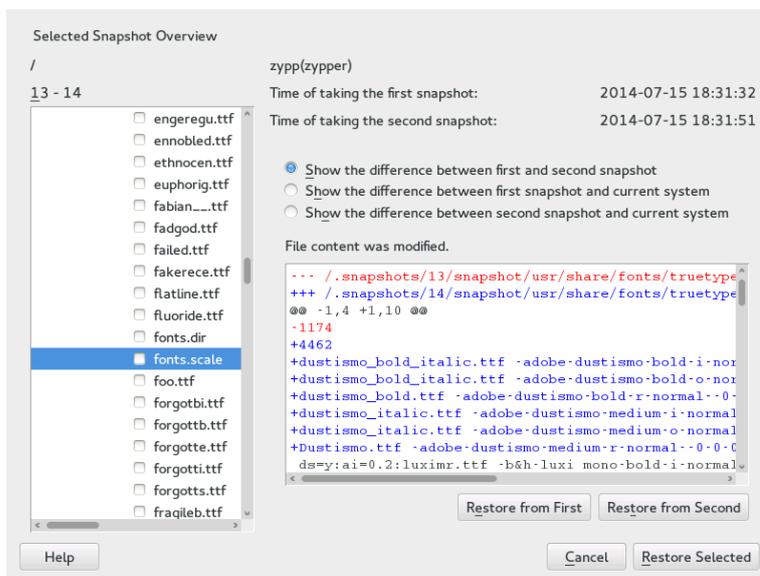
Current Configuration root

ID	Type	Start Date	End Date	Description	User Data
328	Single	Wed May 7 05:15:01 2014		timeline	
329	Single	Wed May 7 06:15:01 2014		timeline	
330	Single	Wed May 7 07:15:01 2014		timeline	
331	Single	Wed May 7 08:15:01 2014		timeline	
332	Single	Wed May 7 09:15:01 2014		timeline	
333	Single	Wed May 7 10:15:01 2014		timeline	
334	Single	Wed May 7 11:15:01 2014		timeline	
335	Single	Wed May 7 12:15:02 2014		timeline	
336	Single	Wed May 7 13:15:01 2014		timeline	
337	Single	Wed May 7 14:15:01 2014		timeline	
338	Single	Wed May 7 15:15:01 2014		timeline	
339	Single	Wed May 7 16:15:01 2014		timeline	
340 - 341	Pre & Post	Wed May 7 16:15:10 2014	Wed May 7 16:15:16 2014	zypp(zypper)	important=no
342 - 343	Pre & Post	Wed May 7 16:20:38 2014	Wed May 7 16:20:42 2014	zypp(y2base)	important=no
344 - 345	Pre & Post	Wed May 7 16:21:23 2014	Wed May 7 16:21:24 2014	zypp(zypper)	important=no
346 - 347	Pre & Post	Wed May 7 16:41:06 2014	Wed May 7 16:41:10 2014	zypp(y2base)	important=no
348 - 349	Pre & Post	Wed May 7 16:44:50 2014	Wed May 7 16:44:53 2014	zypp(y2base)	important=no
350 - 351	Pre & Post	Wed May 7 16:46:27 2014	Wed May 7 16:46:38 2014	zypp(y2base)	important=no

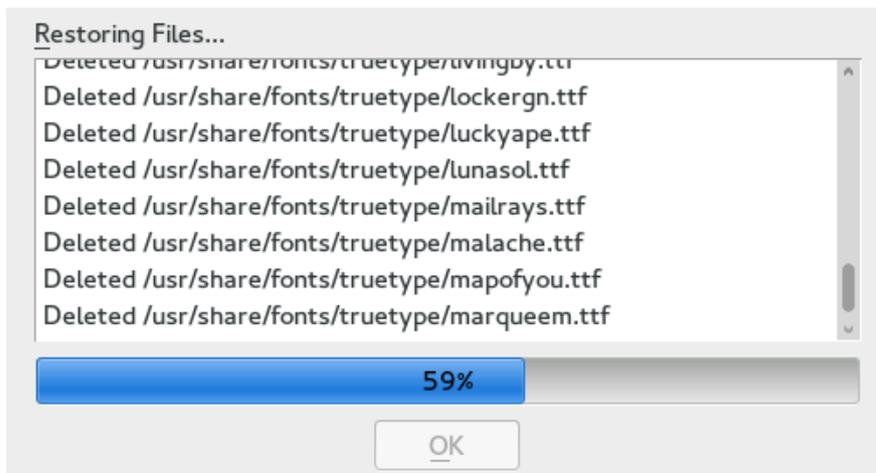
4. Click *Show Changes* to open the list of files that differ between the two snapshots.



- 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*.

PROCEDURE 3.2: UNDOING CHANGES USING THE `snapper` COMMAND

1. Get a list of YaST and Zypper snapshots by running `snapper list -t pre-post`. YaST snapshots are labeled as `yast module_name` in the *Description column*; Zypper snapshots are labeled `zypp(zypper)`.

```
root # snapper list -t pre-post
Pre # | Post # | Pre Date                | Post Date                | Description
-----+-----+-----+-----+-----
311  | 312   | Tue 06 May 2014 14:05:46 CEST | Tue 06 May 2014 14:05:52 CEST | zypp(y2base)
340  | 341   | Wed 07 May 2014 16:15:10 CEST | Wed 07 May 2014 16:15:16 CEST | zypp(zypper)
342  | 343   | Wed 07 May 2014 16:20:38 CEST | Wed 07 May 2014 16:20:42 CEST | zypp(y2base)
344  | 345   | Wed 07 May 2014 16:21:23 CEST | Wed 07 May 2014 16:21:24 CEST | zypp(zypper)
346  | 347   | Wed 07 May 2014 16:41:06 CEST | Wed 07 May 2014 16:41:10 CEST | zypp(y2base)
348  | 349   | Wed 07 May 2014 16:44:50 CEST | Wed 07 May 2014 16:44:53 CEST | zypp(y2base)
350  | 351   | Wed 07 May 2014 16:46:27 CEST | Wed 07 May 2014 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 `-`.

```
root # snapper status 350..351
+.... /usr/share/doc/packages/mikachan-fonts
+.... /usr/share/doc/packages/mikachan-fonts/COPYING
+.... /usr/share/doc/packages/mikachan-fonts/dl.html
c.... /usr/share/fonts/truetype/fonts.dir
c.... /usr/share/fonts/truetype/fonts.scale
+.... /usr/share/fonts/truetype/#####-p.ttf
+.... /usr/share/fonts/truetype/#####-pb.ttf
+.... /usr/share/fonts/truetype/#####-ps.ttf
```

```

+..... /usr/share/fonts/truetype/#####.ttf
c..... /var/cache/fontconfig/7ef2298fde41cc6eeb7af42e48b7d293-x86_64.cache-4
c..... /var/lib/rpm/Basenames
c..... /var/lib/rpm/Dirnames
c..... /var/lib/rpm/Group
c..... /var/lib/rpm/Installtid
c..... /var/lib/rpm/Name
c..... /var/lib/rpm/Packages
c..... /var/lib/rpm/Providename
c..... /var/lib/rpm/Requirename
c..... /var/lib/rpm/Sha1header
c..... /var/lib/rpm/Sigmd5

```

3. To display the diff for a certain file, run **snapper diff PRE..POST FILENAME**. If you do not specify FILENAME, a diff for all files will be displayed.

```

root # 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.

```

root # 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 making use of 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 3.4, "Creating and Modifying Snapper Configurations"](#) for instructions.



Warning: Restoring Files vs. 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:

```
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:

```
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

```
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

```
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 42.1 system rollbacks are only supported by SUSE if the default Snapper configuration (`root`) and the default configuration of the root partition have 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 vs. 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 3.1.2, “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 will not be able to 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, to choose a different snapshot, or to start the rescue system.
 - b. If you want to perform the rollback, run

```
sudo snapper rollback
```

and reboot afterwards. On the boot screen, choose the default boot entry to reboot into the reinstated system.



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 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.
- 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:

```
snapper modify --userdata "bootloader=STRING" NUMBER
```

3.3.2 Limitations

A *complete* system rollback, restoring the complete system to exactly the same state as it was in when a snapshot was taken, is not possible.

3.3.2.1 Directories Excluded from Snapshots

Root file system snapshots do not contain all directories. See *Section 3.1.2, "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.

3.3.2.2 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.

3.4 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/`. The default configuration installed with Snapper for the `/` directory is named `root`. It creates and manages the YaST and Zypper snapshots plus the hourly backup snapshot for `/`.

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`:

```
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 3.4.1, “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:

```
snapper -c www-data create-config -t my_defaults /srv/www
```

3.4.1 Managing Existing Configurations

The `snapper` offers several subcommands for managing existing configurations. You can list, show, delete and modify them:

List Configurations

Use the command `snapper list-configs` to get all existing configurations:

```
root # snapper list-configs
```

```

Config | Subvolume
-----+-----
root   | /
usr    | /usr
local  | /local

```

Delete a Configuration

Use the subcommand **snapper -c CONFIG delete-config** to delete a configuration. *Config* needs to be replaced by a configuration name shown by **snapper list-configs**.

Show a Configuration

Use the subcommand **snapper -c CONFIG get-config** to display the specified configuration. *Config* needs to be replaced by a configuration name shown by **snapper list-configs**. See [Section 3.4.1.1, "Configuration Data"](#) for more information on the configuration options.

Use the subcommand **snapper -c CONFIG set-config OPTION = VALUE** to modify an option in the specified configuration. *Config* needs to be replaced by a configuration name shown by **snapper list-configs**. Possible values for *OPTION* and *VALUE* are listed in [Section 3.4.1.1, "Configuration Data"](#).

3.4.1.1 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:

ALLOW_GROUPS, ALLOW_USERS

Granting permissions to use snapshots to regular users. See [Section 3.4.1.2, "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_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" .

FSTYPE

File system type of the partition. Do not change.

The default value is "btrfs" .

NUMBER_CLEANUP

Defines whether to automatically delete old installation and administration snapshot pairs when the total snapshot count exceeds a number specified with NUMBER_LIMIT *and* an age specified with NUMBER_MIN_AGE . Valid values: yes , no

The default value is "no" .



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 a certain number of snapshots regardless of their age, set NUMBER_MIN_AGE to 0 . On the other hand, if you do not want to keep snapshots beyond a certain age, set NUMBER_LIMIT and NUMBER_LIMIT_IMPORTANT to 0 .

NUMBER_LIMIT

Defines how many installation and administration snapshot pairs that are not marked as important to keep if NUMBER_CLEANUP is set to yes . Only the youngest snapshots will be kept.

The default value is "50" .

NUMBER_LIMIT_IMPORTANT

Defines how many snapshot pairs marked as important to keep if NUMBER_CLEANUP is set to yes . Only the youngest snapshots will be kept.

The default value is "10" .

NUMBER_MIN_AGE

Defines the minimum age in seconds a snapshot pair must have before it can automatically be deleted.

The default value is "1800" .

SUBVOLUME

Mount point of the partition or subvolume to snapshot. Do not change.

SYNC_ACL

If Snapper is to be used by regular users (see [Section 3.4.1.2, "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_CLEANUP

Defines whether to automatically delete old snapshots when the snapshot count exceeds a number specified with the `TIMELINE_LIMIT_*` options *and* an age specified with `TIMELINE_MIN_AGE`. Valid values: `yes`, `no`

The default value is `"no"`.

TIMELINE_CREATE

If set to `yes`, hourly snapshots are created. This is currently the only way to automatically create snapshots, therefore setting it to `yes` is strongly recommended. Valid values: `yes`, `no`

The default value is `"no"`.

TIMELINE_LIMIT_DAILY, TIMELINE_LIMIT_HOURLY, TIMELINE_LIMIT_MONTHLY, TIMELINE_LIMIT_YEARLY

Number of snapshots to keep for hour, day, month, year.

The default value for each entry is `"10"`.

EXAMPLE 3.1: EXAMPLE TIMELINE CONFIGURATION

```
TIMELINE_CLEANUP="yes"
TIMELINE_CREATE="yes"
TIMELINE_LIMIT_DAILY="10"
TIMELINE_LIMIT_HOURLY="10"
TIMELINE_LIMIT_MONTHLY="10"
TIMELINE_LIMIT_YEARLY="10"
TIMELINE_MIN_AGE="1800"
```

This example configuration enables hourly snapshots which are automatically cleaned up. `TIMELINE_MIN_AGE` and `TIMELINE_LIMIT_*` are always evaluated both. 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 ten snapshots that have been made.
- Daily: The first daily snapshot that has been made is kept for the last ten days.
- Monthly: The first snapshot made on the last day of the month is kept for the last ten months.
- Yearly: The first snapshot made on the last day of the year is kept for the last ten years.

`TIMELINE_MIN_AGE`

Defines the minimum age in seconds a snapshot must have before it can automatically be deleted.

The default value is `"1800"`.

3.4.1.2 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 Snapper configurations that grant permissions to users or/and groups can be created. 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 not existing, create a Snapper configuration for the partition or subvolume on which the user should be able to use Snapper. Refer to *Section 3.4, "Creating and Modifying Snapper Configurations"* for instructions. Example:

```
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; see [Section 3.4.1, “Managing Existing Configurations”](#) for details.
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:

```
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
```

3.5 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 3.4, “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. If you want 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`).

3.5.1 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. The following metadata is available for each snapshot:

- **Type:** Snapshot type, see [Section 3.5.1.1, “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 3.5.1.2, “Cleanup-algorithms”](#) for details.

3.5.1.1 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 has got a corresponding post snapshot. Used for the automatic YaST/Zypper snapshots, for example.

post

Snapshot of a file system *after* a modification. Each post snapshot has got a corresponding pre snapshot. Used for the automatic YaST/Zypper snapshots, for example.

single

Stand-alone snapshot. Used for the automatic hourly snapshots, for example. This is the default type when creating snapshots.

3.5.1.2 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 the different types of snapshots to keep in the Snapper configuration (see [Section 3.4.1, “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 a number of hourly, daily, monthly, and yearly snapshots.

empty-pre-post

Deletes pre/post snapshot pairs with empty diffs.

3.5.2 Creating Snapshots

Creating a snapshot is done by running `snapper create` or by clicking *Create* in the YaST module *Snapper*. The following examples explain how to create snapshots from the command line. It should be easy to adopt them when using the YaST interface.



Tip: Snapshot Description

You should always specify a meaningful description to later be able to identify its purpose. Even more information can be specified via the user data option.

```
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 file 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.

3.5.3 Modifying Snapshot Metadata

Snapper allows you to modify the description, the cleanup algorithm, and the userdata 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.

3.5.4 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`**.

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 --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 3.5.1.2, “Cleanup-algorithms”](#) for details.

3.6 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 3.1.2, “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: *How much disk space is used by snapshots? How to free disk space?*

A: Since the `df` does not show the correct disk usage on `Btrfs` file systems, you need to use the command `btrfs filesystem df MOUNT_POINT`. Displaying the amount of disk space a snapshot allocates is currently not supported by the `Btrfs` tools.

To free space on a `Btrfs` partition containing snapshots you need to delete unneeded snapshots rather than files. Older snapshots occupy more space than recent ones. See [Section 3.1.3.4, “Controlling Snapshot Archiving”](#) for details.

Doing an upgrade from one service pack to another results in snapshots occupying a lot of disk space on the system subvolumes, because a lot of data gets changed (package updates). Manually deleting these snapshots after they are no longer needed is recommended. See [Section 3.5.4, “Deleting Snapshots”](#) for details.

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: *Where can I get more information on Snapper?*

A: See the Snapper home page at <http://snapper.io/>.

4 Remote Access with VNC

Virtual Network Computing (VNC) enables you to control a remote computer via a graphical desktop (as opposed to a remote shell access). VNC is platform-independent and lets you access the remote machine from any operating system.

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.



Note: Session Types

A machine 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 One-time VNC Sessions

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. Once 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*.
3. 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*.
4. Confirm your settings with *Finish*.
5. In case not all needed packages are available yet, you need to approve the installation of missing packages.

4.1.1 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 4.1.3, “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. The Java client is served over HTTPS, using the same certificate as VNC.

4.1.2 Initiating a One-time VNC Session

To initiate a one-time VNC session, a VNC viewer must be installed on the client machine. The standard viewer on SUSE Linux products is vncviewer, provided by the package tigervnc. You may also view a VNC session using your Web browser and a Java applet.

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

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

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

```
vncviewer jupiter.example.com::5901
```

Alternatively use a Java-capable Web browser to view the VNC session by entering the following URL: <http://jupiter.example.com:5801>

4.1.3 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 `xinetd` daemon. A configuration file is located at `/etc/xinetd.d/vnc`. By default it offers six configuration blocks: three for VNC viewers (`vnc1` to `vnc3`), and three serving a Java applet (`vnchttpd1` to `vnchttpd3`). By default only `vnc1` and `vnchttpd1` are active.

To activate a configuration, comment the line `disable = yes` with a `#` character in the first column, or remove that line completely. To deactivate a configuration uncomment or add that line.

The `Xvnc` server can be configured via the `server_args` option—see `Xvnc --help` for a list of options.

When adding custom configurations, make sure they are not using ports that are already in use by other configurations, other services, or existing persistent VNC sessions on the same host.

Activate configuration changes by entering the following command:

```
sudo rcxinetd reload
```

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 Guide”, Chapter 15 “Masquerading and Firewalls”* for instructions.

4.2 Persistent VNC Sessions

A 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.

A persistent session can be accessed from multiple clients simultaneously. This is ideal for demonstration purposes or for trainings where the trainer might need access to the trainee's desktop. However, most of the times you probably do not want to share your VNC session.

In contrast to one-time sessions that start a display manager, a persistent session starts a ready-to-operate desktop that runs as the user that started the VNC session. Access to persistent sessions is protected by a password.

PROCEDURE 4.2: STARTING A PERSISTENT VNC SESSION

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 Guide", Chapter 15 "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 1024x769 pixel and with a color depth of 16-bit, enter the following command:

```
vncserver -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 `vncviewer` for the first time, it asks for a password for full access to the session. The password you are providing here is also used for future sessions started by the same user. It 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.

VNC connections are unencrypted, so people who can sniff the network(s) between the two machines can read the password when it gets transferred at the beginning of a session.

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`

4.2.1 Connecting to a Persistent VNC Session

To connect to a persistent VNC session, a VNC viewer must be installed. The standard viewer on SUSE Linux products is `vncviewer`, provided by the package `tigervnc` (default). You may also view a VNC session using your Web browser and a Java applet.

To start your VNC viewer and connect to display `:1` of the VNC server, use the command

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

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

```
vncviewer jupiter.example.com::5901
```

Alternatively use a Java-capable Web browser to view the VNC session by entering the following URL: `http://jupiter.example.com:5801`

4.2.2 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.

5 Advanced Disk Setup

Sophisticated system configurations require specific disk setups. All common partitioning tasks can be done with YaST. 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, and there is also the option to use iSCSI as a networked disk.

5.1 Using the YaST 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 do a complete backup of your data before attempting to do so.

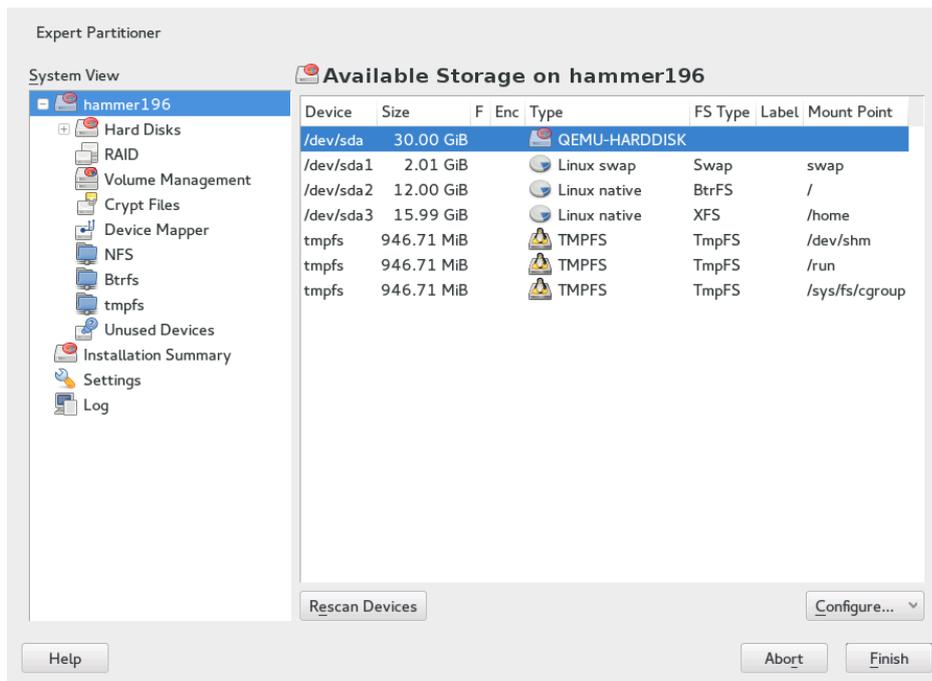


FIGURE 5.1: THE YAST PARTITIONER

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*. Use these views to gather information about existing storage configurations, or to configure functions like RAID, Volume Management, Crypt Files, or 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 starting from the bottom toward the top of the list (starting from the last partition of a hard disk toward the first).

5.1.1 Partition Types

Every hard disk has a partition table with space for four entries. Every entry in the partition table corresponds to a primary partition or an extended partition. Only one extended partition entry is allowed, however.

A primary partition simply consists of a continuous range of cylinders (physical disk areas) assigned to a particular operating system. With primary partitions you would be limited to four partitions per hard disk, because more do not fit in the partition table. This is why extended partitions are used. Extended partitions are also continuous ranges of disk cylinders, but an extended partition may be divided into *logical partitions* itself. Logical partitions do not require entries in the partition table. In other words, an extended partition is a container for logical partitions.

If you need more than four partitions, create an extended partition as the fourth partition (or earlier). This extended partition should occupy the entire remaining free cylinder range. Then create multiple logical partitions within the extended partition. The maximum number of logical partitions is 63, independent of the disk type. It does not matter which types of partitions are used for Linux. Primary and logical partitions both function normally.



Tip: GPT Partition Table

If you need to create more than 4 primary partitions on one hard disk, you need to use the GPT partition type. This type removes the primary partitions number restriction, and supports partitions bigger than 2 TB as well.

To use GPT, run the YaST Partitioner, click the relevant disk name in the *System View* and choose *Expert > Create New Partition Table > GPT*.

5.1.2 Creating a Partition

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

1. Select *Add* and specify the partition type (primary or extended). Create up to four primary partitions or up to three primary partitions and one extended partition. Within the extended partition, create several logical partitions (see [Section 5.1.1, "Partition Types"](#)).

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*. For more information on supported file systems, see [root](#).
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.

5.1.2.1 Btrfs Partitioning

The default file system for the root partition is Btrfs (see [Chapter 3, System Recovery and Snapshot Management with Snapper](#) for more information on Btrfs). 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. If you need to have the root partition encrypted in this setup, 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](#) snapshots the *root* file system by default, it is reasonable to exclude specific directories from being snapshot, depending on the nature of data they hold. And that is why YaST suggests creating the following separate subvolumes.

SUGGESTED BTRFS SUBVOLUMES

/tmp /var/tmp /var/run

Directories with frequently changed content.

/var/spool

Contains user data, such as mails.

/var/lib

Holds dynamic data libraries and files plus state information pertaining to an application or the system.

By default, subvolumes with the option no copy on write are created for: /var/lib/mariadb, /var/lib/pgsql, and /var/lib/libvirt/images.

/var/log

Contains system and applications' log files which should never be rolled back.

/var/crash

Contains memory dumps of crashed kernels.

/srv

Contains data files belonging to FTP and HTTP servers.

/opt

Contains third party software.



Tip: Size of Btrfs Partition

Because saved snapshots require more disk space, it is recommended to reserve more space for Btrfs partition than for a partition not capable of snapshotting (such as Ext3). Recommended size for a root Btrfs partition with suggested subvolumes is 20GB.

5.1.2.1.1 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 off all existing subvolumes of the selected Btrfs partition. You can notice 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*.

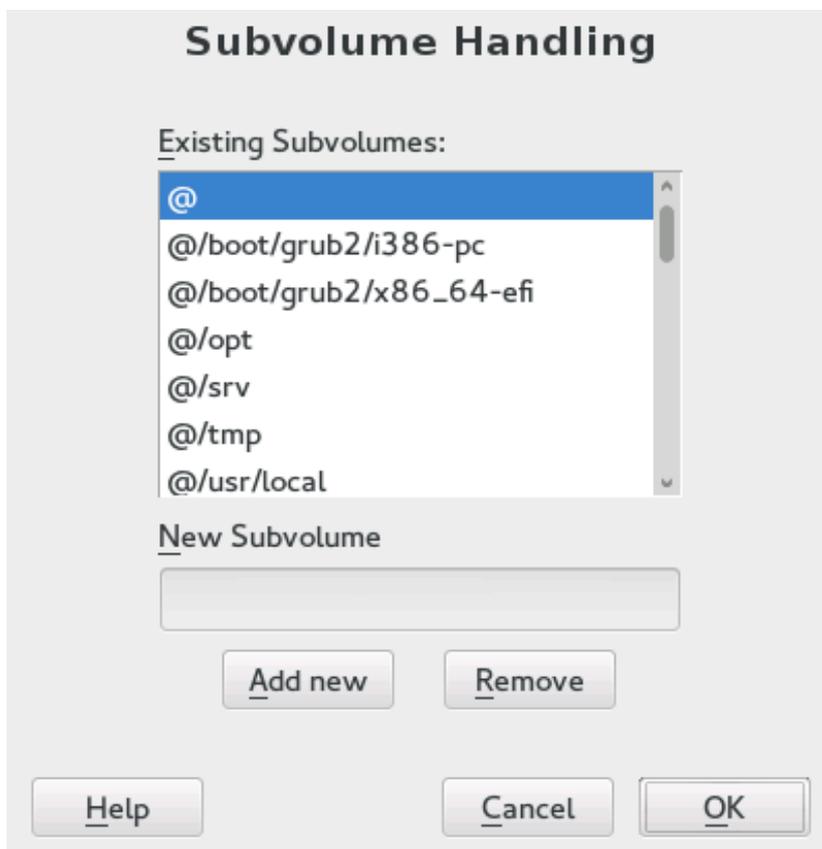


FIGURE 5.2: BTRFS SUBVOLUMES IN YAST PARTITIONER

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 JFS 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.

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 more memory to your system instead of adding more 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 Guide", Chapter 11 "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 3 "Managing Users with YaST", Section 3.3.4 "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.



Tip: 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.

5.1.6.1 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.

PROCEDURE 5.2: ADDING A SWAP FILE MANUALLY

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

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

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

2. Initialize this swap file with the command

```
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 (will update `/etc/fstab`) or adjust `/etc/fstab` manually.

3. Activate the swap with the command

```
swapon /var/lib/swap/swapfile
```

To disable this swap file, use the command

```
swapoff /var/lib/swap/swapfile
```

4. Check the current available swap spaces with the command

```
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.

In case you want to change your `/usr` or `swap`, refer to *Procedure 9.1, "Updating Init RAM Disk When Switching to Logical Volumes"*.

5.2 LVM Configuration

This section briefly describes the principles behind the Logical Volume Manager (LVM) and its multipurpose features. In [Section 5.2.2, “LVM Configuration with YaST”](#), learn how to set up LVM with YaST.



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.

5.2.1 The Logical Volume Manager

The LVM enables flexible distribution of hard disk space over several file systems. It was developed because sometimes the need to change the segmenting of hard disk space arises just after the initial partitioning has been done. Because it is difficult to modify partitions on a running system, LVM provides a virtual pool (volume group, VG for short) of memory space from which logical volumes (LVs) can be created as needed. The operating system accesses these LVs instead of the physical partitions. Volume groups can occupy more than one disk, so that several disks or parts of them may constitute one single VG. This way, LVM provides a kind of abstraction from the physical disk space that allows its segmentation to be changed in a much easier and safer way than with physical repartitioning. Background information regarding physical partitioning can be found in [Section 5.1.1, “Partition Types”](#) and [Section 5.1, “Using the YaST Partitioner”](#).

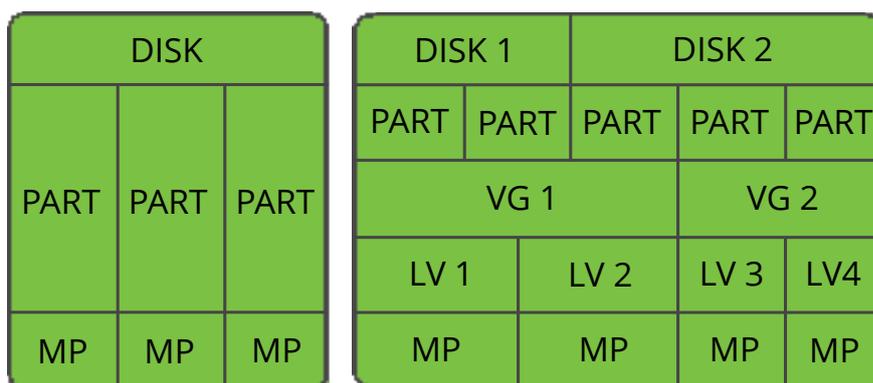


FIGURE 5.3: PHYSICAL PARTITIONING VERSUS LVM

Figure 5.3, "Physical Partitioning versus LVM" compares physical partitioning (left) with LVM segmentation (right). On the left side, one single disk has been divided into three physical partitions (PART), each with a mount point (MP) assigned so that the operating system can gain access. On the right side, two disks have been divided into two and three physical partitions each. Two LVM volume groups (VG 1 and VG 2) have been defined. VG 1 contains two partitions from DISK 1 and one from DISK 2. VG 2 contains the remaining two partitions from DISK 2. In LVM, the physical disk partitions that are incorporated in a volume group are called physical volumes (PVs). Within the volume groups, four LVs (LV 1 through LV 4) have been defined. They can be used by the operating system via the associated mount points. The border between different LVs do not need to be aligned with any partition border. See the border between LV 1 and LV 2 in this example.

LVM features:

- Several hard disks or partitions can be combined in a large logical volume.
- Provided the configuration is suitable, an LV (such as `/usr`) can be enlarged if free space is exhausted.
- With LVM, it is possible to add hard disks or LVs in a running system. However, this requires hotpluggable hardware.
- It is possible to activate a "striping mode" that distributes the data stream of an LV over several PVs. If these PVs reside on different disks, the read and write performance is enhanced, as with RAID 0.
- The snapshot feature enables consistent backups (especially for servers) of the running system.

With these features, LVM is ready for heavily used home PCs or small servers. LVM is well-suited for the user with a growing data stock (as in the case of databases, music archives, or user directories). This would allow file systems that are larger than the physical hard disk. Another advantage of LVM is that up to 256 LVs can be added. However, working with LVM is different from working with conventional partitions. Instructions and further information about configuring LVM is available in the official LVM HOWTO at <http://tldp.org/HOWTO/LVM-HOWTO/>.

Starting from Kernel version 2.6, LVM version 2 is available, which is backward-compatible with the previous LVM and enables the continued management of old volume groups. When creating new volume groups, decide whether to use the new format or the backward-compatible

version. LVM 2 does not require any kernel patches. It uses the device mapper integrated in kernel 2.6. This kernel only supports LVM version 2. Therefore, when talking about LVM, this section always refers to LVM version 2.

5.2.1.1 Thin Provisioning

Starting from Kernel version 3.4, LVM supports thin provisioning. A thin-provisioned volume has a virtual capacity and a real capacity. *Virtual* capacity is the volume storage capacity that is available to a host. *Real* capacity is the storage capacity that is allocated to a volume copy from a storage pool. In a fully allocated volume, the virtual capacity and real capacity are the same. In a thin-provisioned volume, however, the virtual capacity can be much larger than the real capacity. If a thin-provisioned volume does not have enough real capacity for a write operation, the volume is taken offline and an error is logged.

For more general information, see http://wikibon.org/wiki/v/Thin_provisioning.

5.2.2 LVM Configuration with YaST

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

5.2.2.1 Creating Volume Groups

If no volume group exists on your system, you must add one (see [Figure 5.4, “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 Extent 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.

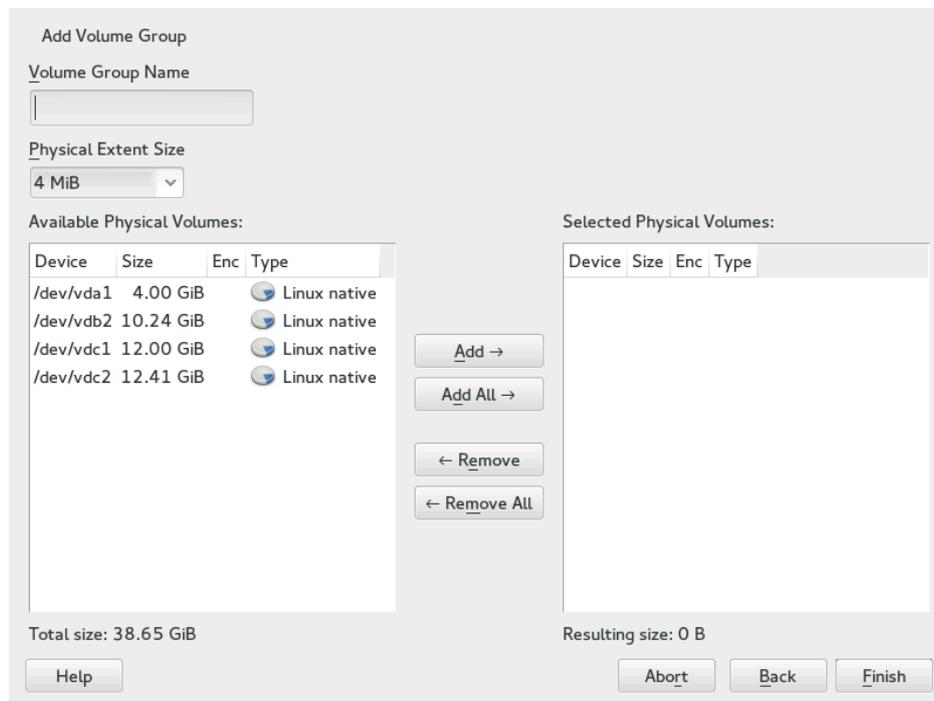


FIGURE 5.4: CREATING A VOLUME GROUP

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.

5.2.2.2 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.

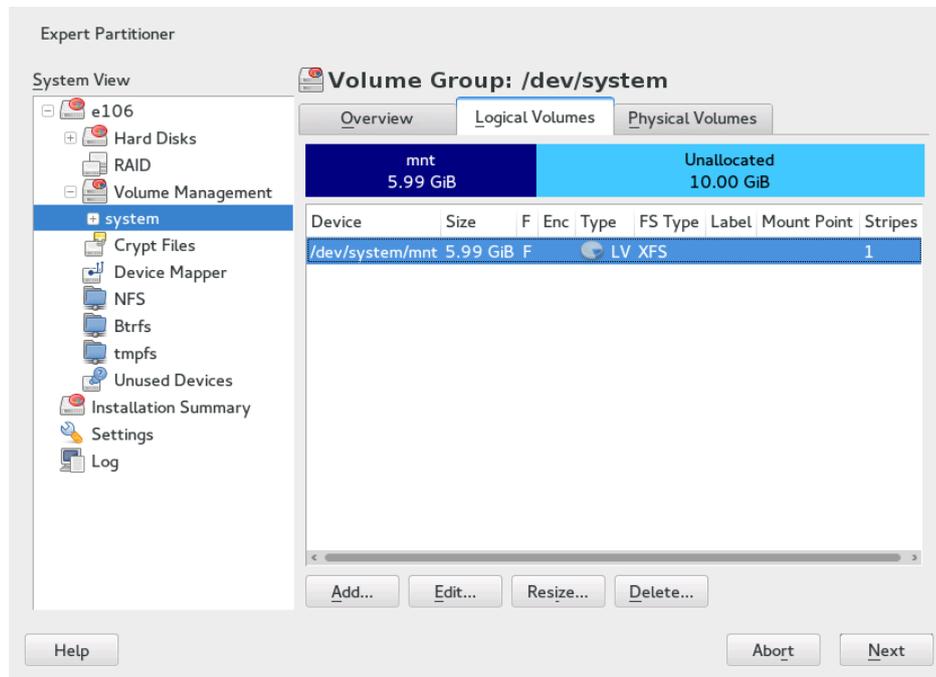


FIGURE 5.5: LOGICAL VOLUME MANAGEMENT

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 Configuration

The purpose of RAID (redundant array of independent disks) is to combine several hard disk partitions into one large *virtual* hard disk to optimize performance and/or data security. Most RAID controllers use the SCSI protocol because it can address a larger number of hard disks in a more effective way than the IDE protocol. It is also more suitable for the parallel command processing. There are some RAID controllers that support IDE or SATA hard disks. Soft RAID provides the advantages of RAID systems without the additional cost of hardware RAID controllers. However, this requires some CPU time and has memory requirements that make it unsuitable for high performance computers.

With openSUSE® Leap , you can combine several hard disks into one soft RAID system. RAID implies several strategies for combining several hard disks in a RAID system, each with different goals, advantages, and characteristics. These variations are commonly known as *RAID levels*.

Common RAID levels are:

RAID 0

This level improves the performance of your data access by spreading out blocks of each file across multiple disk drives. Actually, this is not really a RAID, because it does not provide data backup, but the name *RAID 0* for this type of system is commonly used. With RAID 0, two or more hard disks are pooled together. Performance is enhanced, but the RAID system is destroyed and your data lost if even one hard disk fails.

RAID 1

This level provides adequate security for your data, because the data is copied to another hard disk 1:1. This is known as *hard disk mirroring*. If one disk is destroyed, a copy of its contents is available on the other one. All disks but one could be damaged without endangering your data. However, if the damage is not detected, the damaged data can be mirrored to the undamaged disk. This could result in the same loss of data. The writing performance suffers in the copying process compared to using single disk access (10 to 20 % slower), but read access is significantly faster in comparison to any one of the normal physical hard disks. The reason is that the duplicate data can be parallel-scanned. Generally it can be said that Level 1 provides nearly twice the read transfer rate of single disks and almost the same write transfer rate as single disks.

RAID 5

RAID 5 is an optimized compromise between Level 0 and Level 1, in terms of performance and redundancy. The hard disk space equals the number of disks used minus one. The data is distributed over the hard disks as with RAID 0. *Parity blocks*, created on one of the partitions, exist for security reasons. They are linked to each other with XOR, enabling the contents to be reconstructed by the corresponding parity block in case of system failure. With RAID 5, no more than one hard disk can fail at the same time. If one hard disk fails, it must be replaced as soon as possible to avoid the risk of losing data.

RAID 6

To further increase the reliability of the RAID system, it is possible to use RAID 6. In this level, even if two disks fail, the array still can be reconstructed. With RAID 6, at least 4 hard disks are needed to run the array. Note that when running as software raid, this configuration needs a considerable amount of CPU time and memory.

RAID 10 (RAID 1+0)

This RAID implementation combines features of RAID 0 and RAID 1: the data is first mirrored to separate disk arrays, which are inserted into a new RAID 0; type array. In each RAID 1 sub-array, one disk can fail without any damage to the data. A minimum of four disks and an even number of disks is needed to run a RAID 10. This type of RAID is used for database application where a huge load is expected.

Other RAID Levels

Several other RAID levels have been developed (RAID 2, RAID 3, RAID 4, RAIDn, RAID 10, RAID 0 + 1, RAID 30, RAID 50, etc.), some being proprietary implementations created by hardware vendors. These levels are not very common and therefore are not explained here.

5.3.1 Soft RAID Configuration with YaST

The YaST *RAID* configuration can be reached from the YaST Expert Partitioner, described in [Section 5.1, “Using the YaST 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 *0xFD 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.



Tip: Classify Disks

For RAID types where the order of added disks matters, you can mark individual disks with one of the letters A to E. Click the *Classify* button, select the disk and click of the *Class X* buttons, where X is the letter you want to assign to the disk. Assign all available RAID disks this way, and confirm with *OK*. You can easily sort the classified disks with the *Sorted* or *Interleaved* buttons, or add a sort pattern from a text file with *Pattern File*.

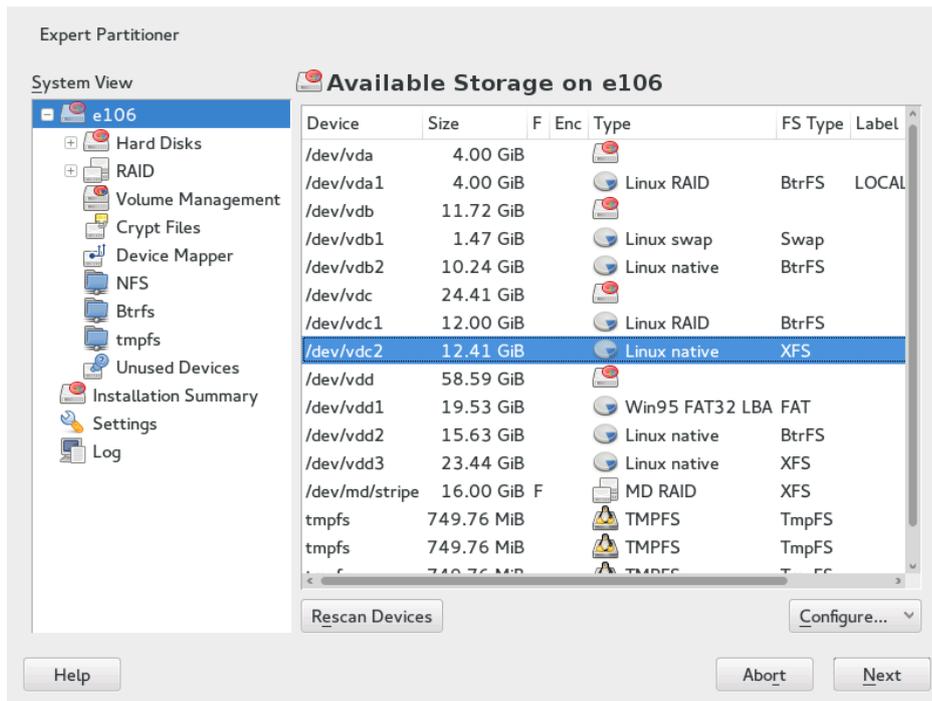


FIGURE 5.6: RAID PARTITIONS

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 in the HOWTOs at:

- </usr/share/doc/packages/mdadm/Software-RAID.HOWTO.html>
- <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 option.

Using this functionality, you can safely test kernel updates while being able to always fall back to the proven former kernel. To do so, do not use the update tools (such as the YaST Online Update or the `updater` applet), but instead follow the process described in this chapter.



Warning: Support Entitlement

Be aware that you lose your entire support entitlement for the machine when installing a self-compiled or a third-party kernel. Only kernels shipped with openSUSE Leap and kernels delivered via the official update channels for openSUSE Leap are supported.



Tip: Check Your Boot Loader Configuration Kernel

It is recommended to check your boot loader configuration after having installed another kernel to set the default boot entry of your choice. See [Section 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 on SUSE Linux Enterprise 12. 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 may 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:

`3.12.24-7.1`: 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,3.12.25.rc7-test
```

Keep the latest kernel, the one currently running, and 3.12.25.rc7-test.



Tip: Keep the running Kernel

Unless using special setups, you probably always want to keep the running Kernel. If not keeping the running Kernel, it will be deleted in case of a Kernel update. This in turn makes it necessary to immediately reboot the system after the update, since modules for the Kernel that is currently running can no longer be loaded since they have been deleted.

6.2 Installing/Removing Multiple Kernel Versions 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*.

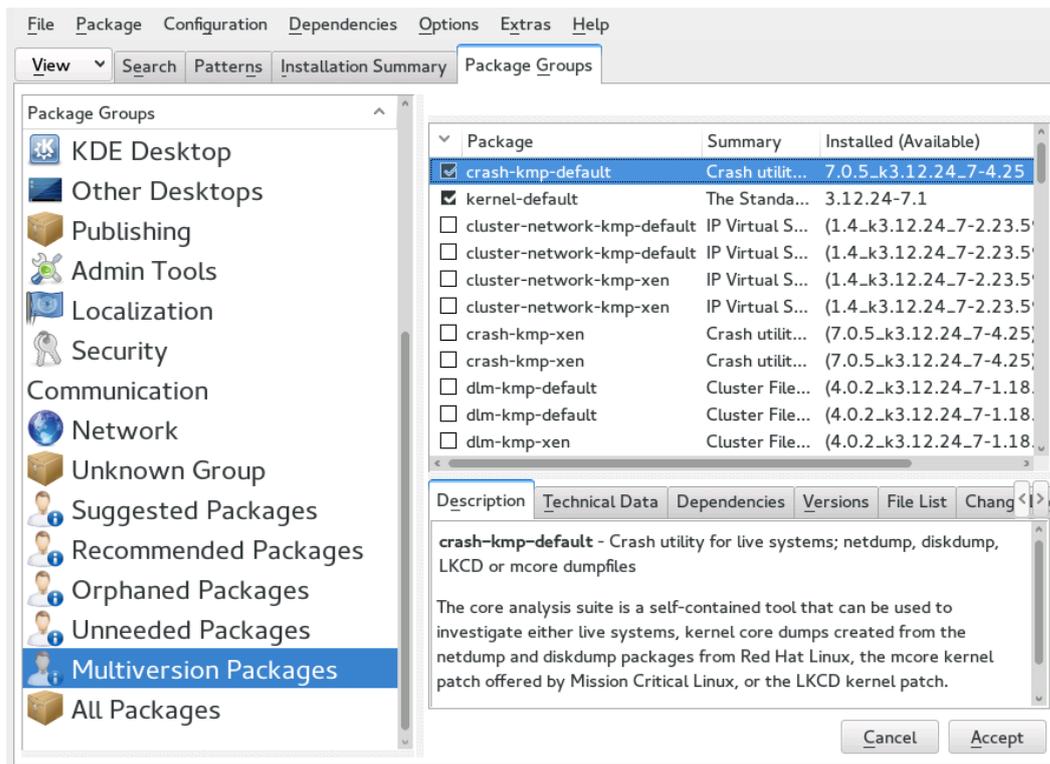


FIGURE 6.1: THE YAST SOFTWARE MANAGER: MULTIVERSION VIEW

3. Select a package and open its *Version* tab in the bottom pane on the left.
4. To install a package, click its check box. A green check mark indicates it is selected for installation.
To remove an already installed package (marked with a white check mark), click its check box 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

1. Use the command `zypper se -s 'kernel*'` to display a list of all kernel packages available:

S	Name	Type	Version	Arch	Repository
v	kernel-default	package	2.6.32.10-0.4.1	x86_64	Alternative Kernel
i	kernel-default	package	2.6.32.9-0.5.1	x86_64	(System Packages)

```
| kernel-default | srcpackage | 2.6.32.10-0.4.1 | noarch | Alternative Kernel
i | kernel-default | package      | 2.6.32.9-0.5.1 | x86_64 | (System Packages)
...
```

2. Specify the exact version when installing:

```
zypper in kernel-default-2.6.32.10-0.4.1
```

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.

7 GNOME Configuration for Administrators

This chapter introduces GNOME configuration options which administrators can use to adjust system-wide settings, such as customizing menus, installing themes, configuring fonts, changing preferred applications, and locking down capabilities.

These configuration options are stored in the GConf system. Access the GConf system with tools such as the [`gconftool-2`](#) command line interface or the [`gconf-editor`](#) GUI tool.

7.1 Starting Applications Automatically

To automatically start applications in GNOME, use one of the following methods:

- To run applications for each user: Put [`.desktop`](#) files in [`/usr/share/gnome/autostart`](#).
- To run applications for an individual user: Put [`.desktop`](#) files in [`~/.config/autostart`](#).

To disable an application that starts automatically, add [`X-Autostart-enabled=false`](#) to the [`.desktop`](#) file.

7.2 Automounting and Managing Media Devices

GNOME Files ([`nautilus`](#)) monitors volume-related events and responds with a user-specified policy. You can use GNOME Files to automatically mount hotplugged drives and inserted removable media, automatically run programs, and play audio CDs or video DVDs. GNOME Files can also automatically import photos from a digital camera.

System administrators can set system-wide defaults. For more information, see [Section 7.3, “Changing Preferred Applications”](#).

7.3 Changing Preferred Applications

To change users' preferred applications, edit [`/etc/gnome_defaults.conf`](#). Find further hints within this file.

For more information about MIME types, see <http://www.freedesktop.org/Standards/shared-mime-info-spec>.

7.4 Adding Document Templates

To add document templates for users, fill in the `Templates` directory in a user's home directory. You can do this manually for each user by copying the files into `~/Templates`, or system-wide by adding a `Templates` directory with documents to `/etc/skel` before the user is created.

A user creates a new document from a template by right-clicking the desktop and selecting *Create Document*.

7.5 For More Information

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

II System

- 8 32-Bit and 64-Bit Applications in a 64-Bit System Environment 95
- 9 Booting a Linux System 100
- 10 The systemd Daemon 105
- 11 **journalctl**: Query the systemd Journal 128
- 12 The Boot Loader GRUB 2 136
- 13 Basic Networking 154
- 14 UEFI (Unified Extensible Firmware Interface) 214
- 15 Special System Features 224
- 16 Dynamic Kernel Device Management with udev 236

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

openSUSE® Leap is available for 64-bit platforms. This does not necessarily mean that all the applications included have already been ported to 64-bit platforms. openSUSE Leap supports the use of 32-bit applications in a 64-bit system environment. This chapter offers a brief overview of how this support is implemented on 64-bit openSUSE Leap platforms. It explains how 32-bit applications are executed (runtime support) and how 32-bit applications should be compiled to enable them to run both in 32-bit and 64-bit system environments. Additionally, find information about the kernel API and an explanation of how 32-bit applications can run under a 64-bit kernel.

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.

8.1 Runtime Support



Important: Conflicts Between Application Versions

If an application is available both for 32-bit and 64-bit environments, parallel installation of both versions is bound to lead to problems. In such cases, decide on one of the two versions and install and use this.

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. On a 64-bit operating system that also runs 32-bit applications it is necessary to always install both versions of a PAM module.

To be executed correctly, every application requires a range of libraries. Unfortunately, the names for the 32-bit and 64-bit versions of these libraries are identical. They must be differentiated from each other in another way.

To retain compatibility with the 32-bit version, the libraries are stored at the same place in the system as in the 32-bit environment. The 32-bit version of `libc.so.6` is located under `/lib/libc.so.6` in both the 32-bit and 64-bit environments.

All 64-bit libraries and object files are located in directories called `lib64`. The 64-bit object files that you would normally expect to find under `/lib` and `/usr/lib` are now found under `/lib64` and `/usr/lib64`. This means that there is space for the 32-bit libraries under `/lib` and `/usr/lib`, so the file name for both versions can remain unchanged.

Subdirectories of 32-bit `/lib` directories which contain data content that does not depend on the word size are not moved. This scheme conforms to LSB (Linux Standards Base) and FHS (File System Hierarchy Standard).

8.2 Software Development

All 64-bit architectures support the development of 64-bit objects. The level of support for 32-bit compiling depends on the architecture. These are the various implementation options for the tool chain from GCC (GNU Compiler Collection) and binutils, which include the assembler `as` and the linker `ld`:

Biarch Compiler

Both 32-bit and 64-bit objects can be generated with a biarch development tool chain. A biarch development tool chain allows generation of 32-bit and 64-bit objects. The compilation of 64-bit objects is the default on almost all platforms. 32-bit objects can be generated if special flags are used. This special flag is `-m32` for GCC. The flags for the binutils are architecture-dependent, but GCC transfers the correct flags to linkers and assemblers. A biarch development tool chain currently exists for amd64 (supports development for x86 and amd64 instructions), for z Systems and for ppc64. 32-bit objects are normally created on the ppc64 platform. The `-m64` flag must be used to generate 64-bit objects.

No Support

openSUSE Leap does not support the direct development of 32-bit software on all platforms. To develop applications for x86 under ia64, use the corresponding 32-bit version of openSUSE Leap.

All header files must be written in an architecture-independent form. The installed 32-bit and 64-bit libraries must have an API (application programming interface) that matches the installed header files. The normal openSUSE Leap environment is designed according to this principle. In the case of manually updated libraries, resolve these issues yourself.

8.3 Software Compilation on Biarch Platforms

To develop binaries for the other architecture on a biarch architecture, the respective libraries for the second architecture must additionally be installed. These packages are called `rpm-name-32bit`. You also need the respective headers and libraries from the `rpmname-devel` packages and the development libraries for the second architecture from `rpmname-devel-32bit`.

For example, to compile a program that uses **libaio** on a system whose second architecture is a 32-bit architecture (x86_64), you need the following RPMs:

libaio-32bit

32-bit runtime package

libaio-devel-32bit

Headers and libraries for 32-bit development

libaio

64-bit runtime package

libaio-devel

64-bit development headers and libraries

Most open source programs use an **autoconf**-based program configuration. To use **autoconf** for configuring a program for the second architecture, overwrite the normal compiler and linker settings of **autoconf** by running the **configure** script with additional environment variables. The following example refers to an x86_64 system with x86 as the second architecture.

1. Use the 32-bit compiler:

```
CC="gcc -m32"
```

2. Instruct the linker to process 32-bit objects (always use **gcc** as the linker front-end):

```
LD="gcc -m32"
```

3. Set the assembler to generate 32-bit objects:

```
AS="gcc -c -m32"
```

4. Specify linker flags, such as the location of 32-bit libraries, for example:

```
LDFLAGS="-L/usr/lib"
```

5. Specify the location for the 32-bit object code libraries:

```
--libdir=/usr/lib
```

6. Specify the location for the 32-bit X libraries:

```
--x-libraries=/usr/lib
```

Not all of these variables are needed for every program. Adapt them to the respective program. An example **configure** call to compile a native 32-bit application on x86_64 could appear as follows:

```
CC="gcc -m32"  
LDFLAGS="-L/usr/lib;"  
./configure --prefix=/usr --libdir=/usr/lib --x-libraries=/usr/lib  
make  
make install
```

8.4 Kernel Specifications

The 64-bit kernels for x86_64 offer both a 64-bit and a 32-bit kernel ABI (application binary interface). The latter is identical with the ABI for the corresponding 32-bit kernel. This means that the 32-bit application can communicate with the 64-bit kernel in the same way as with the 32-bit kernel.

The 32-bit emulation of system calls for a 64-bit kernel 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 that have been specially compiled for this kernel. It is not possible to use 32-bit kernel modules.



Tip: Kernel-loadable Modules

Some applications require separate kernel-loadable modules. If you intend to use such a 32-bit application in a 64-bit system environment, contact the provider of this application and SUSE to 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 Booting a Linux System

Booting a Linux system involves different components and tasks. The hardware itself is initialized by the BIOS or the UEFI, which starts the Kernel by means of a boot loader. 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 setups for everyday usage, maintenance or emergencies.

9.1 The Linux Boot Process

The Linux boot process consists of several stages, each represented by a different component. The following list briefly summarizes the boot process and features all the major components involved:

1. **BIOS/UEFI.** 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 first hard disk and its geometry are recognized, the system control passes from the BIOS to the boot loader. If the BIOS supports network booting, it is also possible to configure a boot server that provides the boot loader. On x86_64 systems, PXE boot is needed. Other architectures commonly use the BOOTP protocol to get the boot loader.
2. **Boot Loader.** The first physical 512-byte data sector of the first hard disk is loaded into the main memory and the *boot loader* that resides at the beginning of this sector takes over. The commands executed by the boot loader determine the remaining part of the boot process. Therefore, the first 512 bytes on the first hard disk are called the *Master Boot Record* (MBR). The boot loader then passes control to the actual operating system, in this case, the Linux Kernel. More information about GRUB 2, the Linux boot loader, can be found in [Chapter 12, The Boot Loader GRUB 2](#). For a network boot, the BIOS 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.
3. **Kernel and `initramfs`.** To pass system control, the boot loader loads both the Kernel and an initial RAM-based file system (`initramfs`) into memory. The contents of the `initramfs` can be used by the Kernel directly. `initramfs` contains a small executable

called `init` that handles the mounting of the real root file system. If special hardware drivers are needed before the mass storage can be accessed, they must be in `initramfs`. For more information about `initramfs`, refer to [Section 9.2, “initramfs”](#). If the system does not have a local hard disk, the `initramfs` must provide the root file system for the Kernel. This can be done with the help of a network block device like iSCSI or SAN, but it is also possible to use NFS as the root device.



Note: The `init` Process Naming

Two different programs are commonly named “init”:

- a. the `initramfs` process mounting the root file system
- b. the operating system process setting up the system

In this chapter we will therefore refer to them as “`init on initramfs`” and “`systemd`”, respectively.

4. `init on initramfs`. 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`. After the 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. For more information about `init on initramfs`, refer to [Section 9.3, “Init on initramfs”](#). Find more information about `udev` in [Chapter 16, Dynamic Kernel Device Management with udev](#).
5. `systemd`. By starting services and mounting file systems, `systemd` handles the actual booting of the system. `systemd` is described in [Chapter 10, The systemd Daemon](#).

9.2 `initramfs`

`initramfs` is a small cpio archive that the Kernel can load into a RAM disk. It 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 may be 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 `udev.service` with the command `udevtrigger`.

If you need to change hardware (for example hard disks) in an installed system and this hardware requires different drivers to be in the Kernel at boot time, you must update the `initramfs` file. This is done by calling `dracut -f` (the option `-f` overwrites the existing `initramfs` file). To add a driver for the new hardware, edit `/etc/dracut.conf.d/01-dist.conf` and add the following line.

```
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 (`driver1 driver2`).

Important: Updating `initramfs` or `init`

The boot loader loads `initramfs` or `init` in the same way as the Kernel. It is not necessary to re-install GRUB 2 after updating `initramfs` or `init`, because GRUB 2 searches the directory for the right file when booting.

Tip: Changing Kernel Variables

If you change the values of some 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 calling `dracut -f` (the option `-f` overwrites the existing `initramfs` file).

9.3 Init on `initramfs`

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

For each loaded module, the Kernel generates device events. `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.

In case you want to change your `/usr` or `swap` partitions directly without the help of YaST, further actions are needed. If you forget these steps, your system will start in emergency mode. To avoid starting in emergency mode, perform the following steps:

PROCEDURE 9.1: UPDATING INIT RAM DISK WHEN SWITCHING TO LOGICAL VOLUMES

1. Edit the corresponding entry in `/etc/fstab` and replace your previous partitions with the logical volume.
2. Execute the following commands:

```
root # mount -a
root # swapon -a
```

3. Regenerate your initial ram disk (`initramfs`) with `mkinitrd` or `dracut`.
4. For z Systems, additionally run `grub2-install`.

Find more information about RAID and LVM in [Chapter 5, Advanced Disk Setup](#).

Managing Network Configuration

If you configured your system to use a network-mounted root file system (mounted via NFS), `init` on `initramfs` 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`.

When `init` on `initramfs` is called during the initial boot as part of the installation process, its tasks differ from those mentioned above:

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, “initramfs”](#), the boot process starts with a minimum set of drivers that can be used with most hardware configurations. `init` starts an initial hardware scanning process that determines the set of drivers suitable for your hardware configuration. 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

As soon as the hardware is properly recognized, the appropriate drivers are loaded. The `udev` program creates the special device files and `init` starts the installation system with the YaST installer.

Starting YaST

Finally, `init` starts YaST, which starts package installation and system configuration.

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.

Starting with openSUSE Leap 12 `systemd` is a replacement for the popular 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 encodes 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 different 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

```
rc<my_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:

```
systemctl start <my_1st_service> <my_2nd_service>
```

If you want to list all services available on the system:

```
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	<code>systemd</code> Command	System V <code>init</code> Command
Starting.	<code>start</code>	<code>start</code>
Stopping.	<code>stop</code>	<code>stop</code>
Restarting. Shuts down services and starts them afterwards. If a service is not yet running it will be started.	<code>restart</code>	<code>restart</code>
Restarting conditionally. Restarts services if they are currently running. Does nothing for services that are not running.	<code>try-restart</code>	<code>try-restart</code>
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.	<code>reload</code>	<code>reload</code>
Reloading or restarting. Reloads services if reloading is supported, otherwise restarts them. If a service is not yet running it will be started.	<code>reload-or-restart</code>	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.	<code>reload-or-try-restart</code>	n/a
Getting detailed status information. Lists information about the status of services. The <code>systemd</code> command shows details such as	<code>status</code>	<code>status</code>

Task	systemd Command	System V init Command
description, executable, status, cgroup, and messages last issued by a service (see Section 10.6.8, “Debugging Services”). The level of details displayed with the System V init differs from service to service.		
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</u> Command	System V init Command
Enabling.	<u>systemctl enable <my_service(s)></u>	<u>insserv <my_service(s)></u>

Task	<u>systemd</u> Command	System V init Command
Disabling.	<u>systemctl disable <my_service(s)>.service</u>	<u>insserv -r <my_service(s)></u>
Checking. Shows whether a service is enabled or not.	<u>systemctl is-enabled <my_service></u>	n/a
Re-enabling. Similar to restarting a service, this commands first disables and then enables a service. Useful to re-enable a service with its defaults.	<u>systemctl reenab <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 vs. 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 command line option 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 run-level	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> , <u>get</u> ,	Single-user mode
2	<u>runlevel2.target</u> , <u>multi-user.target</u> ,	Local multiuser without remote network
3	<u>runlevel3.target</u> , <u>multi-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
6	<u>runlevel6.target</u> , <u>reboot.target</u> , <u>get</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	systemd Command	System V init Command
Change the current target/run-level	<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 systemd 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/<my_target>.target /etc/systemd/system/default.target</u>	Use the Services Manager or change the line <u>id: X:initdefault:</u> <u>in /etc/inittab</u>
Change the default runlevel for the current boot process	Enter the following option at the boot prompt <u>systemd.unit= <my_target>.target</u>	Enter the desired run-level number at the boot prompt.
Show a target's/runlevel's dependencies	<u>systemctl show -p "Requires" <my_target>.target</u> <u>systemctl show -p "Wants" <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 conveniently 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
ntpd.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  
6472ms systemd-modules-load.service  
5833ms remount-rootfs.service  
4597ms network.service  
4254ms systemd-vconsole-setup.service  
4096ms postfix.service  
2998ms xdm.service  
2483ms localnet.service  
2470ms SuSEfirewall2_init.service  
2189ms avahi-daemon.service  
2120ms systemd-logind.service  
1210ms xinetd.service  
1080ms ntp.service  
[...]  
75ms fbset.service  
72ms purge-kernels.service  
47ms dev-vda1.swap  
38ms bluez-coldplug.service  
35ms splash_early.service
```

Services Start-Up Time Graphics

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


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*.

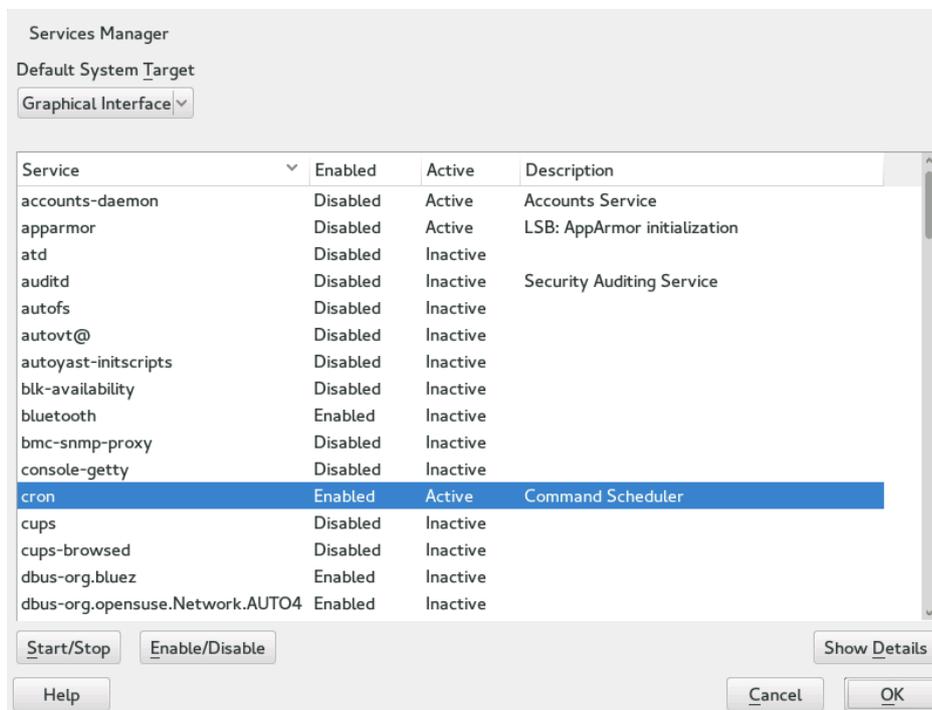


FIGURE 10.1: SERVICES MANAGER

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 Customization of systemd

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



Warning: Avoiding Overwritten Customizations

Always do `systemd` customization in `/etc/systemd/`, *never* in `/usr/lib/systemd/`. Otherwise your changes will be overwritten by the next update of `systemd`.

10.5.1 Customizing Service Files

The `systemd` service files are located in `/usr/lib/systemd/system`. If you want to customize them, proceed as follows:

1. Copy the files you want to modify from `/usr/lib/systemd/system` to `/etc/systemd/system`. Keep the file names identical to the original ones.
2. Modify the copies in `/etc/systemd/system` according to your needs.
3. For an overview of your configuration changes, use the `systemd-delta` command. It can compare and identify configuration files that override other configuration files. For details, refer to the `systemd-delta` man page.

The modified files in /etc/systemd will take precedence over the original files in /usr/lib/systemd/system, provided that their file name is the same.

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/<my_service>.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. Once you have finished setting up the target, reload the systemd configuration to make the new target available:

```
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:

```
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 2014-09-09 15:30:36 CEST; 1 weeks 6 days ago
Docs: man:tmpfiles.d(5)
      man:systemd-tmpfiles(8)

Sep 09 15:30:36 jupiter systemd[1]: Starting Daily Cleanup of Temporary Directories.
Sep 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 10.6.8, “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 `systemd-journalctl` to display the complete log messages starting with the oldest entries. Refer to `man 1 systemd-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

```
systemctl snapshot <my_snapshot>.snapshot
```

Delete a Snapshot

```
systemctl delete <my_snapshot>.snapshot
```

View a Snapshot

```
systemctl show <my_snapshot>.snapshot
```

Activate a Snapshot

```
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 installes 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):

```
systemctl daemon-reload
systemctl enable before
```

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

```
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
    ├──ntpd.service
    │   └─1328 /usr/sbin/ntpd -p /var/run/ntp/ntpd.pid -g -u ntp:ntp -c /etc/ntp.conf
    ├──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.

```
systemctl kill <my_service>
```

Sending SIGNAL to a Service

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

```
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:

```
systemctl kill -s SIGHUP --kill-who=main <my_service>
```

10.6.8 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 Jun 2012 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

Jun 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:

```
systemctl status ntp
systemctl --lines=20 status ntp
```

Show Service Messages in Append Mode

To display a “live stream” of service messages, use the **--follow** option, which works like **tail -f**:

```
systemctl --follow status ntp
```

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 SUSE Linux Enterprise 12 (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, from user processes, from standard input and error of system services. The `systemd-journald` service is on by default:

```
# 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/` with correct ownership and permissions must exist, where 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.

```
# 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:

```
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:

```
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 a 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"](#)).

```
# 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

```
# 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, -2 for the second previous, etc. The second column contains the boot ID, and then the limiting time stamps of the specific boot follow.

Show all messages from the current boot:

```
# journalctl -b
```

If you need to see journal message from the previous boot, add an offset parameter. The following example outputs the previous boot messages:

```
# journalctl -b -1
```

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

```
# 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, the 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", which 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:

```
# journalctl --since "now" -f
```

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

```
# 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 event more. See [man 7 systemd.journal-fields](#) for a list of default fields.

Show messages produced by a specific process ID:

```
# 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):

```
# journalctl _TRANSPORT=kernel
```

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

```
# journalctl _TRANSPORT=stdout
```

Show messages produced by a specified service only:

```
# journalctl __SYSTEMD_UNIT=avahi-daemon.service
```

If two different fields are specified, only entries that match both expressions at the same time are shown:

```
# journalctl __SYSTEMD_UNIT=avahi-daemon.service _PID=1488
```

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

```
# 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:

```
# 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:

```
# 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:

```
# 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:

```
# 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

```
# 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.

```
# rpm -q rsyslog
rsyslog-7.4.8-2.16.x86_64
```

- `rsyslog` service is enabled.

```
# systemctl is-enabled rsyslog
enabled
```

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

```
ForwardToSyslog=yes
```

11.6 Using YaST to Filter the systemd Journal

For an easy way of filtering the systemd journal (without having to deal with the `journalctl` syntax), you can use the YaST journal module. After installing it with **`sudo zypper in yast2-journal`**, start it from YaST by selecting *System > Systemd Journal*. Alternatively, start it from command line by entering **`sudo yast2 journal`**.

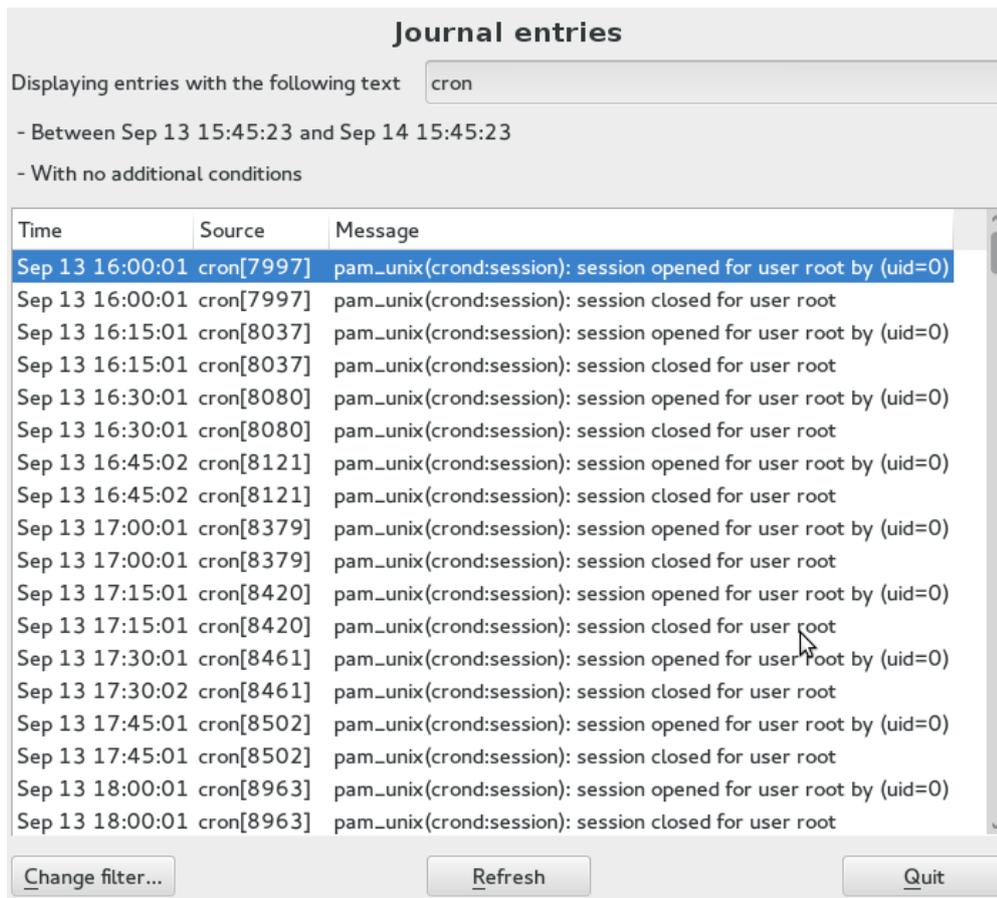


FIGURE 11.1: YAST SYSTEMD JOURNAL

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.

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 of the traditional GRUB boot loader—now called “GRUB 2 Legacy”. GRUB 2 has become the default boot loader in openSUSE® Leap since version 12. A YaST module is available for configuring the most important settings. The boot procedure as a whole is outlined in *Chapter 9, Booting a Linux System*. 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 is automatically generated by the grub2-mkconfig command, and should not be edited.

/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.

/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 **grub2-mkconfig** command. Their instructions are integrated into the main configuration file /boot/grub/grub.cfg.

/etc/sysconfig/bootloader

This configuration file is used when configuring the boot loader with YaST and every time a new kernel is installed. It is evaluated by the perl-bootloader which modifies the boot loader configuration file (for example /boot/grub2/grub.cfg for GRUB 2) accordingly. /etc/sysconfig/bootloader is not a GRUB 2-specific configuration file—the values are applied to any boot loader installed on openSUSE Leap.

/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** to activate the changes. However, this is not necessary when changing the configuration with YaST, since it will automatically run **grub2-mkconfig**.

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 by the `grub2-mkconfig` from the file `/etc/default/grub` and scripts found in the `/etc/grub.d/` directory. 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:

```
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`, run `grub2-mkconfig` to update the main configuration file.



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-reboot` 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.

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 vbeinfo 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 read *Section 3.3, "System Rollback by Booting from Snapshots"*.



Note: Parameter Handling

All *_DEFAULT parameters can be handled manually or by YaST.

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). For a complete list of possible parameters, see <http://en.opensuse.org/Linuxrc>.

12.2.3 Scripts in `/etc/grub.d`

The scripts in this directory are read during execution of the `grub2-mkconfig` command, and 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. The most important scripts are:

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 OS X.

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.

90_persistent

This is a special script that copies a corresponding part of the `grub.cfg` file and outputs it back unchanged. This way you can modify that part of `grub.cfg` directly and the change survives the execution of `grub2-mkconfig`.

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.

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.

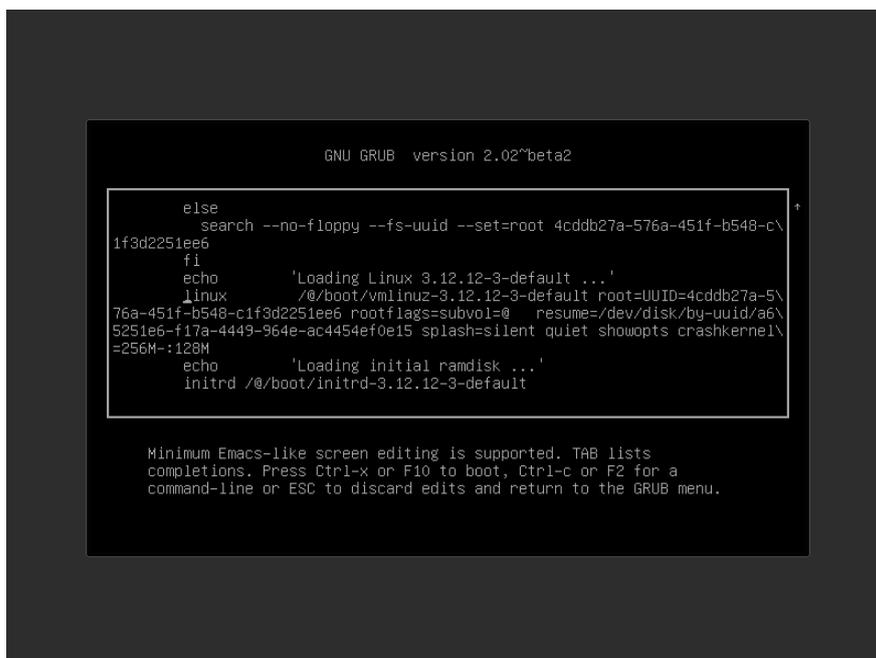


FIGURE 12.1: GRUB 2 BOOT EDITOR

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 16 "Common Problems and Their Solutions", Section 16.2.3 "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 options, 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. Run **`grub2-mkconfig`** to import the changes into the main configuration file.

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.

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.

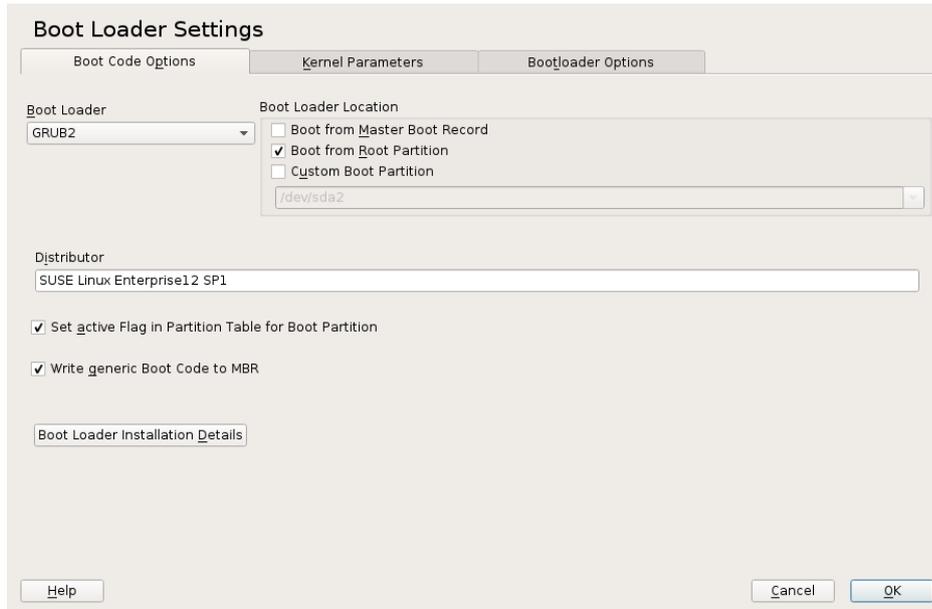


FIGURE 12.2: BOOT CODE OPTIONS

! 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 Modifying the Boot Loader Location

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 first disk (according to the boot sequence preset in the BIOS).

Boot from Root Partition

This installs the boot loader in the boot sector of the partition (this is the default).

Custom Boot Partition

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

2. Click *OK* to apply your changes.

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. For more information look at *Section 12.2.4, "Mapping between BIOS Drives and Linux Devices"*.

PROCEDURE 12.2: SETTING THE DISK ORDER

1. Open the *Boot Code Options* tab.
2. Click *Boot Loader Installation Details*.
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 options can be configured via the *Boot Loader Options* tab.

12.3.3.1 *Boot Loader Options* Tab

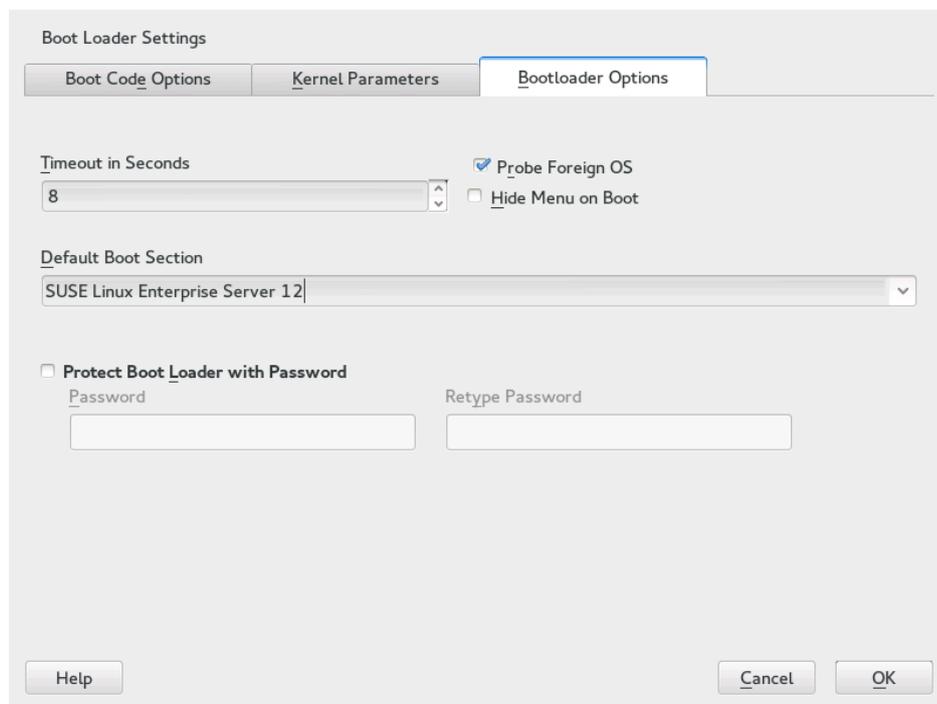


FIGURE 12.3: **BOOT LOADER OPTIONS**

Boot Loader Time-Out

Change the value of *Time-Out in Seconds* by typing in a new value and clicking the appropriate arrow key with your mouse.

Probe Foreign OS

When selected, the boot loader searches for other systems like Windows or other Linux installations.

Hide Menu on Boot

Hides the boot menu and boots the default entry.

Adjusting the Default Boot Entry

Select the desired entry from the “Default Boot Section” list. Note that the “>” sign in the boot entry name delimits the boot section and its subsection.

Protect Boot Loader with Password

Protects the boot loader and the system with an additional password. For more details see [Section 12.2.6, “Setting a Boot Password”](#).

12.3.3.2 Kernel Parameters Tab

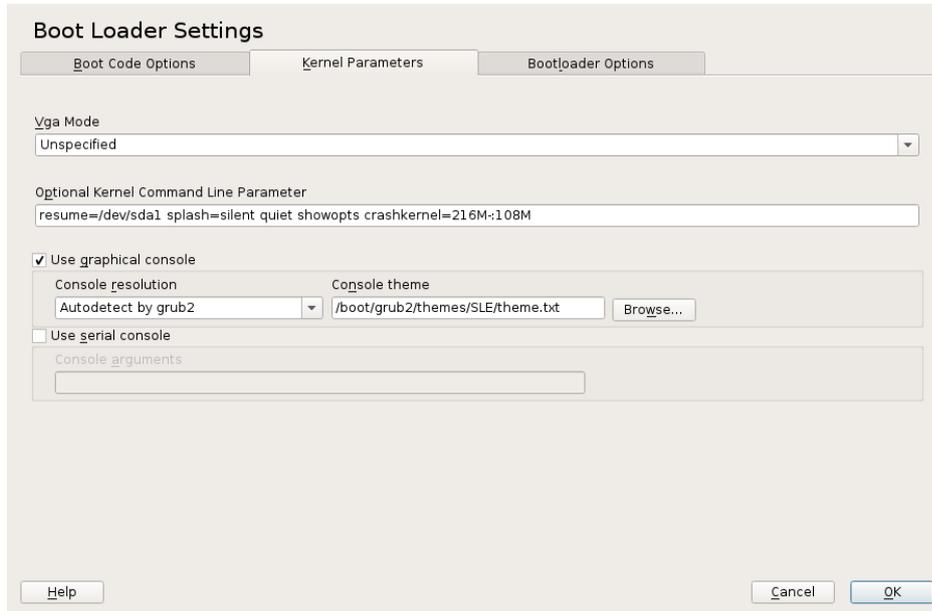


FIGURE 12.4: KERNEL PARAMETERS

VGA Mode

The VGA Mode option specifies the default screen resolution during the boot process.

Kernel Command Line Parameter

The optional kernel parameters are added at the end of the default parameters. For a list of all possible parameters, see <http://en.opensuse.org/Linuxrc>.

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 can be then set from the *Console resolution* list, and graphical theme definition file can be specified with the *Console theme* file-chooser.

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>.

12.3.3.3 *Boot Code Options Tab*

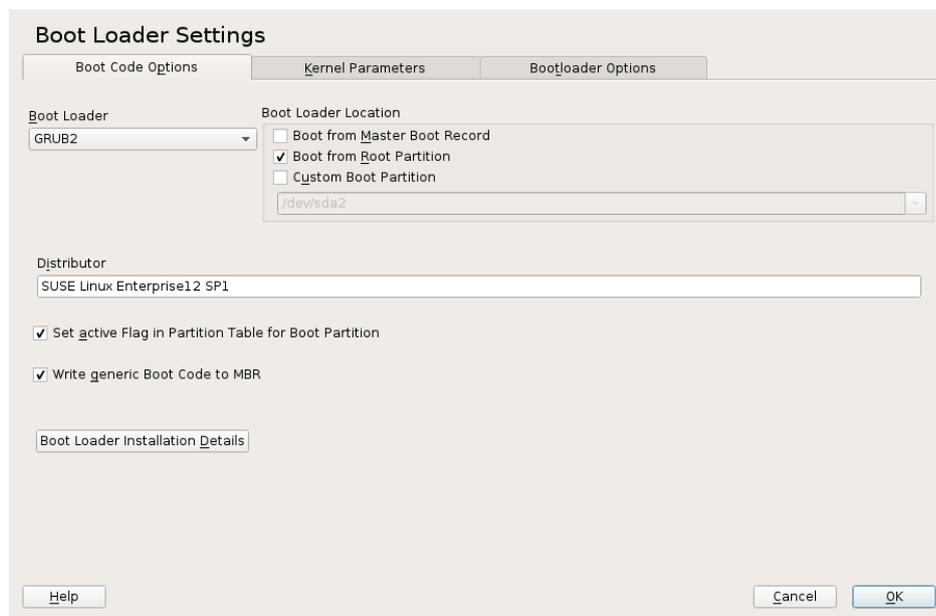


FIGURE 12.5: CODE OPTIONS

Set Active Flag in Partition Table for Boot Partition

Activates the partition that contains the boot loader. Some legacy operating systems (such as Windows) can only boot from an active partition.

Write Generic Boot Code to MBR

Replaces the current MBR with generic, operating system independent code.

12.4 Differences in Terminal Usage on z Systems

On 3215 and 3270 terminals there are some differences and limitations on how to move the cursor and how to issue editing commands within GRUB 2.

12.4.1 Limitations

Interactivity

Interactivity is strongly limited. Typing often does not result in visual feedback. To see where the cursor is, type an underscore (`_`).



Note: 3270 vs. 3215

The 3270 terminal is much better at displaying and refreshing screens than the 3215 terminal.

Cursor Movement

“Traditional” cursor movement is not possible. `Alt`, `Meta`, `Ctrl` and the cursor keys do not work. To move the cursor, use the key combinations listed in [Section 12.4.2, “Key Combinations”](#).

Caret

The caret (`^`) is used as a control character. To type a literal `^` followed by a letter, type `^`, `^`, LETTER.

Enter

The `Enter` key does not work, use `^J` instead.

12.4.2 Key Combinations

Common Substitutes:	<code>^J</code>	engage (“Enter”)
	<code>^L</code>	abort, return to previous “state”
	<code>^I</code>	tab completion (in edit and shell mode)
Keys Available in Menu Mode:	<code>^A</code>	first entry
	<code>^E</code>	last entry
	<code>^P</code>	previous entry
	<code>^N</code>	next entry
	<code>^G</code>	previous page
	<code>^C</code>	next page

	-F	boot selected entry or enter submenu (same as -J)
	E	edit selected entry
	C	enter GRUB-Shell
Keys Available in Edit Mode:	-P	previous line
	-N	next line
	-B	backward char
	-F	forward char
	-A	beginning of line
	-E	end of line
	-H	backspace
	-D	delete
	-K	kill line
	-Y	yank
	-O	open line
	-L	refresh screen
	-X	boot entry
	-C	enter GRUB-Shell
Keys Available in Command Line Mode:	-P	previous command
	-N	next command from history
	-A	beginning of line

	end of line
	backward char
	forward char
	backspace
	delete
	kill line
	discard line
	yank

12.5 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 STD-OUT where it can be reviewed. Use **grub2-script-check** after `/boot/grub2/grub.cfg` has been written to check its syntax.

Important: **grub2-mkconfig** Can Not Repair UEFI Secure Boot Tables

If you are using UEFI Secure Boot and your system is not reaching GRUB 2 correctly any more, 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-check-config /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.

12.6 More Information

Extensive information about GRUB 2 is available at <http://www.gnu.org/software/grub/>⁷. Also refer to the **grub** info page. You can also search for the keyword “GRUB 2” in the Technical Information Search at <http://www.suse.com/support>⁷ to get information about special issues.

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: Essentially, 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.

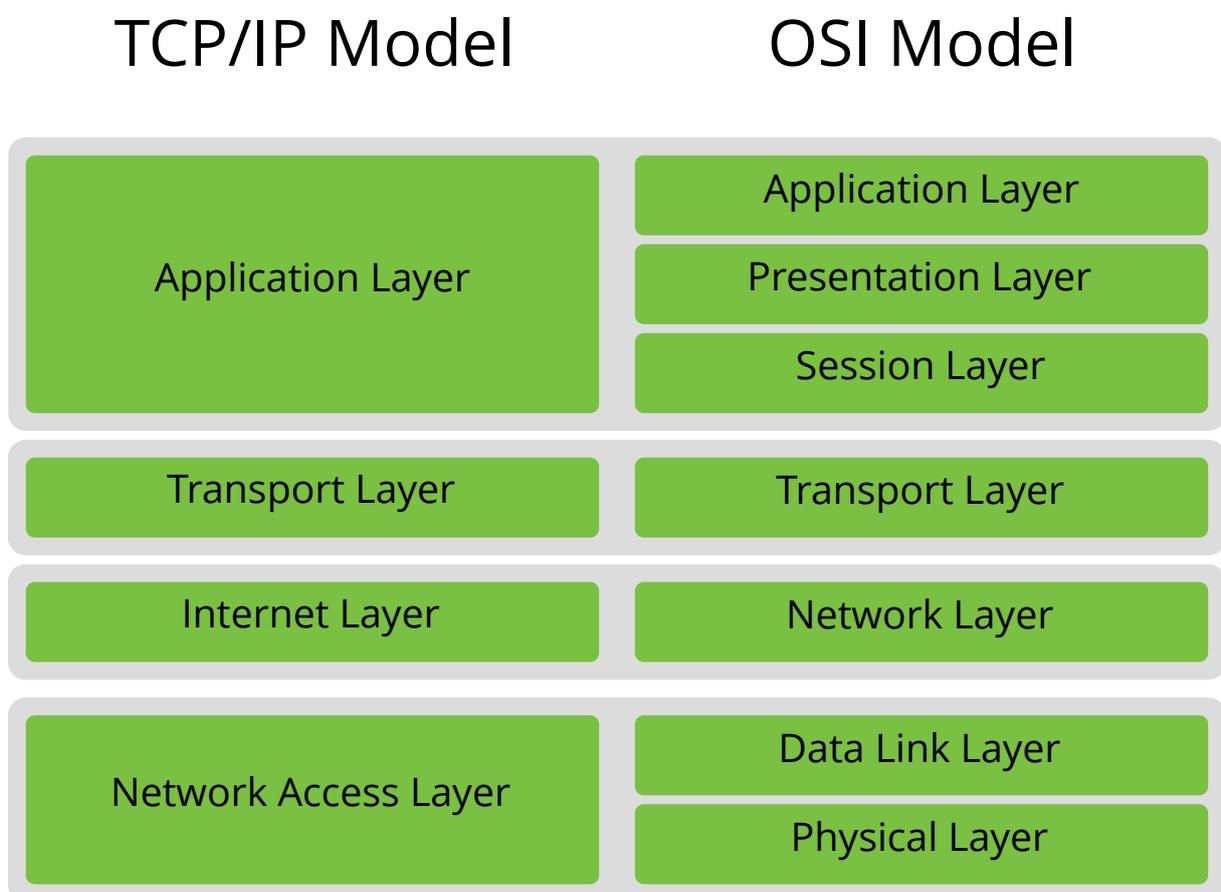


FIGURE 13.1: SIMPLIFIED LAYER MODEL FOR TCP/IP

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.

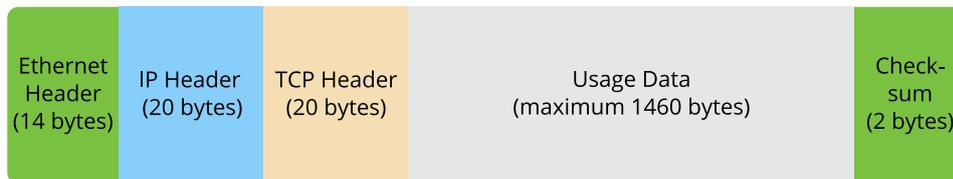


FIGURE 13.2: TCP/IP ETHERNET PACKET

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 100 Mbit/s FDDI network or via a 56-Kbit/s modem line. 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 WWW (World Wide Web), 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 as soon as it enters the corresponding area, so you can be reached under the same number everywhere and are able to 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*—by addressing several hosts as parts of a group (which 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 is lacking in two important aspects: 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 countered 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—when 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 as soon as 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 by the fact that 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, check or uncheck the *Enable IPv6* option as necessary. If you want 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 2640

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 `host-name.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 with the help of 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.

If you want 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. If you want to manually configure a card that was not automatically detected, see [Section 13.4.1.3, “Configuring an Undetected Network Card”](#). If you want 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.

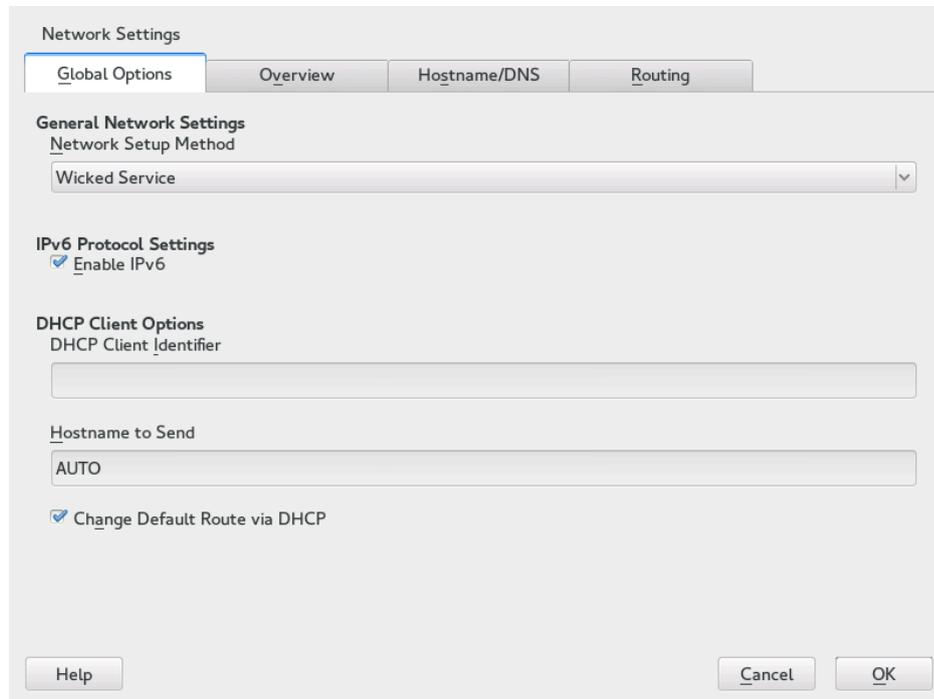


FIGURE 13.3: CONFIGURING NETWORK SETTINGS

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*.

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 later 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 as soon as available. It is similar to the *At Boot Time* option, and only differs in the fact 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 at all. 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 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>.

13.4.1.2.8 Configuring the Firewall

Without having to enter the detailed firewall setup as described in *Book “Security Guide”, Chapter 15 “Masquerading and Firewalls”, Section 15.4.1 “Configuring the Firewall with YaST”*, you can determine the basic firewall setup 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 PCMCIA or 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. See the **ethtool** manual page for available options. 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** prepends `-s interface_name`.

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* and, if needed, the *Domain Name*. The domain is especially important if the machine is a mail server. Note that the host name is global and applies to all set 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. 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 /etc/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 sufficient in most cases.

If the *Only Manually* option is selected, **netconfig** is not allowed to modify the /etc/resolv.conf file. However, this file can be edited manually.

If the *Custom Policy* option is selected, a *Custom Policy Rule* string defining the merge policy should be specified. The string consists of a comma-separated list of interface names to be considered a valid source of settings. Except for complete interface names, basic wild cards to match multiple interfaces are allowed, as well. For example, eth* ppp? will first target all eth and then all ppp0-ppp9 interfaces. There are two special policy values that indicate how to apply the static settings defined in the /etc/sysconfig/network/config file:

STATIC

The static settings need to be merged together with the dynamic settings.

STATIC_FALLBACK

The static settings are used only when no dynamic configuration is available.

For more information, see the man page of **netconfig(8)** (**man 8 netconfig**).

4. Enter the *Name Servers* and fill in the *Domain Search* list. Name servers must be specified by IP addresses, such as 192.168.1.116, not by host names. Names specified in the *Domain Search* tab are domain names used for resolving host names without a specified domain. If more than one *Domain Search* is used, separate domains with commas or white space.
5. To activate the configuration, confirm the settings.

It is also possible to edit the host name using YaST from the command line. The changes made by YaST take effect immediately (which is not the case when editing the `/etc/HOSTNAME` file manually). To change the host name, use the following command:

```
yast dns edit hostname=hostname
```

To change the name servers, use the following commands:

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

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 his 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 Guide", Chapter 9 "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, that interact with each other in a not-really-well-defined way, with side effects that are difficult to be aware of, 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, with address configuration, the administrator may request that an interface should be configured via DHCP or IPv4 zeroconf, and all the address configuration service does is obtain the lease from its server, and pass it on to the `wicked` server process, which 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 example, for an Ethernet device, you may want to 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 `org.opensuse.Network.Ethernet`, which is a 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 that needs to be created first—such as a VLAN, which is really a virtual network interface that sits on top of an Ethernet device. For these, `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 Supported Features

`wicked` currently supports:

- 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 as soon as the device is available (interface hotplugging) and set up the IP configuration when a link (carrier) is detected.

13.6.1.2 Using **wicked**

On SUSE Linux Enterprise, **wicked** is running by default. In case 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 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.

13.6.1.6 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 different 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"/>
</dbus-service>
```

```
<!-- 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` interface and defines commands to execute for the actions of this interface. Further, the declaration can define and initialize environment variables passed to the actions.

13.6.1.7 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 `/etc/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/sysconfig/network/ifcfg-*`

These files contain the traditional configurations for network interfaces. In SUSE Linux Enterprise 11, 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 SUSE Linux Enterprise Server 12 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 13.6.2.2, “/etc/sysconfig/network/config, /etc/sysconfig/network/dhcp, and /etc/sysconfig/network/wireless”](#).

13.6.2.2 `/etc/sysconfig/network/config, /etc/sysconfig/network/dhcp, and /etc/sysconfig/network/wireless`

The file `config` contains general settings for the behavior of `ifup`, `ifdown` and `ifstatus`. `dhcp` contains settings for DHCP and `wireless` for wireless LAN cards. The variables in all three configuration files are commented. Some variables from `/etc/sysconfig/network/config` can also be used in `ifcfg-*` files, where they are given a higher priority. The `/etc/sysconfig/network/ifcfg.template` file lists variables that can be specified in a per interface scope. However, most of the `/etc/sysconfig/network/config` variables are global and cannot be overridden in `ifcfg`-files. For example, `NETWORKMANAGER` or `NETCONFIG_*` variables are global.



Note: Using DHCPv6

In SUSE Linux Enterprise 11, DHCPv6 used to work even on networks where IPv6 Router Advertisements (RAs) were not configured properly. Starting with SUSE Linux Enterprise 12, DHCPv6 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 those networks where the router cannot be configured correctly, there is an `ifcfg` option that 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
```

13.6.2.3 `/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
```

13.6.2.4 `/etc/resolv.conf`

The domain to which the host belongs is specified in `/etc/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 3 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, “/etc/resolv.conf”](#).

However, the `/etc/resolv.conf` should not be edited by hand. Instead, it is generated by the `netconfig` script. 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 `/etc/resolv.conf`, for example:

```
debug attempts:1 timeout:10
```

For more information, see the `resolv.conf` man page.

NETCONFIG_DNS_RESOLVER_SORTLIST

list of up to 10 items, for example:

```
130.155.160.0/255.255.240.0 130.155.0.0
```

For more information, see the `resolv.conf` man page.

To disable DNS configuration using `netconfig`, set `NETCONFIG_DNS_POLICY=''`. For more information about `netconfig`, see the `netconfig(8)` man page ([man 8 netconfig](#)).

EXAMPLE 13.6: `/etc/resolv.conf`

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

13.6.2.5 `/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 a modificatory 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, if you want 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`.

13.6.2.6 `/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
```

13.6.2.7 `/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
```

13.6.2.8 `/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

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

13.6.2.9 `/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.

<u>netmasks</u>	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</u> , <u>nisplus</u>	NIS, see also <i>Book "Security Guide", Chapter 3 "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>

13.6.2.10 `/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` and `groups` 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 or groups. `hosts` is not cached by default, because the mechanism in `nscd` to cache hosts makes the local system unable to trust forward and reverse lookup checks. Instead of asking `nscd` to cache names, set up a caching DNS server.

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:

```
systemctl restart nscd
```

13.6.2.11 `/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 **ip link set device_name** . For example, to deactivate device eth0, enter **ip link set eth0 down**. To activate it again, use **ip link set eth0 up**.

After activating a device, you can configure it. To set the IP address, use **ip addr add ip_address + dev device_name**. For example, to set the address of the interface eth0 to 192.168.12.154/30 with standard broadcast (option **brd**), enter **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 **ip route add gateway_ip_address**. To translate one IP address to another, use **nat: ip route add nat ip_address via other_ip_address**.

To display all devices, use **ip link ls**. To display the running interfaces only, use **ip link ls up**. To print interface statistics for a device, enter **ip -s link ls device_name**. To view addresses of your devices, enter **ip addr**. In the output of the **ip addr**, also find information about MAC addresses of your devices. To show all routes, use **ip route show**.

For more information about using **ip**, enter **ip help** or see the **ip(8)** man page. The **help** option is also available for all **ip** subcommands. If, for example, you need help for **ip addr**, enter **ip addr help**. Find the **ip** manual in /usr/share/doc/packages/iproute2/ip-cre-f.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 as soon as 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 meaning 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.

xinetd

Starts `xinetd`. `xinetd` can be used to make server services available on the system. For example, it can start `vsftpd` whenever an FTP connection is initiated.

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 SUSE Linux Enterprise Server you can build a router with features such as NAT (Network Address Translation) or advanced firewalling.

The following are basic steps to turn SUSE Linux Enterprise Server 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, such as IPv6 does not accept an IPv6 RA (router advertisement) anymore, what also prevents the creation of a default route.

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

```

Lastly configure the firewall. In `SuSEfirewall2`, must set `FW_ROUTE="yes"` (otherwise it will also reset forwarding `sysctl` again) and define the interfaces in the `FW_DEV_INT`, `FW_DEV_EXT` (and `FW_DEV_DMZ`) zone variables as needed, perhaps also `FW_MASQUERADE="yes"` and `FW_MASQ_DEV`.

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 `mode=active-backup` which means that a different slave device will become active if the active slave fails.

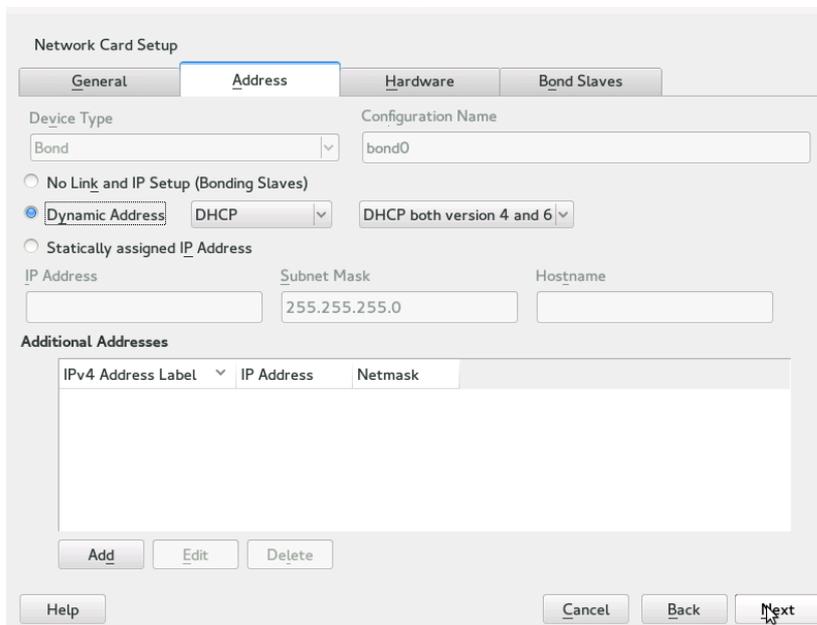


Tip: Bonding and Xen

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*.



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*. The modes that are available for configuration are the following:

- balance-rr
- active-backup

- balance-xor
- broadcast
- 802.3ad
 - 802.3ad is the standardized LACP “IEEE 802.3ad Dynamic link aggregation” mode.
- balance-tlb
- balance-alb

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.

All modes, and many more options are explained in detail in the *Linux Ethernet Bonding Driver HOWTO* found at </usr/src/linux/Documentation/networking/bonding.txt> after installing the package [kernel-source](#).

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 as soon as 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 to allow to replacement of defective hardware (a network card in the same slot but with a different MAC), and to avoid confusion as 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 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*). That 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.

13.9.1 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.
- 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 to, for example, 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.

13.9.2 Installing Open vSwitch

1. Install Open vSwitch and supplemental 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
```

13.9.3 Overview of Open vSwitch Daemons and Utilities

Open vSwitch consists of several components. Among them are a kernel module and various userspace components. The kernel module is used for accelerating the data path, but is not necessary for a minimal Open vSwitch installation.

13.9.3.1 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.

13.9.3.2 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.

13.9.4 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`.

13.9.5 Using Open vSwitch Directly with KVM

After having created the bridge as described before in [Section 13.9.4, “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"*.

13.9.6 Using Open vSwitch with libvirt

After having created the bridge as described before in *Section 13.9.4, "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 13.9.4, "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"*.

13.9.7 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 SUSE Linux Enterprise.

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 SUSE Linux Enterprise, the mainboard and the on-board firmware could be considered the “platform”. Or, put slightly differently, 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.

Essentially, if you want 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 SUSE Linux Enterprise

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.

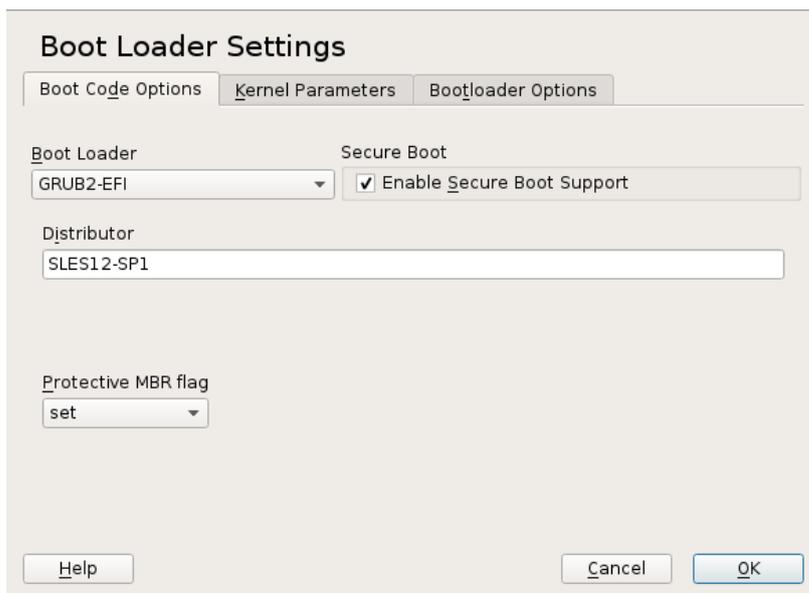


FIGURE 14.1: SECURE BOOT SUPPORT

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 essentially requires having a boot loader with a digital signature that the firmware recognizes as a trusted key. To be useful for SUSE Linux Enterprise customers, 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).

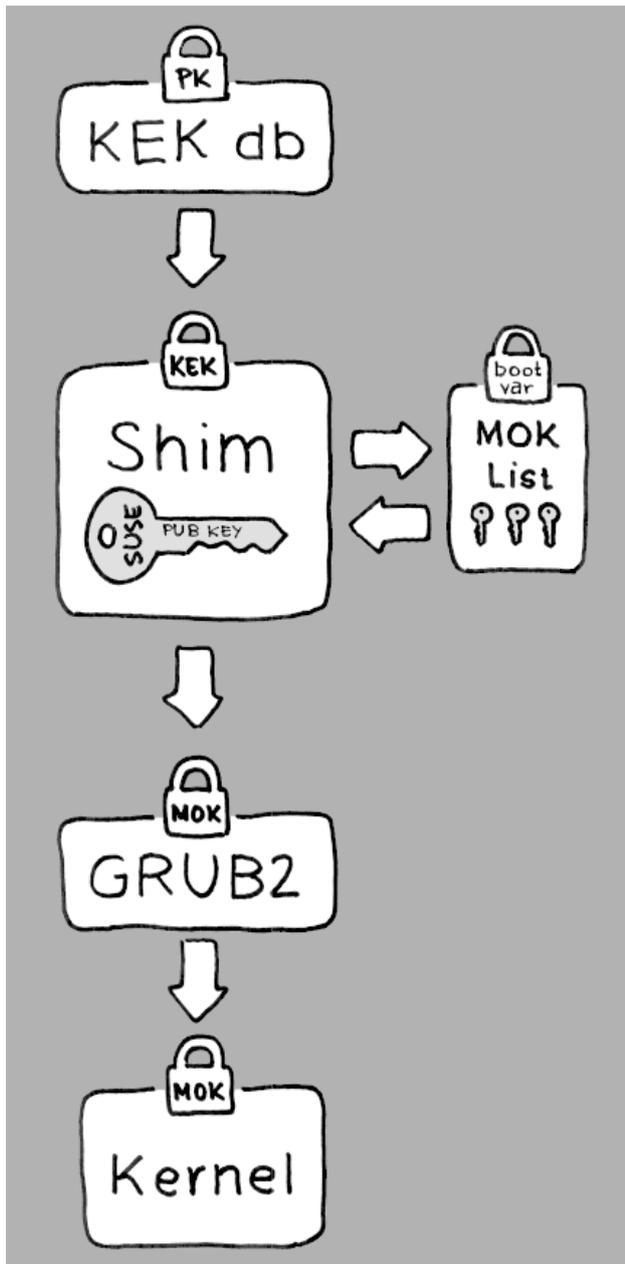


FIGURE 14.2: UEFI: SECURE BOOT PROCESS

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 ELILO or GRUB 2 and verify it; this boot loader in turn will load kernels signed by a SUSE key only. SUSE provides this functionality since SLE11 SP3 on fresh installations with UEFI Secure Boot enabled.

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, but 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 matter that is both secure and open source friendly, and thus compatible with GPLv3.

SUSE starts with `shim`—a small and simple EFI boot loader—which was originally developed by Fedora. It is signed by a certificate signed by the SUSE KEK and a Microsoft-issued certificate, based on which KEKs are available in the UEFI key database on the system.

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)

If the user (“machine owner”) wants to replace any components of the boot process, Machine Owner Keys (MOKs) are to be used. The `mokutils` tool will help with signing components and managing MOKs.

The enrollment process begins with rebooting the machine and interrupting the boot process (for example, pressing a key) when `shim` loads. `shim` will then go into enrollment mode, allowing the user to replace the default SUSE key with keys from a file on the boot partition. If the user chooses to do so, `shim` will then calculate a hash of that file and put the result in a “Boot Services Only” variable. This allows `shim` to detect any change of the file made outside of Boot Services and thus avoid tampering with the list of user-approved MOKs.

All of this happens during boot time—only verified code is executing now. Therefore, only a user present at the console can use the machine owner's set of keys. It cannot be malware or a hacker with remote access to the OS because hackers or malware can only change the file, but not the hash stored in the “Boot Services Only” variable.

The boot loader, after having been loaded and verified by `shim`, will call back to `shim` when it wants to verify the kernel—to avoid duplication of the verification code. `shim` will use the same list of MOKs for this and tell the boot loader whether it can load the kernel.

This way, you can install your own kernel or boot loader. It is only necessary to install a new set of keys and authorize them by being physically present during the first reboot. Because MOKs are a list and not just a single MOK, you can make `shim` trust keys from several different vendors, allowing dual- and multi-boot from the boot loader.

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:

```
openssl pkcs12 -export -inkey key.asc -in cert.pem \  
-name kernel_cert -out cert.p12
```

3. Generate an NSS database for use with **pesign**:

```
certutil -d . -N
```

4. Import the key and the certificate contained in PKCS#12 into the NSS database:

```
pk12util -d . -i cert.p12
```

5. “Bless” the kernel with the new signature using **pesign**:

```
pesign -n . -c kernel_cert -i arch/x86/boot/bzImage \  
-o vmlinuz.signed -s
```

6. List the signatures on the kernel image:

```
pesign -n . -S -i vmlinuz.signed
```

At that point, you can install the kernel in `/boot` as usual. Because the kernel now has a custom signature the certificate used for signing needs to be imported into the UEFI firmware or MOK.

7. Convert the certificate to the DER format for import into the firmware or MOK:

```
openssl x509 -in cert.pem -outform der -out cert.der
```

8. Copy the certificate to the ESP for easier access:

```
sudo cp cert.der /boot/efi/
```

9. Use **mokutil** to launch the MOK list automatically.

- - a. Import the certificate to MOK:

```
mokutil --root-pw --import cert.der
```

The `--root-pw` option enables usage of the `root` user directly.

- b. Check the list of certificates that are prepared to be enrolled:

```
mokutil --list-new
```

c. Reboot the system; `shim` should launch MokManager. You need to enter the `root` password to confirm the import of the certificate to the MOK list.

d. Check if the newly imported key was enrolled:

```
mokutil --list-enrolled
```

- a. Alternatively, this is the procedure 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 SLE) 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 SUSE Linux Enterprise 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 disk of the SUSE Linux Enterprise 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 get 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

- <http://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):
 - <http://www.suse.com/blogs/uefi-secure-boot-plan/>
 - <http://www.suse.com/blogs/uefi-secure-boot-overview/>
 - <http://www.suse.com/blogs/uefi-secure-boot-details/>
- <http://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 programs `bash`, `cron`, `logrotate`, `locate`, `ulimit` and `free` are very important for system administrators and many users. Man pages and info pages are two useful sources of information about commands, but both are not always available. GNU Emacs is a popular and very configurable text editor.

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:

```
mv ~/.bashrc ~/.bashrc.old
```

```
cp /etc/skel/.bashrc ~/.bashrc
mv ~/.profile ~/.profile.old
cp /etc/skel/.profile ~/.profile
```

Then copy personal adjustments back from the *.old files.

15.1.2 The cron Package

If you want to run commands regularly and automatically in the background at predefined times, cron is the tool to use. cron is driven by specially formatted time tables. Some come with the system and users can write their own 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.

A number of 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.

Configure logrotate with the file

`/etc/logrotate.conf`. In particular, the `include` specification primarily configures the additional files to read. Programs that produce log files install individual configuration files in `/etc/logrotate.d`. For example, such files ship with the packages `apache2` (`/etc/logrotate.d/apache2`) and `syslog-service` (`/etc/logrotate.d/syslog`).

EXAMPLE 15.3: EXAMPLE FOR /ETC/LOGROTATE.CONF

```
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
```

```

rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own lastlog or wtmp - we'll rotate them here
#/var/log/wtmp {
#   monthly
#   create 0664 root utmp
#   rotate 1
#}

# system-specific logs may be also be configured here.

```

logrotate is controlled through cron and is called daily by /etc/cron.daily/logrotate.

Important: Permissions

The create option reads all settings made by the administrator in /etc/permissions*. Ensure that no conflicts arise from any personal modifications.

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**

<u>-m</u>	The maximum resident set size
<u>-v</u>	The maximum amount of virtual memory available to the shell
<u>-s</u>	The maximum size of the stack
<u>-c</u>	The maximum size of core files created
<u>-a</u>	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/.bashrc`.

EXAMPLE 15.4: **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 instance, `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, plus 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 with the help of 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. If you want 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). This extension is also used by the desktop environment GNOME (`gswitchit`).



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 localizations (*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).

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

A yes or no variable. If set to no, root always works in the 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 connected by special characters:

```
LANG=<<language>[[<_<COUNTRY>].<Encoding>[@<Modifier>]]
```

15.4.1 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 instance, 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.

15.4.2 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_` name space prefixes. For example, use `LANG` instead of `RC_LANG`:

```
LANG=cs_CZ.UTF-8
LC_COLLATE=C
```

15.4.3 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.

15.4.4 For More Information

- *The GNU C Library Reference Manual*, Chapter “Locales and Internationalization”. It is included in `glibc-info`. The package is available from the SUSE Linux Enterprise SDK. The SDK is a module for SUSE Linux Enterprise and is available via an online channel from the SUSE Customer Center. Alternatively, go to <http://download.suse.com/>, search for `SUSE Linux Enterprise Software Development Kit` and download it from there. Refer to *Book “Start-Up”, Chapter 10 “Installing Add-On Products”* for details.
- Markus Kuhn, *UTF-8 and Unicode FAQ for Unix/Linux*, currently at <http://www.cl.cam.ac.uk/~mgk25/unicode.html>.
- *Unicode-HOWTO* by Bruno Haible, available at <http://tldp.org/HOWTO/Unicode-HOWTO-1.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 as soon as 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 enables you to customize `udev` device handling by, for example, 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 provided rules from the `/etc/udev/rules.d/*.rules` files once at start-up and keeps them in memory. If rules files are changed, added or removed,

the daemon can reload the in-memory representation of all rules with the command `udevadm control reload_rules`. 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 possibly 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. Every event is matched against all provided rules. All rules are located in the `/etc/udev/rules.d` directory.

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 that is located under `/etc/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 different 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 different 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}, \$attr{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 upwards for a matching device name.

SUBSYSTEMS

Let udev search the device path upwards for a matching device subsystem name.

DRIVERS

Let udev search the device path upwards for a matching device driver name.

ATTRS{filename}

Let udev search the device path upwards 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 a number of 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 different 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 possible 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/*

udev event matching rules.

/usr/lib/tmpfiles.d/ and /etc/tmpfiles.d/

Responsible for static /dev content.

/usr/lib/udev/*

Helper programs called from udev rules.

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.

udev

Information about the udev event managing daemon.

III Services

- 17 SLP 249
- 18 Time Synchronization with NTP 253
- 19 The Domain Name System 259
- 20 DHCP 284
- 21 Samba 300
- 22 Sharing File Systems with NFS 322
- 23 On-Demand Mounting with Autofs 333
- 24 The Apache HTTP Server 341
- 25 Setting up an FTP Server with YaST 383
- 26 The Proxy Server Squid 387

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:ypserv
```

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 that is provided with the SUSE Software Development Kit.

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 the help of reliable time servers in the network. It further enables the management of local reference clocks, such as radio-controlled clocks.



Note

To enable time synchronization by means of active directory, follow the instructions found at *Book "Security Guide", Chapter 6 "Active Directory Support", Section 6.3 "Configuring a Linux Client for Active Directory", Joining an AD Domain*.

18.1 Configuring an NTP Client with YaST

The NTP daemon (`ntpd`) coming with the `ntp` package is preset to use the local computer clock as a time reference. Using the hardware clock, however, only serves as a fallback for cases where no time source of better precision is available. YaST simplifies the configuration of an NTP client.

18.1.1 Basic Configuration

The YaST NTP client configuration (*Network Services > NTP Configuration*) consists of tabs. Set the start mode of `ntpd` and the server to query on the *General Settings* tab.

Only Manually

Select *Only Manually*, if you want to manually start the `ntpd` daemon.

Now and On Boot

Select *Now and On Boot* to start `ntpd` automatically when the system is booted. This setting is strongly recommended.

18.1.2 Changing Basic Configuration

The servers and other time sources for the client to query are listed in the lower part of the *General Settings* tab. Modify this list as needed with *Add*, *Edit*, and *Delete*. *Display Log* provides the possibility to view the log files of your client.

Click *Add* to add a new source of time information. In the following dialog, select the type of source with which the time synchronization should be made. The following options are available:

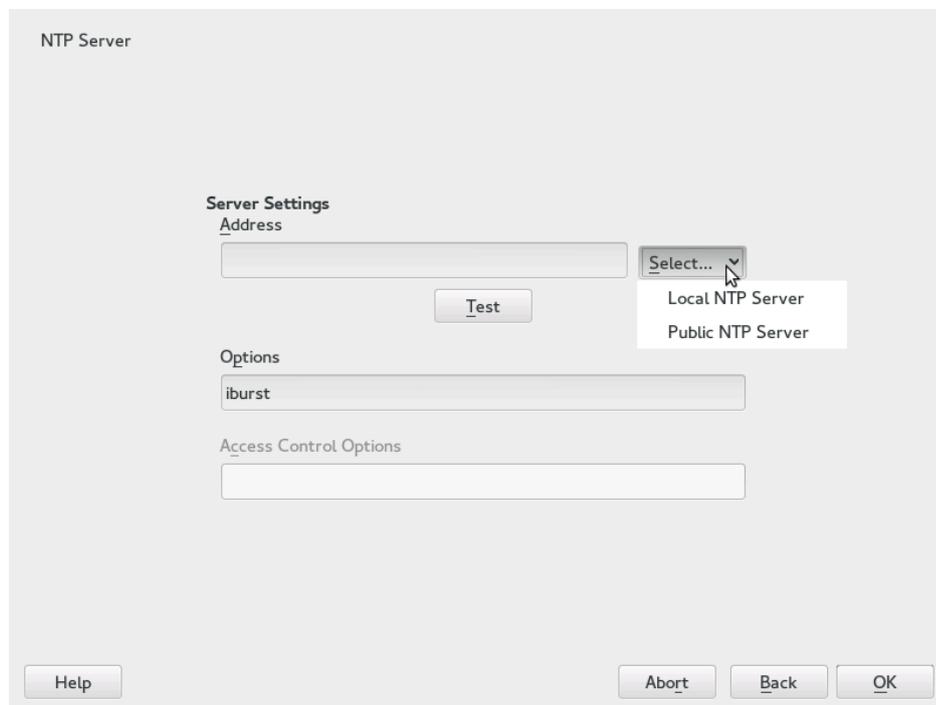


FIGURE 18.1: YAST: NTP SERVER

Server

In the pull-down *Select* list (see *Figure 18.1, “YaST: NTP Server”*), determine whether to set up time synchronization using a time server from your local network (*Local NTP Server*) or an Internet-based time server that takes care of your time zone (*Public NTP Server*). For a local time server, click *Lookup* to start an SLP query for available time servers in your network. Select the most suitable time server from the list of search results and exit the

dialog with *OK*. For a public time server, select your country (time zone) and a suitable server from the list under *Public NTP Server* then exit the dialog with *OK*. In the main dialog, test the availability of the selected server with *Test*. *Options* allows you to specify additional options for `ntpd`.

Using *Access Control Options*, you can restrict the actions that the remote computer can perform with the daemon running on your computer. This field is enabled only after checking *Restrict NTP Service to Configured Servers Only* on the *Security Settings* tab (see [Figure 18.2, "Advanced NTP Configuration: Security Settings"](#)). The options correspond to the `restrict` clauses in `/etc/ntp.conf`. For example, `nomodify notrap noquery` disallows the server to modify NTP settings of your computer and to use the trap facility (a remote event logging feature) of your NTP daemon. Using these restrictions is recommended for servers out of your control (for example, on the Internet).

Refer to [/usr/share/doc/packages/ntp-doc](#) (part of the `ntp-doc` package) for detailed information.

Peer

A peer is a machine to which a symmetric relationship is established: it acts both as a time server and as a client. To use a peer in the same network instead of a server, enter the address of the system. The rest of the dialog is identical to the *Server* dialog.

Radio Clock

To use a radio clock in your system for the time synchronization, enter the clock type, unit number, device name, and other options in this dialog. Click *Driver Calibration* to fine-tune the driver. Detailed information about the operation of a local radio clock is available in [/usr/share/doc/packages/ntp-doc/refclock.html](#).

Outgoing Broadcast

Time information and queries can also be transmitted by broadcast in the network. In this dialog, enter the address to which such broadcasts should be sent. Do not activate broadcasting unless you have a reliable time source like a radio controlled clock.

Incoming Broadcast

If you want your client to receive its information via broadcast, enter the address from which the respective packets should be accepted in this fields.

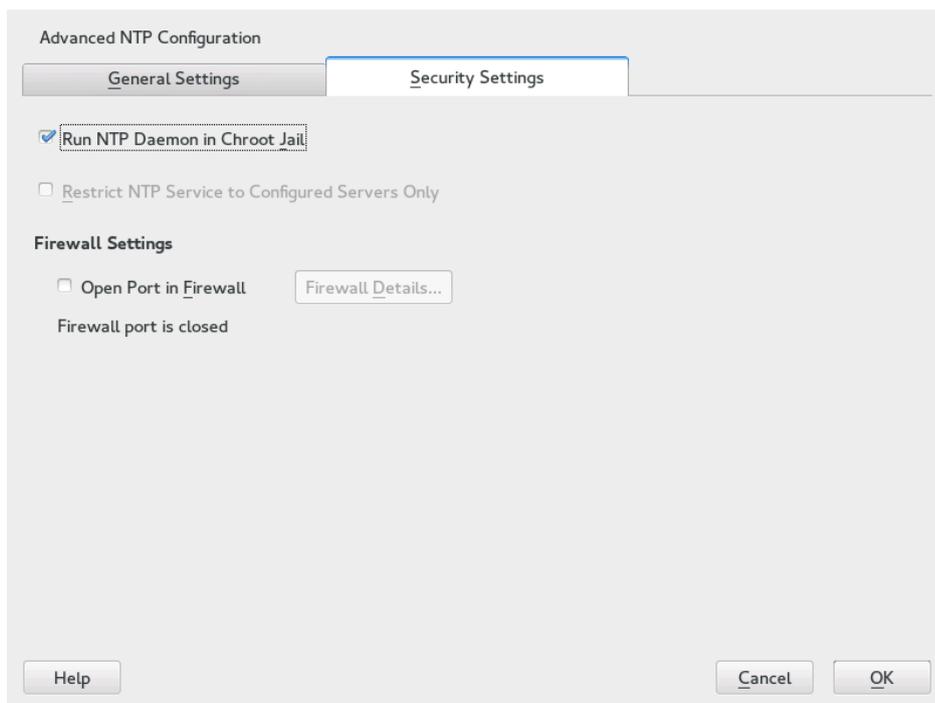


FIGURE 18.2: ADVANCED NTP CONFIGURATION: SECURITY SETTINGS

In the *Security Settings* tab (see [Figure 18.2, “Advanced NTP Configuration: Security Settings”](#)), determine whether `ntpd` should be started in a chroot jail. By default, *Run NTP Daemon in Chroot Jail* is activated. This increases the security in the event of an attack over `ntpd`, as it prevents the attacker from compromising the entire system.

Restrict NTP Service to Configured Servers Only increases the security of your system by disallowing remote computers to view and modify NTP settings of your computer and to use the trap facility for remote event logging. After being enabled, these restrictions apply to all remote computers, unless you override the access control options for individual computers in the list of time sources in the *General Settings* tab. For all other remote computers, only querying for local time is allowed.

Enable *Open Port in Firewall* if `SuSEFirewall2` is active (which it is by default). If you leave the port closed, it is not possible to establish a connection to the time server.

18.2 Manually Configuring NTP in the Network

The easiest way to use a time server in the network is to set server parameters. For example, if a time server called `ntp.example.com` is reachable from the network, add its name to the file `/etc/ntp.conf` by adding the following line:

```
server ntp.example.com
```

To add more time servers, insert additional lines with the keyword `server`. After initializing `ntpd` with the command `systemctl start ntp`, it takes about one hour until 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 as soon as the computer is powered on. The correction is used immediately, resulting in a higher stability of the system time.

There are two possible ways to use the NTP mechanism as a client: First, the client can query the time from a known server in regular intervals. With many clients, this approach can cause a high load on the server. Second, the client can wait for NTP broadcasts sent out by broadcast time servers in the network. This approach has the disadvantage that the quality of the server is unknown and a server sending out wrong information can cause severe problems.

If the time is obtained via broadcast, you do not need the server name. In this case, enter the line `broadcastclient` in the configuration file `/etc/ntp.conf`. To use one or more known time servers exclusively, enter their names in the line starting with `servers`.

18.3 Dynamic Time Synchronization at Runtime

If the system boots without network connection, `ntpd` 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.

If you want `ntpd` to resolve DNS names at runtime, you must set the `dynamic` option. Then, when the network is established some time after booting, `ntpd` looks up the names again and can reach the time servers to get the time.

Manually edit `/etc/ntp.conf` and add `dynamic` to one or more `server` entries:

```
server ntp.example.com dynamic
```

Or use YaST and proceed as follows:

1. In YaST click *Network Services > NTP Configuration*.

2. Select the server you want to configure. Then click *Edit*.
3. Activate the *Options* field and add `dynamic`. Separate it with a space, if there are already other options entered.
4. Click *Ok* to close the edit dialog. Repeat the previous step to change all servers as wanted.
5. Finally click *Ok* to save the settings.

18.4 Setting Up a Local Reference Clock

The software package `ntpd` contains drivers for connecting local reference clocks. A list of supported clocks is available in the `ntp-doc` package in the file `/usr/share/doc/packages/ntp-doc/refclock.html`. Every driver is associated with a number. In NTP, the actual configuration takes place by means of pseudo IP addresses. The clocks are entered in the file `/etc/ntp.conf` as though they existed in the network. For this purpose, they are assigned special IP addresses in the form `127.127.t.u`. Here, `t` stands for the type of the clock and determines which driver is used and `u` for the unit, which determines the interface used.

Normally, the individual drivers have special parameters that describe configuration details. The file `/usr/share/doc/packages/ntp-doc/drivers/driverNN.html` (where `NN` is the number of the driver) provides information about the particular type of clock. For example, the “type 8” clock (radio clock over serial interface) requires an additional mode that specifies the clock more precisely. The Conrad DCF77 receiver module, for example, has mode 5. To use this clock as a preferred reference, specify the keyword `prefer`. The complete `server` line for a Conrad DCF77 receiver module would be:

```
server 127.127.8.0 mode 5 prefer
```

Other clocks follow the same pattern. Following the installation of the `ntp-doc` package, the documentation for NTP is available in the directory `/usr/share/doc/packages/ntp-doc`. The file `/usr/share/doc/packages/ntp-doc/refclock.html` provides links to the driver pages describing the driver parameters.

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 instance, 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.

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](#).

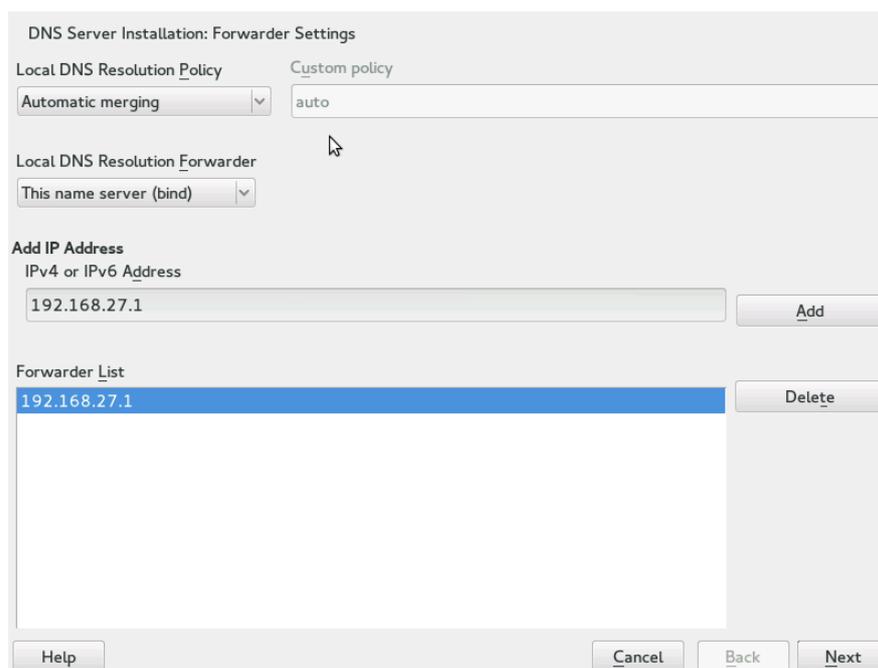


FIGURE 19.1: DNS SERVER INSTALLATION: FORWARDER SETTINGS

Forwarders are DNS servers to which your DNS server sends queries it cannot answer itself. Enter their IP address and click *Add*.

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”.

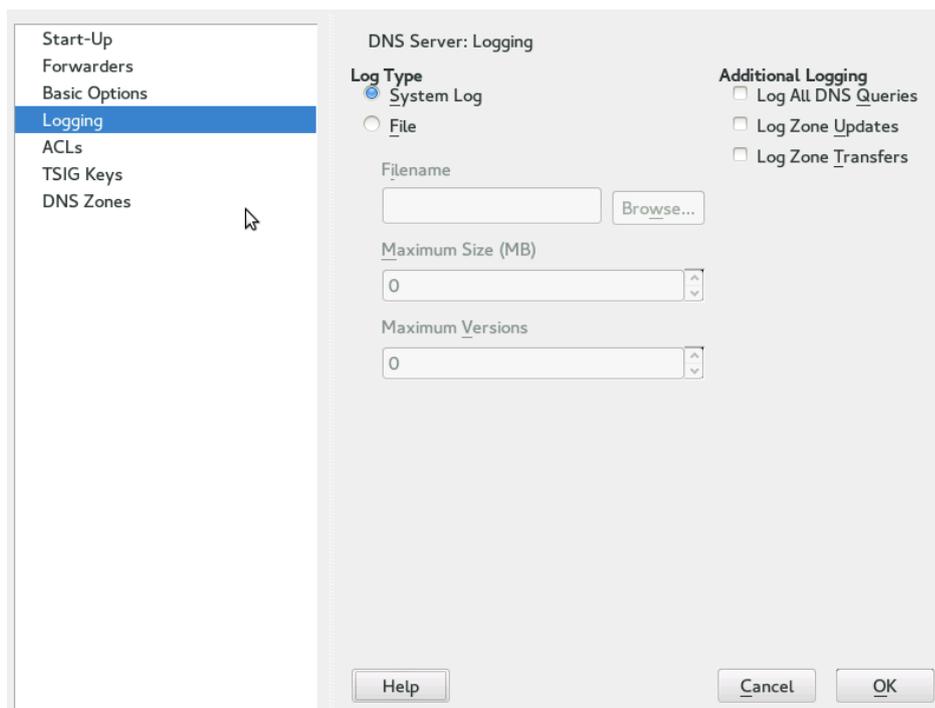


FIGURE 19.4: DNS SERVER: LOGGING

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* as well as 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.

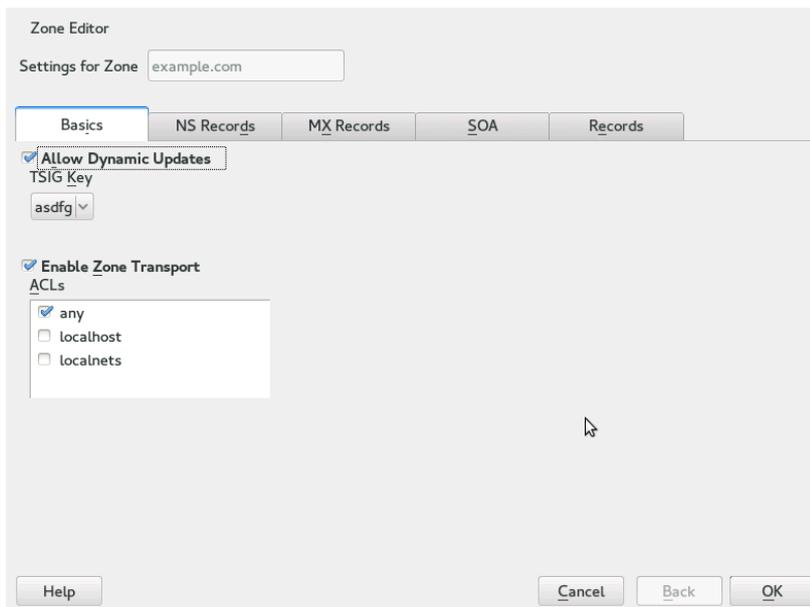


FIGURE 19.5: DNS SERVER: ZONE EDITOR (BASICS)

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)"*.

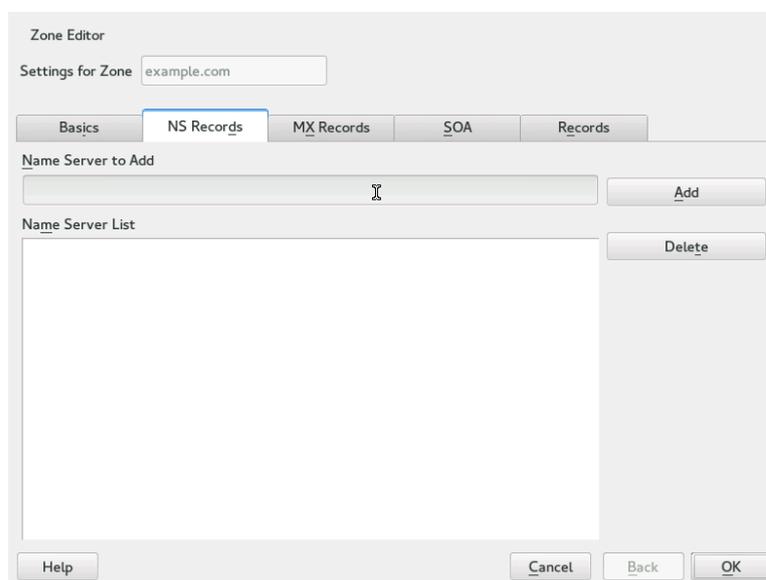


FIGURE 19.6: DNS SERVER: ZONE EDITOR (NS RECORDS)

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)”*.

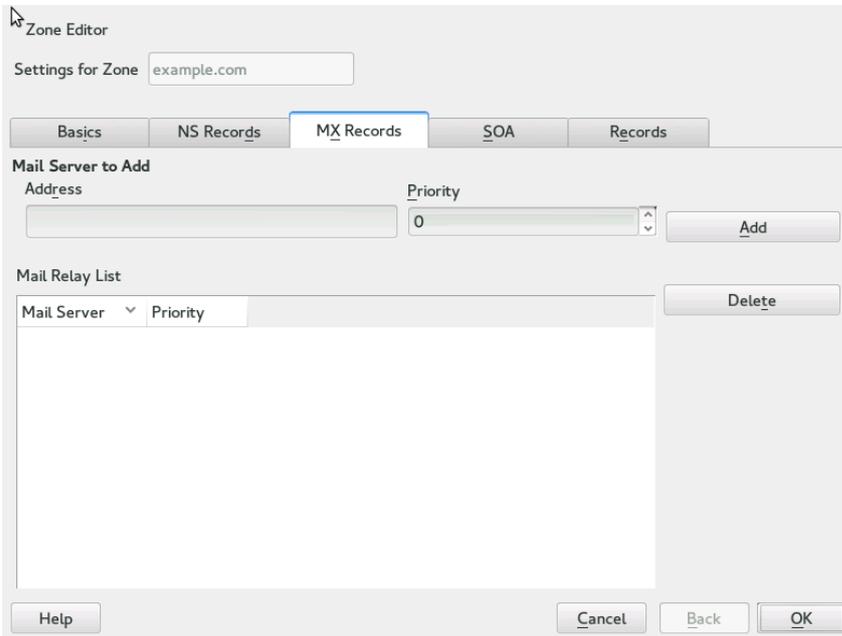


FIGURE 19.7: DNS SERVER: ZONE EDITOR (MX RECORDS)

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.

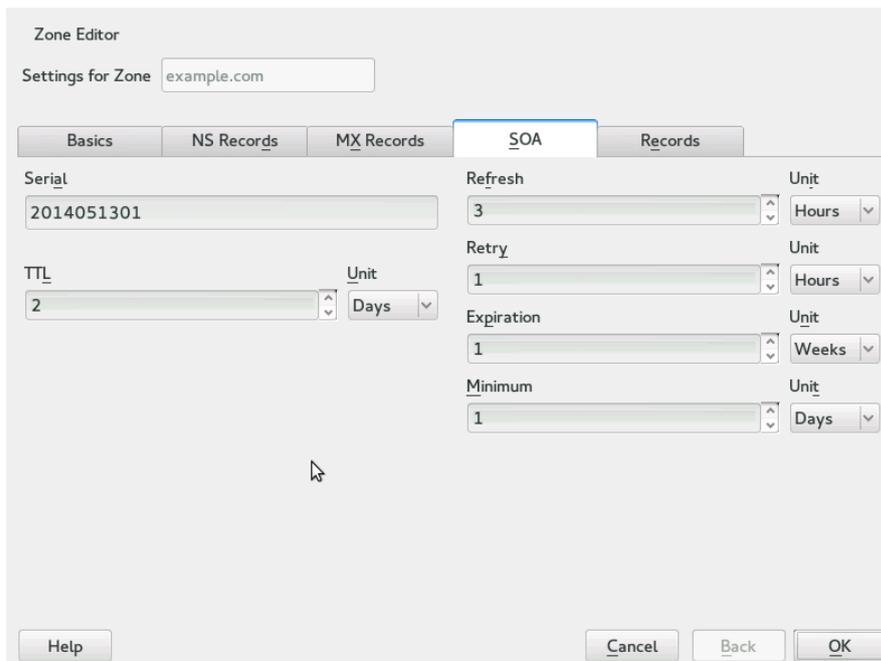


FIGURE 19.8: DNS SERVER: ZONE EDITOR (SOA)

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.

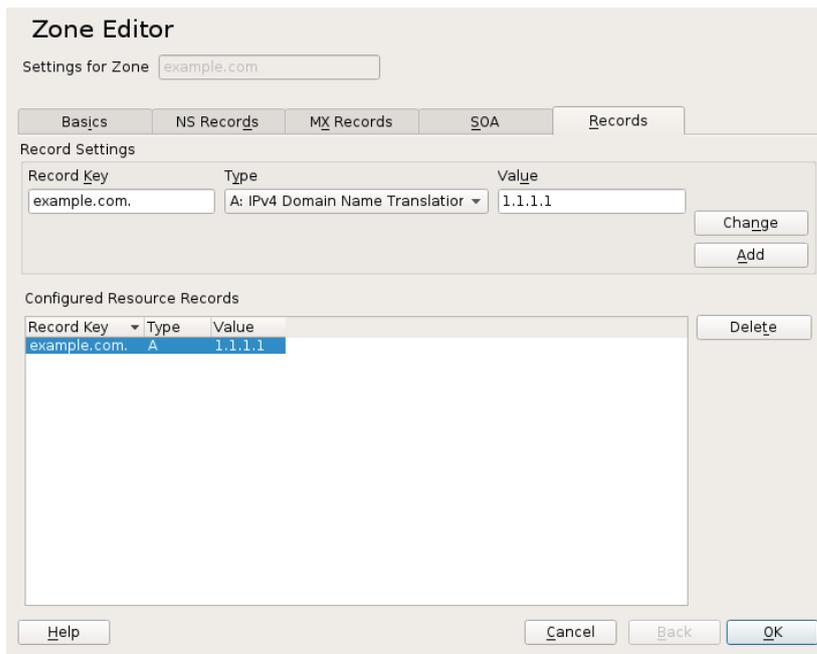


FIGURE 19.9: ADDING A RECORD FOR A MASTER ZONE

4. Back in the *DNS Zones* window, add a reverse master zone.

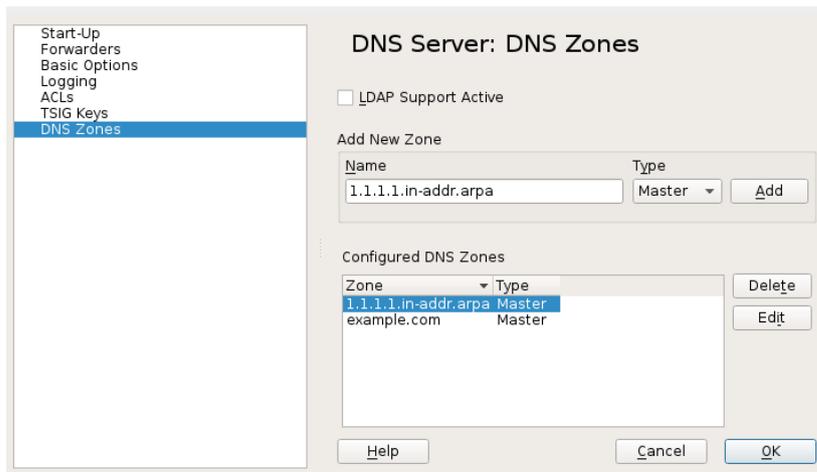


FIGURE 19.10: ADDING A REVERSE ZONE

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*.

Zone Editor

Settings for Zone

Basics NS Records SOA **Records**

Record Settings

Record Key: Type: PTR: Reverse Translat Value:

Configured Resource Records

Record Key	Type	Value
1.1.1.1.in-addr.arpa.	PTR	example.com.

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. If you already have a functioning Internet connection and have entered `127.0.0.1` as the name server address for `localhost` in `/etc/resolv.conf`, you normally 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, `/etc/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 at all. 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

```
1. $TTL 2D
2. example.com. IN SOA      dns root.example.com. (
3.                2003072441 ; serial
4.                1D        ; refresh
5.                2H        ; retry
6.                1W        ; expiry
7.                2D )      ; minimum
8.
9.                IN NS     dns
10.               IN MX     10 mail
11.
12. gate          IN A      192.168.5.1
```

```

13.          IN A      10.0.0.1
14. dns      IN A      192.168.1.116
15. mail     IN A      192.168.3.108
16. jupiter IN A      192.168.2.100
17. venus    IN A      192.168.2.101
18. saturn   IN A      192.168.2.102
19. mercury  IN A      192.168.2.103
20. ntp      IN CNAME  dns
21. dns6     IN A6     0      2002:c0a8:174::

```

Line 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).

Line 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.

Line 3:

The serial number is an arbitrary number that is increased 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.

Line 4:

The refresh rate specifies the time interval at which the secondary name servers verify the zone serial number. In this case, one day.

Line 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.

Line 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.

Line 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.

Line 9:

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.

Line 10:

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 and, if mail delivery to this server fails, an attempt is made with the next higher value.

Lines 12–19:

These 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
```

Line 20:

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

```
1. $TTL 2D
2. 168.192.in-addr.arpa. IN SOA dns.example.com. root.example.com. (
3. 2003072441 ; serial
4. 1D ; refresh
5. 2H ; retry
6. 1W ; expiry
7. 2D ) ; minimum
8.
9. IN NS dns.example.com.
10.
11. 1.5 IN PTR gate.example.com.
12. 100.3 IN PTR www.example.com.
13. 253.2 IN PTR cups.example.com.
```

Line 1:

`$TTL` defines the standard TTL that applies to all entries here.

Line 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.

Lines 3–7:

See the previous example for `example.com`.

Line 9:

Again 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.

Lines 11–13:

These 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 the help of 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`):

```
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 at all. 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 be fixed in most cases), 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 other 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. If you want to use LDAP, set up your LDAP environment before configuring the DHCP server.

For more information about LDAP, see *Book "Security Guide", Chapter 5 "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*. After this, select *Open Firewall for Selected Interfaces* to open the firewall for this interface, and click *Next*. See *Figure 20.1, "DHCP Server: Card Selection"*.

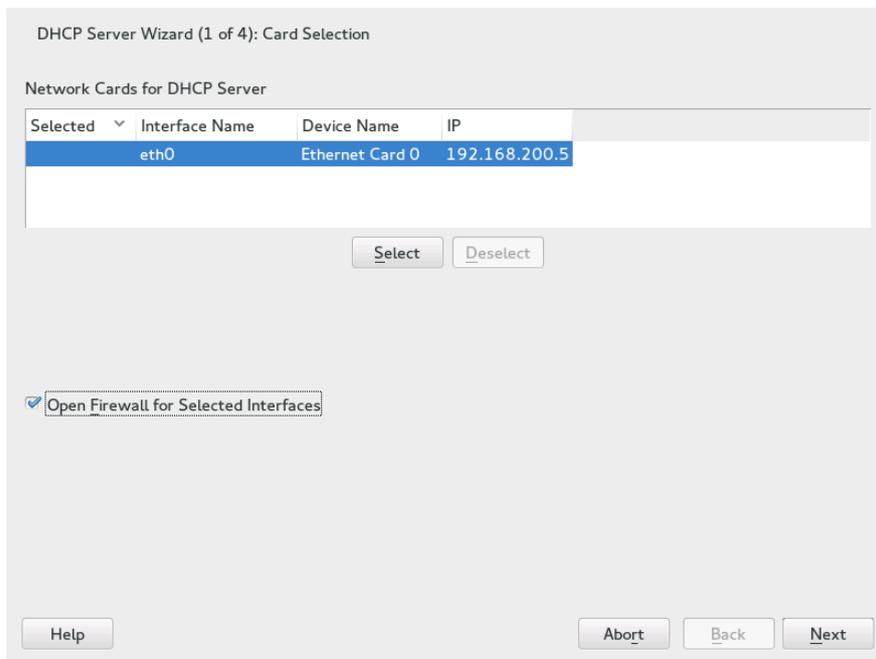


FIGURE 20.1: DHCP SERVER: CARD SELECTION

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: example.org

NTP Time Server: 192.168.200.10

Primary Name Server IP: 192.168.200.2

Print Server:

Secondary Name Server IP: 192.168.200.3

WINS Server:

Default Gateway (Router): 192.168.200.1

Default Lease Time: 1

Units: Hours

Buttons: Help, Abort, Back, Next

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="192.168.200.0"/>	<input type="text" value="255.255.255.0"/>	<input type="text" value="24"/>
Minimum IP Address	Maximum IP Address	
<input type="text" value="192.168.200.1"/>	<input type="text" value="192.168.200.254"/>	

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="1"/>	<input type="text" value="Hours"/>	<input type="text" value="2"/>	<input type="text" value="Days"/>

FIGURE 20.3: DHCP SERVER: DYNAMIC DHCP

- 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"](#).



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"](#).

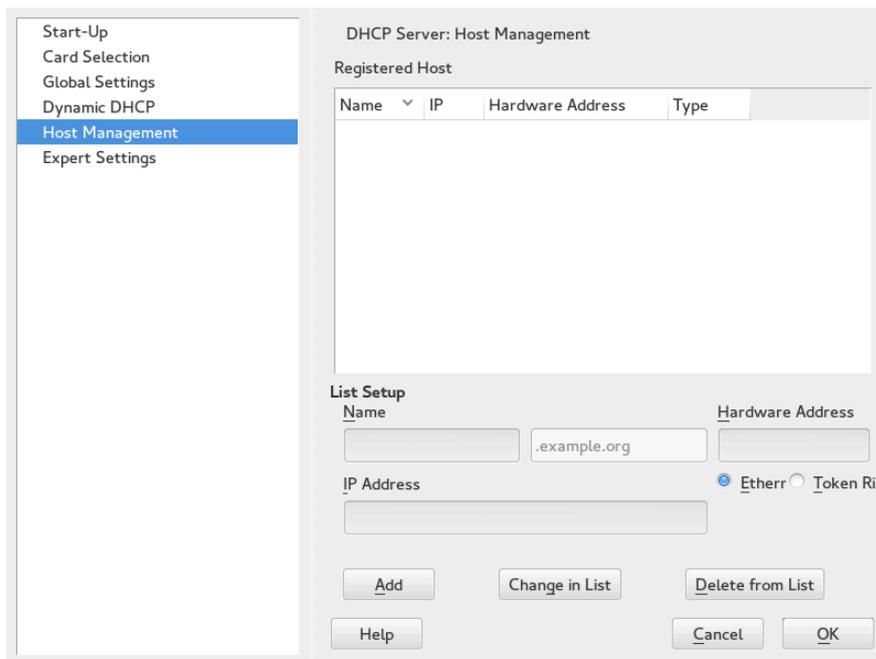


FIGURE 20.5: DHCP SERVER: HOST MANAGEMENT

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 behind bars in the chroot jail, which prevents him from touching 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.

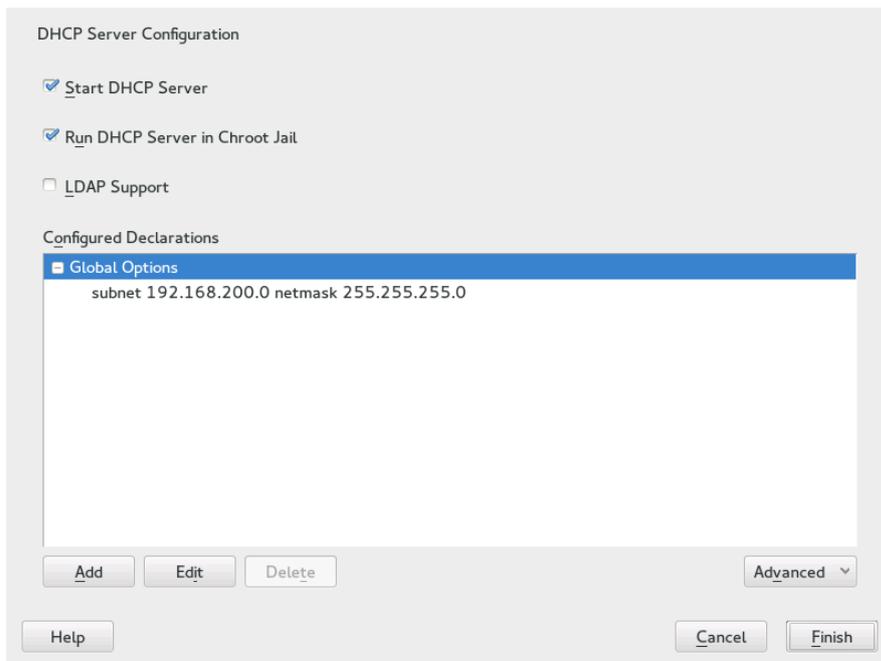


FIGURE 20.6: DHCP SERVER: CHROOT JAIL AND DECLARATIONS

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"](#)).

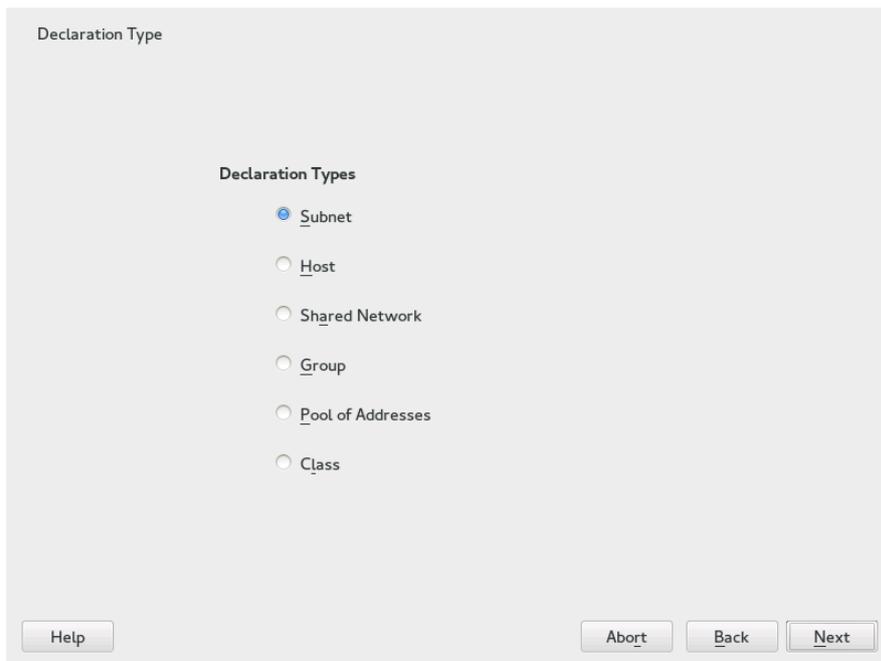


FIGURE 20.7: DHCP SERVER: SELECTING A DECLARATION TYPE

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*.

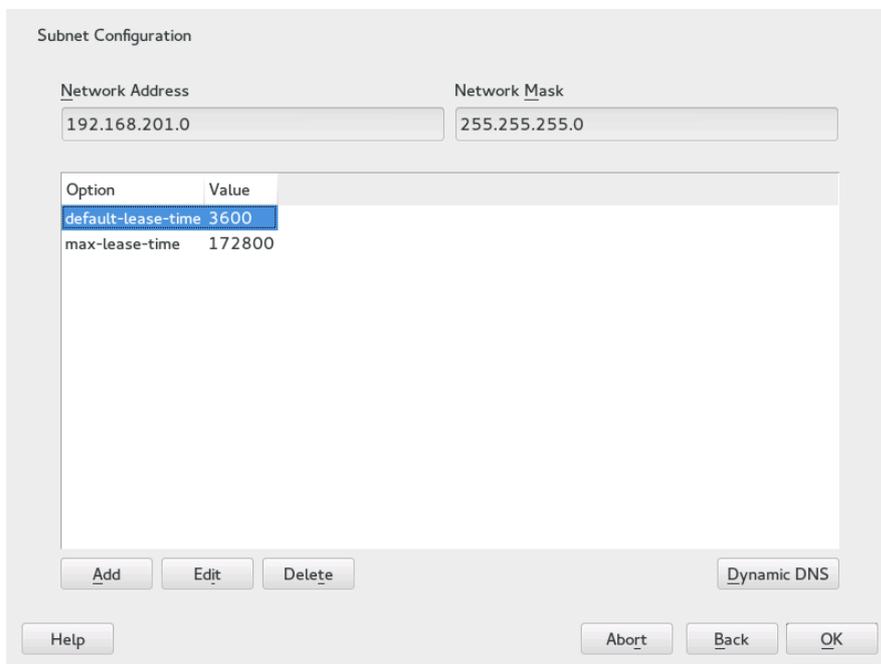


FIGURE 20.8: DHCP SERVER: CONFIGURING SUBNETS

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”](#)).

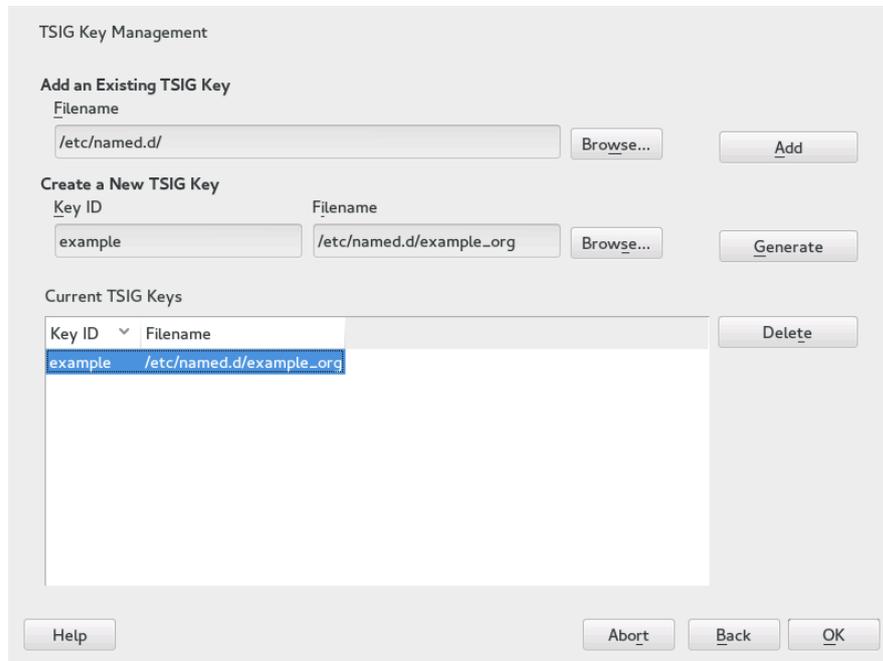


FIGURE 20.9: DHCP SERVER: TSIG CONFIGURATION

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.

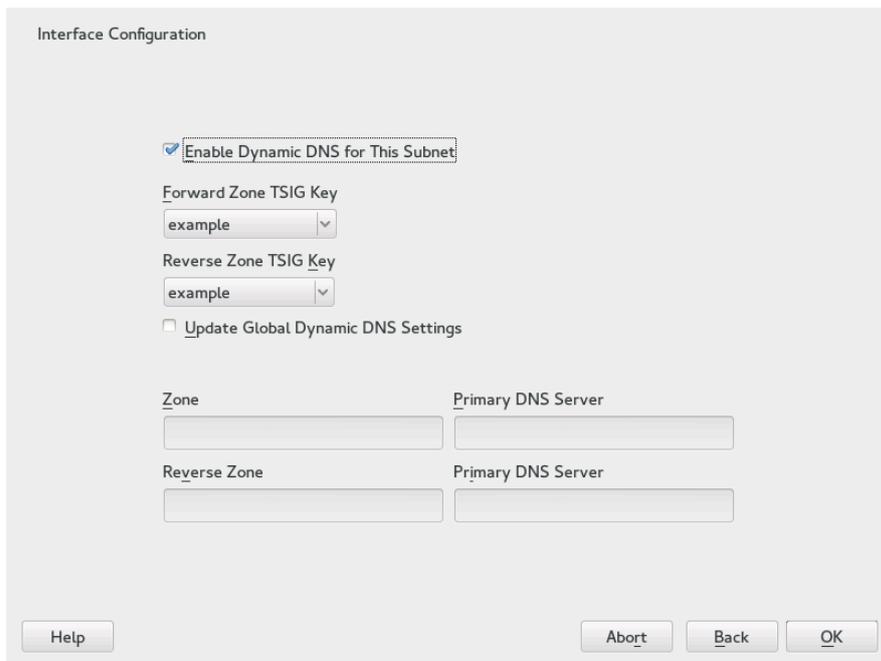


FIGURE 20.10: DHCP SERVER: INTERFACE CONFIGURATION FOR DYNAMIC DNS

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. To do so, select *Adapt Firewall Settings*. YaST then adjusts the rules of SuSEFirewall2 to the new conditions (see [Figure 20.11, “DHCP Server: Network Interface and Firewall”](#)), after which you can return to the original dialog by selecting *OK*.

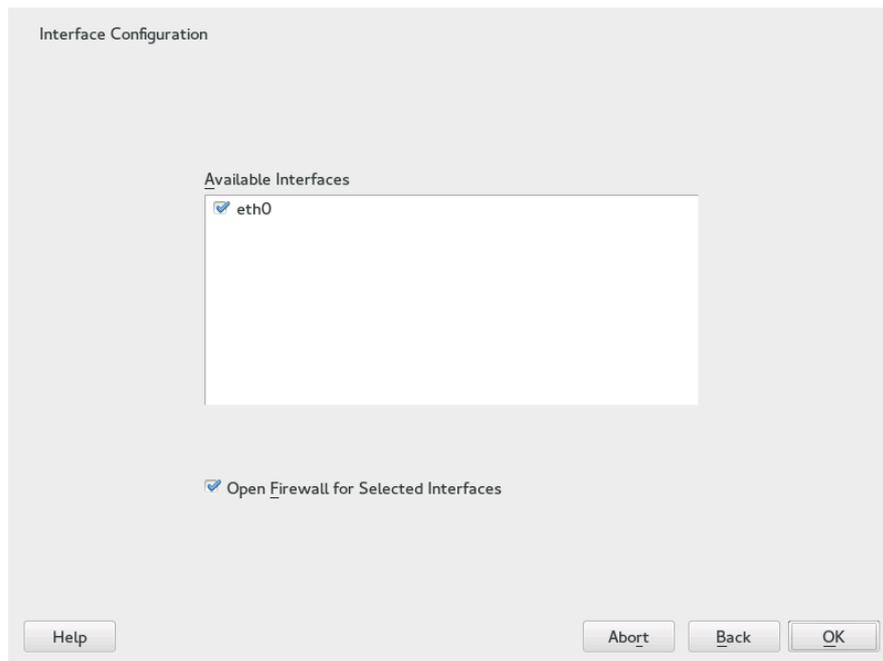


FIGURE 20.11: DHCP SERVER: NETWORK INTERFACE AND FIREWALL

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 SUSE Linux Enterprise Server. The DHCP server available is dhcpd (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 `rcd-hcpd 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 host name`, 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`
- `/etc/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* (<http://www.isc.org/products/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 Mac OS X, 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

CIFS (common Internet file system) protocol is another protocol supported by Samba. 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.

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. All common operating systems, such as Mac OS X, Windows, and OS/2, 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 to act as a domain controller at all. 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 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.

21.4.1.2.1 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*.

21.4.1.2.2 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*.

21.4.1.2.3 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*.

21.4.1.2.4 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*.

21.4.1.2.5 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 Guide”, Chapter 5 “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$]`

By means of this approach, details regarding the shares can be set differently or globally in the `[global]` section, which enhances the structural transparency of the configuration file.

21.4.2.1 The global Section

The following parameters of the `[global]` section need some adjustment 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 disturbances caused by a misconfigured Samba server. More in-

formation about this important 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 present 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 need 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 his own home directory, he 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 as soon as 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.

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 his or her 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 will be able to 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.

21.6 Samba as Login Server

In networks where predominantly Windows clients are found, it is often preferable that users may only register with a valid account and password. 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\$\n\nsmbpasswd -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" \n\n-s /bin/false %m\$\n\n
```

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 SUSE Linux Enterprise Server to Active Directory (AD).

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.

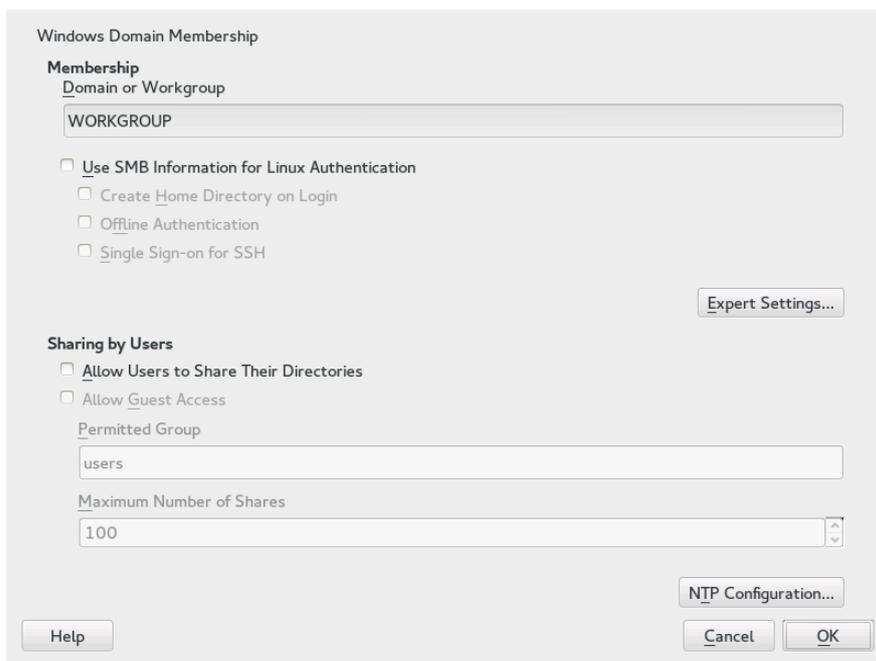


FIGURE 21.1: DETERMINING WINDOWS DOMAIN MEMBERSHIP

4. Check *Also Use SMB Information for Linux Authentication* to use the SMB source for Linux authentication on your SUSE Linux Enterprise 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-HOW-TO-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:

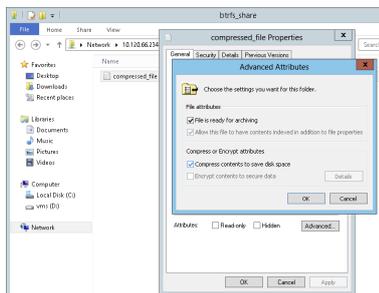


FIGURE 21.2: WINDOWS EXPLORER ADVANCED ATTRIBUTES DIALOG

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:

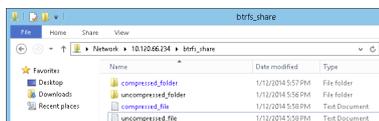


FIGURE 21.3: WINDOWS EXPLORER DIRECTORY LISTING WITH COMPRESSED FILES

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 thin-provisioned LVM volumes. The Samba suite supports managing of 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

```
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:

```
systemctl restart nmb smb
```

New Share

Identification

Share Name
Snapshotted Share

Share Description

Share Type

Printer

Directory

Share Path
/var/tmp

Read-Only

Inherit ACLs

Expose Snapshots

Utilize Btrfs Features

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.

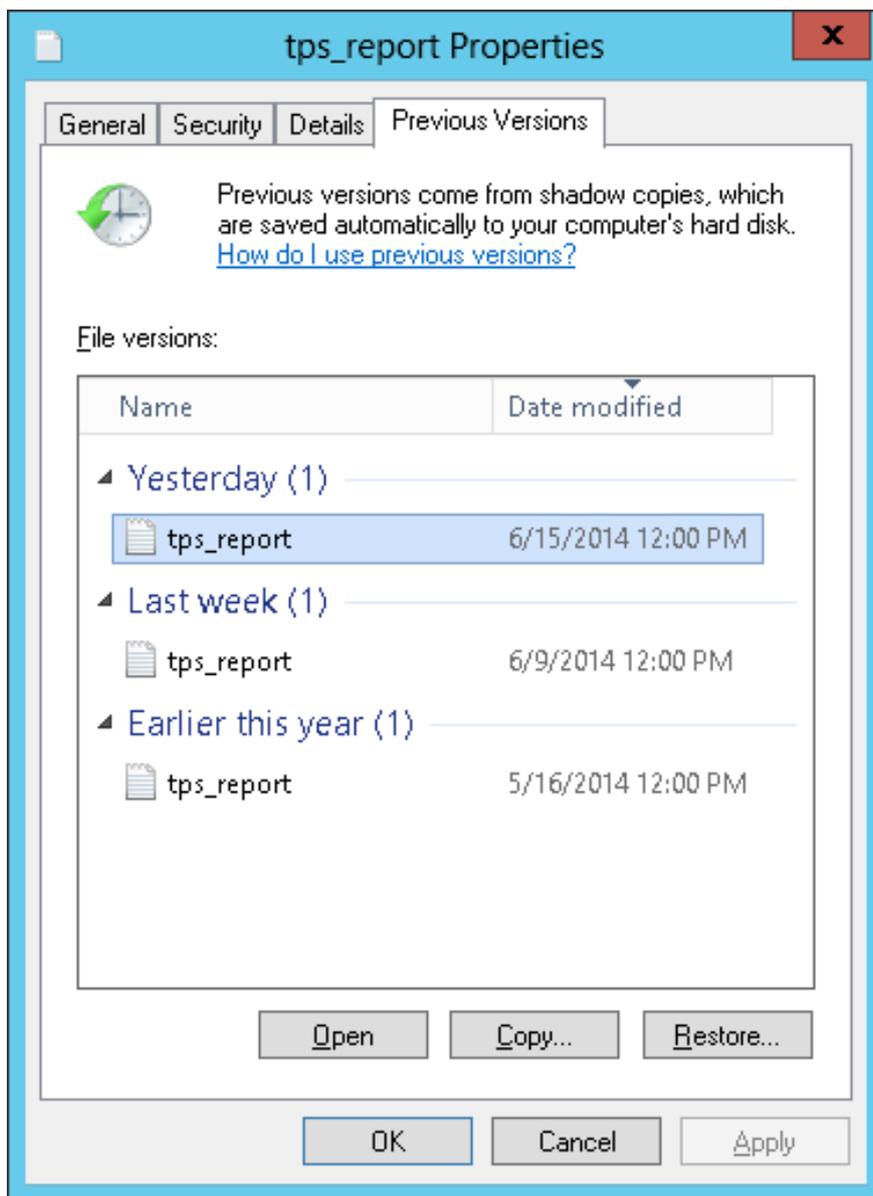


FIGURE 21.5: THE PREVIOUS VERSIONS TAB IN WINDOWS EXPLORER

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 SUSE Linux Enterprise Server 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:

```
# 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`:

```
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:

```
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:

```
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:

```
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:

```
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:

```
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 SUSE Linux Enterprise server.
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

Documentation for Samba ships with the `samba-doc` package which is not installed by default. Install it with `zypper install samba-doc`. Enter `apropos samba` at the command line to display some manual pages or browse the `/usr/share/doc/packages/samba` directory for more online documentation and examples. Find a commented example configuration (`smb.conf.SUSE`) in the `examples` subdirectory. Another file to look for Samba related information is `/usr/share/doc/packages/samba/README.SUSE`.

The Samba HOWTO (see <https://wiki.samba.org>) provided by the Samba team includes a section about troubleshooting. In addition to that, Part V of the document provides a step-by-step guide to checking your configuration.

22 Sharing File Systems with NFS

Distributing and sharing file systems over a network is a common task in corporate environments. The well-proven network file system (*NFS*) works with *NIS*, the yellow pages protocol. For a more secure protocol that works with *LDAP* and Kerberos, check *NFSv4*. Combined with pNFS, you can eliminate performance bottlenecks.

NFS with NIS makes a network transparent to the user. With NFS, it is possible to distribute arbitrary file systems over the network. With an appropriate setup, users always find themselves in the same environment regardless of the terminal they currently use.

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.

22.1 Terminology

The following are terms used in the YaST module.

Exports

A directory *exported* by an NFS server, which clients can integrate it into their system.

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` (user and group name mappings to IDs and vice versa), `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.

22.2 Installing NFS Server

The NFS server software is not part of the default installation. If you configure an NFS server as described in *Section 22.3, “Configuring NFS Server”* you will automatically be prompted to install the required packages. Alternatively, install the package `nfs-kernel-server` with YaST or Zypper. 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 SUSE Linux Enterprise systems, as is the case on all Enterprise-class Linux systems.

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.

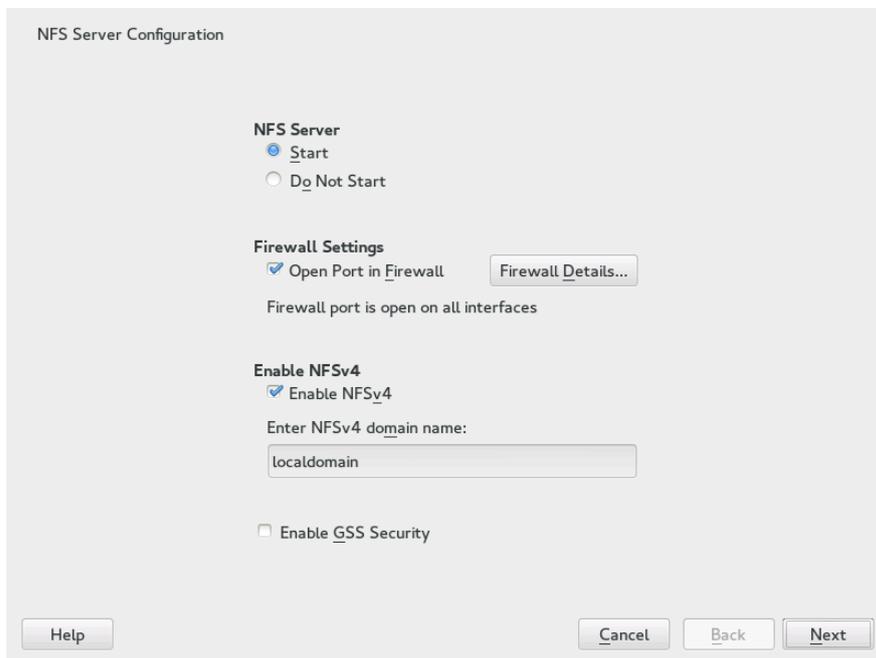


FIGURE 22.1: NFS SERVER CONFIGURATION TOOL

2. Activate the *Start* radio button.
3. If a firewall is active on your system (SuSEFirewall2), check *Open Ports in Firewall*. YaST adapts its configuration for the NFS server by enabling the `nfs` service.
4. Check whether you want to *Enable NFSv4*. If you deactivate NFSv4, YaST will only support NFSv3 and NFSv2.
 - If NFSv4 is selected, additionally enter the appropriate NFSv4 domain name.

Make sure the name is the same as the one in the `/etc/idmapd.conf` file of any NFSv4 client that accesses this particular server. This parameter is for the `idmapd` daemon that is required for NFSv4 support (on both server and client). Leave it as `localdomain` (the default) if you do not have any special requirements.

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. To start or restart the services, run the command `systemctl restart nfsserver`. This also starts the `rpc.idmapd` if NFSv4 is configured in `/etc/sysconfig/nfs`. The NFS server depends on a running RPC portmapper. Therefore, it also starts or restarts the portmapper service.



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 the previous SUSE Linux Enterprise Server 11 version, 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 **exports** (`man exports`).

`/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.



Tip: Mount Options

On SUSE Linux Enterprise prior to version 12, 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.

`/etc/idmapd.conf`

Every user on a Linux machine has a name and an ID. `idmapd` does the name-to-ID mapping for NFSv4 requests to the server and replies to the client. It must be running on both server and client for NFSv4, because NFSv4 uses only names for its communication.

Make sure that there is a uniform way in which user names and IDs (uid) 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
```

For more information, see the man pages of [idmapd](#) and [idmapd.conf](#) ([man idmapd](#) and [man idmapd.conf](#)).

After changing [/etc/exports](#) or [/etc/sysconfig/nfs](#), start or restart the NFS server service:

```
systemctl restart nfsserver
```

After changing [/etc/idmapd.conf](#), reload the configuration file:

```
killall -HUP rpc.idmapd
```

If the NFS service needs to start at boot time, run:

```
systemctl enable nfsserver
```

22.3.3 NFS with Kerberos

To use Kerberos authentication for NFS, GSS security 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. If you want 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 Guide", Chapter 7 "Network Authentication with Kerberos"*.
2. Start the `gssd` service on the client with **`systemctl start gssd`**.
3. Start the `svcgssd` service on the server with **`systemctl start svcgssd`**.

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`**:

```
mount host:remote-path local-path
```

To import user directories from the `nfs.example.com` machine, for example, use:

```
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:

```
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 large numbers of clients want to access data, an NFS server becomes a bottleneck and significantly impacts 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 “normal” 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.

SUSE Linux Enterprise 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 (v4.1)* 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:

```
mount -t nfs4 -o minorversion=1 MDS_SERVER MOUNTPOINT
```

To help with debugging, change the value in the `/proc` file system:

```
echo 32767 > /proc/sys/sunrpc/nfsd_debug  
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 \(http://nfs.sourceforge.net/\)](http://nfs.sourceforge.net/) ↗.
- For instructions for setting up kerberized NFS, refer to [NFS Version 4 Open Source Reference Implementation \(http://www.citi.umich.edu/projects/nfsv4/linux/krb5-setup.html\)](http://www.citi.umich.edu/projects/nfsv4/linux/krb5-setup.html) ↗.
- If you have questions on NFSv4, refer to the [Linux NFSv4 FAQ \(http://www.citi.umich.edu/projects/nfsv4/linux/faq/\)](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, **auto-mount** 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

```
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 Guide", Chapter 3 "Using NIS", Section 3.1 "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
-------------	---------	----------

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

```
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`.

```
sudo systemctl stop autofs
```

2. From one terminal, run `automount` manually in the foreground, producing verbose output.

```
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 makes use of 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`:

```
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:

```
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:

```
sudo systemctl restart autofs
```

```
# ls -l /nfs/export
total 20
drwxr-xr-x 6 1001 users 4096 Oct 25 08:56 ./
drwxr-xr-x 3 root root    0 Apr  1 09:47 ../
drwxr-xr-x 5 1001 users 4096 Jan 14 2013 .images/
drwxr-xr-x 10 1001 users 4096 Aug 16 2013 .profiled/
drwxr-xr-x 3 1001 users 4096 Aug 30 2013 .tmp/
drwxr-xr-x 4 1001 users 4096 Oct 25 08:56 SLE-12-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
```

```
systemctl restart autofs
```

For example, if you have a server named `jupiter` with an NFS share called `/export`, you can mount it by typing

```
# 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—then `autofs` has a handy solution for you.

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 the help of 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 the SuSE-Firewall2 to allow the service *HTTP Server* in the external zone. This can be done using YaST. See *Book "Security Guide", Chapter 15 "Masquerading and Firewalls", Section 15.4.1 "Configuring the Firewall with YaST"* 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:

```
root # 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 if you want 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 `httpd2 -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 IPs 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 IPs `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 present 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

The `Require all granted` statement used to be expressed as

```
Order allow,deny
Allow from all
```

in previous versions of Apache. 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 referred to as 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*.

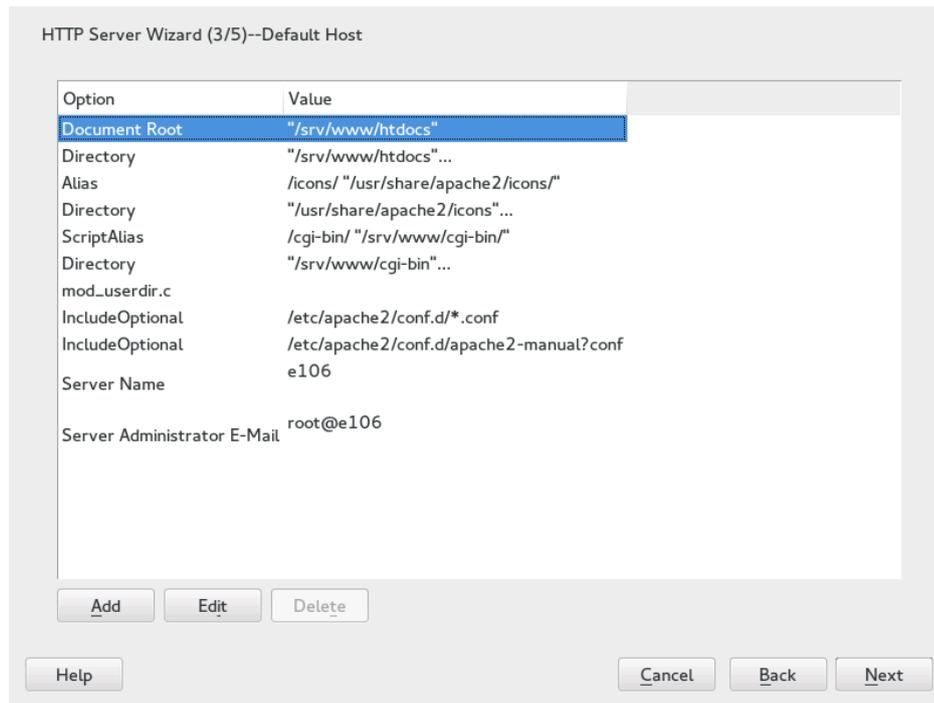


FIGURE 24.1: HTTP SERVER WIZARD: DEFAULT HOST

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

With the help of 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) if you want 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. If you want 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”](#).

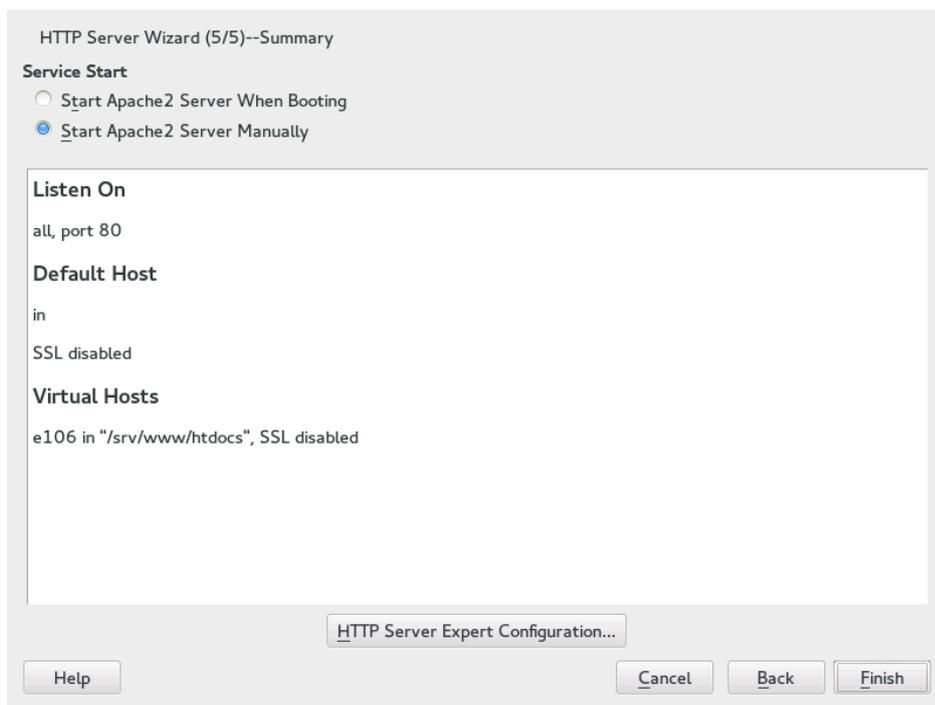


FIGURE 24.2: HTTP SERVER WIZARD: SUMMARY

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.

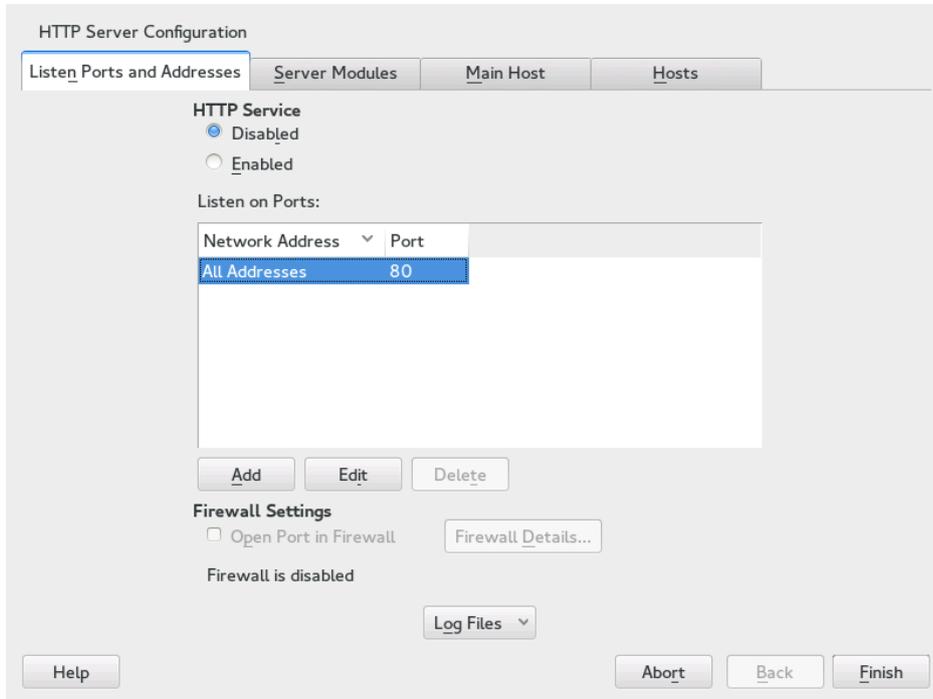


FIGURE 24.3: HTTP SERVER CONFIGURATION: LISTEN PORTS AND ADDRESSES

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”](#).

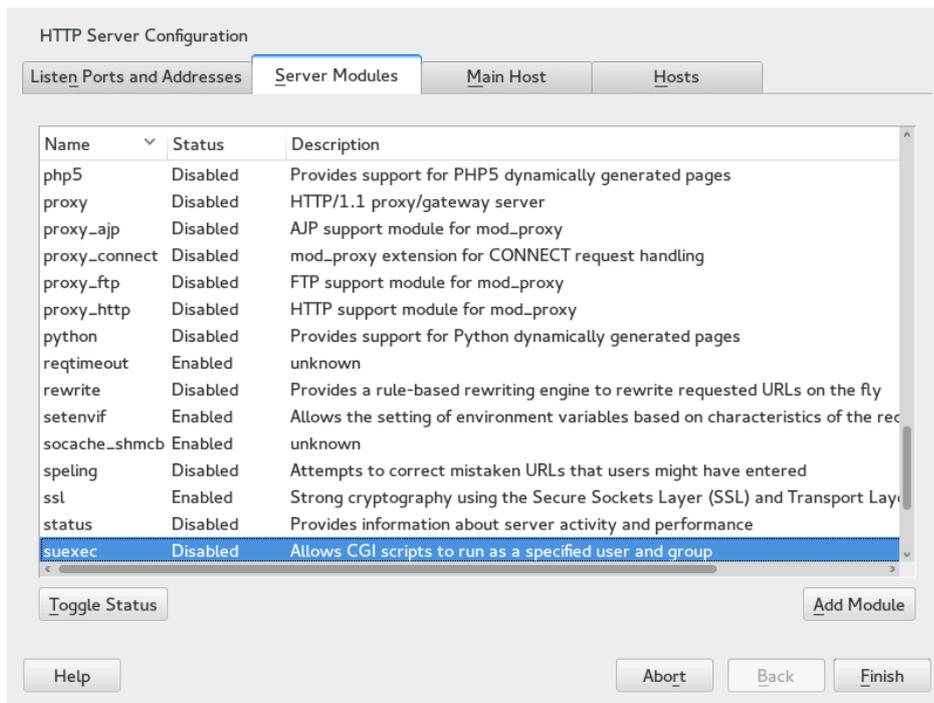


FIGURE 24.4: HTTP SERVER CONFIGURATION: SERVER MODULES

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.

apachectl -k graceful

Starts a second Web server that immediately serves all incoming requests. The previous instance of the Web server continues to handle all existing requests for a defined period of time configured with GracefulShutdownTimeout.

This command is useful either when upgrading to a new version or when having changed configuration options that require a restart. Using this option ensures a minimum server downtime.

If GracefulShutdownTimeout is set to zero, the server will wait indefinitely until all remaining requests have been fully served.

A graceful restart can fail if the original Apache instance is not able to clear all necessary resources. In this case, the command will result in a graceful stop.

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).



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 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 (PHP5, Perl, 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 mod_foo` or `a2dismod mod_foo`, 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 not loaded by default. If you need the same modules on each virtual host, you can include `*.conf` from this directory. Otherwise include individual files. 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_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_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 makes certain 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](#)
- [Section 24.4.4.2, “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` and `mod_perl`.

Package Name: `apache2-mod_apparmor`

More Information: *Book “Security Guide”*

mod_perl

mod_perl enables you to run Perl scripts in an embedded interpreter. The persistent interpreter embedded in the server avoids the overhead of starting an external interpreter and the penalty of Perl start-up time.

Package Name: apache2-mod_perl

Configuration File: /etc/apache2/conf.d/mod_perl.conf

More Information: /usr/share/doc/packages/apache2-mod_perl

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/lib/apache2`.
- `/usr/sbin/apxs2-prefork`—suitable for prefork MPM modules. The installation location is `/usr/lib/apache2-prefork`.
- `/usr/sbin/apxs2-worker`—suitable for worker MPM modules. The installation location is `/usr/lib/apache2-worker`.

Install and activate a module from source code with the following commands:

```
cd /path/to/module/source; apxs2 -cia
    mod_foo.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 Getting CGI Scripts to Work

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 Perl or 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`.

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

- 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`. Its 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 SSL/TLS, 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 SSL/TSL 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 SSL/TSL 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 SSL/TSL, 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 the visitors to such a site will see the annoying "this is an untrusted site" warning because Web browsers do not know the self-signed certificate.

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.

```
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 (e.g. 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).

```
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.

```
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.

```
sudo cp new.cert.cert /etc/apache2/ssl.crt/server.crt  
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.

The best-known official CAs are Thawte (<http://www.thawte.com/>) or Verisign (<http://www.verisign.com>). These and other CAs are already compiled into all browsers, so certificates signed by these certificate authorities are automatically accepted by the browser.

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:

```
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 SSL and TLS requests on the Web server side is 443. There is no conflict between a “regular” Apache listening on port 80 and an SSL/TLS-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 YaST as described in *Book “Security Guide”, Chapter 15 “Masquerading and Firewalls”, Section 15.4.1 “Configuring the Firewall with YaST”*.

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`. 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. Although SNI is supported by most browsers, some browsers for mobile hardware as well as Internet Explorer and Safari on Windows* XP lack SNI support. See http://en.wikipedia.org/wiki/Server_Name_Indication for details.

Configure how to handle non-SNI capable browser with 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 12 SP1, 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:

```
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:

```
systemctl start apache2@instance_name
```

For example:

```
systemctl start apache2@example_web.org
```

This instance tries to read its configuration from `/etc/sysconfig/apache2@example_web.org`. If the file is not present or it does not set `APACHE_HTTPD_CONF`, the instance reads `/etc/apache2@example_web.org/httpd.conf`.

An example to set up an additional instance of Apache follows. Note that you need to execute all the commands as `root`.

PROCEDURE 24.4: CONFIGURING ADDITIONAL APACHE INSTANCE

1. Create a new configuration file based on `/etc/sysconfig/apache2`, for example `/etc/sysconfig/apache2@example_web.org`:

```
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`.

```
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`.

```
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
```

If you want 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:

```
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:

```
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 any variable that `/etc/sysconfig/apache2` can, 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

```
export HTTPD_INSTANCE=example_web.org
a2enmod access_compat
a2enmod status
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 as soon as possible. The SUSE security announcements are available from the following locations:

- **Web Page.** <http://www.suse.com/support/security/>
- **Mailing List Archive.** <http://lists.opensuse.org/opensuse-security-announce/>
- **List of Security Announcements.** <http://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 Perl, 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/httpd2`, rather use the `systemctl` commands instead (described in [Section 24.3, “Starting and Stopping Apache”](#)). It 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 the help of any 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_auth_kerb

<http://modauthkerb.sourceforge.net/>

mod_perl

<http://perl.apache.org/>

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/> ↗

Writing Apache Modules with Perl and C

<http://www.modperl.com/> ↗

24.10.4 Miscellaneous Sources

If you experience difficulties specific to Apache in openSUSE Leap, take a look at the Technical Information Search at <http://www.suse.com/support> ↗. The history of Apache is provided at http://httpd.apache.org/ABOUT_APACHE.html ↗. This page also explains why the server is called Apache.

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.

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 various 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. Press *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 xinetd*.

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*.

The *Selected Service* frame of the *FTP Start-Up* dialog shows which FTP server is used: either *vsftpd* or *pure-ftpd*. If both servers are installed, you can switch between them—the current configuration will automatically be converted.

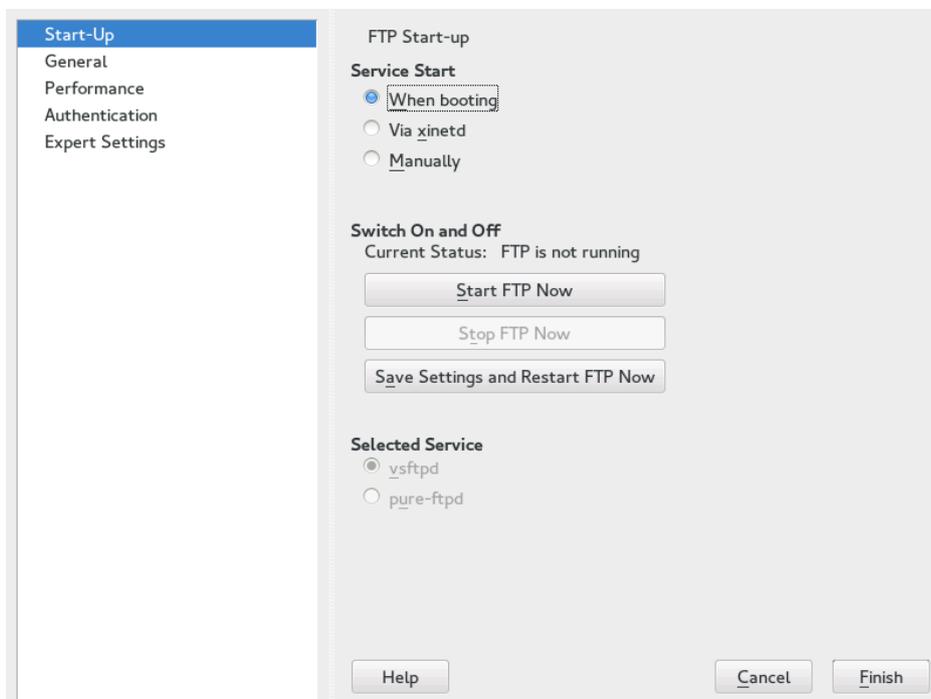


FIGURE 25.1: FTP SERVER CONFIGURATION — START-UP

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.



Note: Write Permissions in FTP Directory

The `pure-ftpd` server allows the FTP directory for anonymous users to be writable. When switching between servers, make sure you remove the write permissions in the directory that was used with `pure-ftpd` before switching back to the `vsftpd` server.

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.

If you want to allow users to upload files to the FTP server, check *Enable Upload* in the *Uploading* frame of the *Authentication* dialog. Here you are able to allow uploading or creating directories even for anonymous users by checking the respective box.



Note: vsftpd—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, uncheck *Enable Passive Mode* option in *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 The Proxy Server Squid

Squid is a widely-used proxy cache 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 may be set up in multiple hierarchies to assure optimal response times and low bandwidth usage, even in modes that are transparent for the end user. Additional software like squid-Guard may be used to filter Web contents.

Squid acts as a proxy cache. 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. One of the advantages 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 the actual caching, Squid offers a wide range of features such as distributing the load over intercommunicating hierarchies of proxy servers, defining strict access control lists for all clients accessing the proxy, allowing or denying access to specific Web pages with the help of other applications, and generating statistics about frequently-visited Web pages for the assessment of the users' surfing habits. Squid is not a generic proxy. 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 Real Audio, news, or video conferencing. Because Squid only supports the UDP protocol to provide communication between different caches, many other multimedia programs are not supported.

26.1 Some Facts about Proxy Caches

As a proxy cache, 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 using a proxy cache. The firewall denies all clients access to external services except Squid. All Web connections must be established by the proxy. With this configuration, Squid completely controls Web access.

If the firewall configuration includes a DMZ, the proxy should operate within this zone. [Section 26.5, “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.

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 finding an object already existing in 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 get 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, which in turn is connected to the proxy cache 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 get the objects, one cache sends an ICP request to all sibling proxies. These answer the requests via ICP responses with a HIT code if the object was detected or a MISS if it was not. 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.



Tip: Avoiding 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 one.

26.1.3 Caching Internet Objects

Not all objects available in the network are static. There are a lot of dynamically generated CGI pages, visitor counters, and encrypted SSL content documents. Objects like this are not cached because they change each time they are accessed.

The question remains as to how long all the other objects stored in the cache should stay there. To determine this, all objects in the cache are assigned one of various possible 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 are used as well.

Objects in the cache are normally replaced, because of a lack of free hard disk space, using algorithms such as LRU (last recently used). This means that the proxy expunges the objects that have not been requested for the longest time.

26.2 System Requirements

The most important thing is to determine the maximum network load the system must bear. Therefore, it is important to pay more attention to the load peaks, because these might be more than four times the day's average. When in doubt, it would be better to overestimate the system's requirements, because having Squid working close to the limit of its capabilities could lead to a severe loss in the quality of the service. The following sections point to the system factors in order of significance.

26.2.1 Hard Disks

Speed plays an important role in the caching process, so this factor deserves special attention. For hard disks, this parameter is described as *random seek time*, measured in milliseconds. Because the data blocks that Squid reads from or writes to the hard disk tend to be rather small, the seek time of the hard disk is more important than its data throughput. For the purposes of a proxy, hard disks with high rotation speeds are probably the better choice, because they allow the read-write head to be positioned in the required spot more quickly. One possibility to speed up the system is to use a number of disks concurrently or to employ striping RAID arrays.

26.2.2 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 the less requested objects are replaced by newer ones. If, for example, one GB is available for the cache and the users only surf ten MB per day, it would take more than one hundred days to fill the cache.

The easiest way to determine the needed cache size is to consider the maximum transfer rate of the connection. With a 1 Mbit/s connection, the maximum transfer rate is 125 KB/s. If all this traffic ends up in the cache, in one hour it would add up to 450 MB and, 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. This is why 2 GB of disk space is required in the example for Squid to keep one day's worth of browsed data cached.

26.2.3 RAM

The amount of memory (RAM) required by Squid directly correlates to the number of objects in the cache. Squid also stores cache object references and frequently requested objects in the main memory to speed up retrieval of this data. Random access memory is much faster than a hard disk.

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.

It is very important to have sufficient memory for the Squid process, because system performance is dramatically reduced if it must be swapped to disk. The `cachemgr.cgi` tool can be used for the cache memory management. This tool is introduced in [Section 26.6, “cachemgr.cgi”](#).

26.2.4 CPU

Squid is not a program that requires intensive CPU usage. The load of the processor is only increased while the contents of the cache are loaded or checked. Using a multiprocessor machine does not increase the performance of the system. To increase efficiency, it is better to buy faster disks or add more memory.

26.3 Starting Squid

If not already installed, install the `squid` package. `squid` does not belong to the default openSUSE® Leap installation scope.

Squid is already preconfigured in SUSE® Linux Enterprise Server, you can start it right 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 entered, because Squid does not start if it does not detect a DNS server in `/etc/resolv.conf`.

26.3.1 Commands for Starting and Stopping Squid

To start Squid, enter `systemctl start squid` at the command line as `root`. In the initial start-up, the directory structure of the cache must first be defined in `/var/cache/squid`. This is done automatically by the start script and can take a few seconds or even minutes. If `done` appears to the right in green, Squid has been successfully loaded. To test the functionality of Squid on the local system, enter `localhost` as the proxy and `3128` as the port in the browser.

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 that control access to the proxy. More information about this is available in [Section 26.4.2, “Options for Access Controls”](#).

After modifying the configuration file `/etc/squid/squid.conf`, Squid must reload the configuration file. Do this with `systemctl reload squid`. Alternatively, completely restart Squid with `systemctl restart squid`.

The command `systemctl status squid` can be used to check if the proxy is running. The command `systemctl stop squid` causes Squid to shut down. This can take a while, because Squid waits up to half a minute (`shutdown_lifetime` option in `/etc/squid/squid.conf`) before dropping the connections to the clients and writing its data to the disk.



Warning: Terminating Squid

Terminating Squid with `kill` or `killall` can damage the cache. To be able to restart Squid, a damaged cache must be deleted.

If Squid dies after a short period of time even though it was started successfully, check whether there is a faulty name server entry or whether the `/etc/resolv.conf` file is missing. Squid logs the cause of a start-up failure in the file `/var/log/squid/cache.log`. If Squid should be loaded automatically when the system boots, enable the service with `systemctl enable squid`.

An uninstall of Squid does not remove the cache hierarchy or the log files. To remove these, delete the `/var/cache/squid` directory manually.

26.3.2 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 `/etc/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 enter the local DNS server in the `/etc/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, enter it in the configuration file `/etc/named.conf` under `forwarders` along with its IP address. With dynamic DNS, this can be achieved automatically during connection establishment 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. You must, however, enter the local DNS server in the file `/etc/resolv.conf` as described above. Additionally, the providers static name server must be entered 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 `/etc/squid/squid.conf` 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. Nearly all entries begin with `#` (the lines are commented) and the relevant specifications can be found at the end of the line. The given values almost always correlate with the default values, so removing the comment signs without changing any of the parameters usually has little effect. If possible, leave the sample as it is and insert the options along with the modified parameters 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. If you try to use the old `squid.conf`, you risk that the configuration no longer works, because options are sometimes modified and new changes added.

26.4.1 General Configuration Options (Selection)

`http_port 3128`

This is the port on which Squid listens for client requests. The default port is `3128`, but `8080` is also common. If desired, specify several port numbers separated by blank spaces.

`cache_peer hostname type proxy-port icp-port`

Here, enter a parent proxy, for example, if you want to use the proxy of your ISP. As `hostname`, enter the name or IP address of the proxy to use and, as `type`, enter `parent`. For `proxy-port`, enter the port number that is also given by the operator of the parent for use in the browser (usually `8080`). Set the `icp-port` to `7` or `0` if the ICP port of the parent is not known and its use is irrelevant to the provider. In addition, `default` and `no-query` may be specified after the port numbers to prohibit the use of the ICP protocol. Squid then behaves like a normal browser as far as the provider's proxy is concerned.

`cache_mem 8 MB`

This entry 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 ufs /var/cache/squid/ 100 16 256`

The entry `cache_dir` defines the directory where all the objects are stored on disk. The numbers at the end indicate the maximum disk space in MB to use and the number of directories in the first and second level. The `ufs` parameter should be left alone. The default is 100 MB occupied disk space in the `/var/cache/squid` directory and creation of 16 subdirectories inside it, each containing 256 more subdirectories. When specifying the disk space to use, leave sufficient reserve disk space. Values from a minimum of 50% to a maximum of 80% of the available disk space make the most sense here. The last two numbers for the directories should only be increased with caution, because too many directories can also lead to performance problems. If you have several disks that share the cache, enter several `cache_dir` lines.

`cache_access_log /var/log/squid/access.log,`

`cache_log /var/log/squid/cache.log,`

`cache_store_log /var/log/squid/store.log`

These three entries specify the paths where Squid log files all its actions. Normally, nothing is changed here. If Squid is experiencing a heavy usage burden, it might make sense to distribute the cache and the log files over several disks.

`emulate_httpd_log off`

If the entry is set to *on*, obtain readable log files. Some evaluation programs cannot interpret this, however.

`client_netmask 255.255.255.255`

With this entry, mask IP addresses of clients in the log files. The last digit of the IP address is set to zero if you enter 255.255.255.0 here. You may protect the privacy of your clients this way.

`ftp_user Squid@`

With this, set the password Squid should use for the anonymous FTP login. It can make sense to specify a valid e-mail address here, because some FTP servers check these for validity.

`cache_mgr webmaster`

An e-mail address to which Squid sends a message if it unexpectedly crashes. The default is *webmaster*.

`logfile_rotate 0`

If you run `squid -k rotate`, **Squid** can rotate secured log files. The files are numbered in this process and, after reaching the specified value, the oldest file is overwritten. The default value is 0 because archiving and deleting log files in openSUSE Leap is carried out by a cron job set in the configuration file /etc/logrotate/squid.

`append_domain <domain>`

With *append_domain*, specify which domain to append automatically when none is given. Usually, your own domain is entered here, so entering *www* in the browser accesses your own Web server.

`forwarded_for on`

If you set the entry to *off*, Squid removes the IP address and the system name of the client from HTTP requests. Otherwise it adds a line to the header like

```
X-Forwarded-For: 192.168.0.1
```

negative_ttl 5 minutes; negative_dns_ttl 5 minutes

Normally, you do not need to change these values. If you have a dial-up connection, however, the Internet may, at times, not be accessible. Squid makes a note of the failed requests then refuses to issue new ones, although the Internet connection has been reestablished. In a case such as this, change the *minutes* to *seconds*. Then, after clicking *Reload* in the browser, the dial-up process should be reengaged after a few seconds.

never_direct allow acl_name

To prevent Squid from taking requests directly from the Internet, use the above command to force connection to another proxy. This must have previously been entered in *cache_peer*. If all is specified as the acl_name, force all requests to be forwarded directly to the *parent*. This might be necessary, for example, if you are using a provider that strictly stipulates the use of its proxies or denies its firewall direct Internet access.

26.4.2 Options for Access Controls

Squid provides a detailed system for controlling the access to the proxy. By implementing ACLs, it can be configured easily and comprehensively. This involves 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 with *http_access* rules.

acl <acl_name> <type> <data>

An ACL requires at least three specifications to define it. 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 and can also be read from a file, for example, via host names, IP addresses, or URLs. The following are some simple examples:

```
acl mysurfers srcdomain .my-domain.com
acl teachers src 192.168.1.0/255.255.255.0
acl students src 192.168.7.0-192.168.9.0/255.255.255.0
acl lunch time MTWHF 12:00-15:00
```

`http_access allow <acl_name>`

`http_access` defines who is allowed to use the proxy and who can access what on the Internet. For this, ACLs must be given. `localhost` and `all` have already been defined above, which 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, and, 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, the `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 gets access Monday to Friday during lunch time.

```
http_access deny localhost
http_access allow teachers
http_access allow students lunch time
http_access deny all
```

The list with the `http_access` entries should only be entered, for the sake of readability, at the designated position in the `/etc/squid/squid.conf` file. That is, between the text

```
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR
# CLIENTS
```

and the last

```
http_access deny all
```

`redirect_program /usr/bin/squidGuard`

With this option, specify a redirector such as `squidGuard`, which allows the blocking of unwanted URLs. Internet access can be individually controlled for various user groups with the help of proxy authentication and the appropriate ACLs. `squidGuard` is a separate package that can be installed and configured.

`auth_param basic program /usr/sbin/pam_auth`

If users must be authenticated on the proxy, set a corresponding program, such as `pam_auth`. When accessing `pam_auth` for the first time, the user sees a login window in which to enter the user name and password. In addition, an ACL is still required, so only clients with a valid login can use the Internet:

```
acl password proxy_auth REQUIRED
```

```
http_access allow password
http_access deny all
```

The *REQUIRED* after *proxy_auth* can be replaced with a list of permitted user names or with the path to such a list.

`ident_lookup_access allow <acl_name>`

With this, have an ident request run for all ACL-defined clients to find each user's identity. If you apply *all* to the *<acl_name>*, this is valid for all clients. Also, an ident daemon must be running on all clients. For Linux, install the `pidentd` package for this purpose. For Microsoft Windows, free software is available for download from the Internet. To ensure that only clients with a successful ident lookup are permitted, define a corresponding ACL here:

```
acl identhosts ident REQUIRED

http_access allow identhosts
http_access deny all
```

Here, too, replace *REQUIRED* with a list of permitted user names. Using *ident* can slow down the access time quite a bit, because ident lookups are repeated for each request.

26.5 Configuring a Transparent Proxy

The usual way of working with proxy servers is the following: the Web browser sends requests to a certain port in the proxy server and the proxy provides these required objects, whether they are in its cache or not. When working in a network, several situations may arise:

- For security reasons, it is recommended that all clients use a proxy to surf the Internet.
- All clients must use a proxy, regardless of whether they are aware of it.
- The proxy in a network is moved, but the existing clients need to retain their old configuration.

In all these cases, a transparent proxy may be used. The principle is very easy: the proxy intercepts and answers the requests of the Web browser, so the Web browser receives the requested pages without knowing from where they are coming. As the name indicates, the entire process is done transparently.

26.5.1 Configuration Options in `/etc/squid/squid.conf`

To inform squid that it should act as a transparent proxy, use the option `transparent` at the tag `http_port` in the main configuration file `/etc/squid/squid.conf`. After restarting squid, the only other thing that must be done is to reconfigure the firewall to redirect the HTTP port to the port given in `http_port`. In the following squid configuration line, this would be the port 3128.

```
http_port 3128 transparent
```

26.5.2 Firewall Configuration with SuSEFirewall2

Now redirect all incoming requests via the firewall with help of a port forwarding rule to the Squid port. To do this, use the enclosed tool SuSEFirewall2, described in *Book "Security Guide", Chapter 15 "Masquerading and Firewalls", Section 15.4.1 "Configuring the Firewall with YaST"*. Its configuration file can be found in `/etc/sysconfig/SuSEfirewall2`. The configuration file consists of well-documented entries. To set a transparent proxy, you must configure several firewall options:

- Device pointing to the Internet: `FW_DEV_EXT = "eth1"`
- Device pointing to the network: `FW_DEV_INT = "eth0"`

Define ports and services (see `/etc/services`) on the firewall that are accessed from untrusted (external) networks such as the Internet. In this example, only Web services are offered to the outside:

```
FW_SERVICES_EXT_TCP="www"
```

Define ports or services (see `/etc/services`) on the firewall that are accessed from the secure (internal) network, both via TCP and UDP:

```
FW_SERVICES_INT_TCP="domain www 3128"  
FW_SERVICES_INT_UDP="domain"
```

This allows accessing Web services and Squid (whose default port is `3128`). The service “domain” stands for DNS (domain name service). This service is commonly used. Otherwise, simply take it out of the above entries and set the following option to `no`:

```
FW_SERVICE_DNS="yes"
```

The most important option is option number 15:

EXAMPLE 26.1: FIREWALL CONFIGURATION: OPTION 15

```
# 15.)
# Which accesses to services should be redirected to a local port on
# the firewall machine?
#
# This option can be used to force all internal users to surf via
# your squid proxy, or transparently redirect incoming webtraffic to
# a secure webserver.
#
# Format:
# list of <source network>[,<destination network>,<protocol>[,dport[:lport]]
# Where protocol is either tcp or udp. dport is the original
# destination port and lport the port on the local machine to
# redirect the traffic to
#
# An exclamation mark in front of source or destination network
# means everything EXCEPT the specified network
#
# Example: "10.0.0.0/8,0/0,tcp,80,3128 0/0,172.20.1.1,tcp,80,8080"
```

The comments above show the syntax to follow. First, enter the IP address and the netmask of the internal networks accessing the proxy firewall. Second, enter the IP address and the netmask to which these clients send their requests. In the case of Web browsers, specify the networks 0/0, a wild card that means “to everywhere.” After that, enter the original port to which these requests are sent and, finally, the port to which all these requests are redirected. Because Squid supports protocols other than HTTP, redirect requests from other ports to the proxy, such as FTP (port 21), HTTPS, or SSL (port 443). In this example, Web services (port 80) are redirected to the proxy port (port 3128). If there are more networks or services to add, they must be separated by a blank space in the respective entry.

```
FW_REDIRECT="192.168.0.0/16,0/0,tcp,80,3128"
```

To start the firewall and the new configuration with it, change an entry in the /etc/sysconfig/SuSEfirewall2 file. The entry START_FW must be set to "yes".

Start Squid as shown in [Section 26.3, “Starting Squid”](#). To verify that everything is working properly, check the Squid log files in /var/log/squid/access.log.

To verify that all ports are correctly configured, perform a port scan on the machine from any computer outside your network. Only the Web services (port 80) should be open. To scan the ports with nmap, the command syntax is nmap -0 IP_address.

26.6 cachemgr.cgi

The cache manager (`cachemgr.cgi`) is a CGI utility for displaying statistics about the memory usage of a running Squid process. It is also a more convenient way to manage the cache and view statistics without logging the server.

26.6.1 Setup

First, a running Web server on your system is required. Configure Apache as described in *Chapter 24, The Apache HTTP Server*. To check if Apache is already running, as `root` enter the command `systemctl status apache2`. Otherwise, enter `systemctl start apache2` to start Apache with the openSUSE Leap default settings. The last step to set it up is to copy the file `cachemgr.cgi` to the Apache directory `cgi-bin`. For 32-bit, this works as follows:

```
cp /usr/lib/squid/cachemgr.cgi /srv/www/cgi-bin/
```

In a 64-bit environment, the file `cachemgr.cgi` is located below `/usr/lib64/squid/` and the command to copy it to the Apache directory is the following:

```
cp /usr/lib64/squid/cachemgr.cgi /srv/www/cgi-bin/
```

26.6.2 Cache Manager ACLs in `/etc/squid/squid.conf`

There are some default settings in the original file required for the cache manager. First, two ACLs are defined, then `http_access` options use these ACLs to grant access from the CGI script to Squid. The first ACL is the most important, because the cache manager tries to communicate with Squid over the `cache_object` protocol.

```
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
```

The following rules give Apache the access rights to Squid:

```
http_access allow manager localhost
http_access deny manager
```

These rules assume that the Web server and Squid are running on the same machine. If the communication between the cache manager and Squid originates at the Web server on another computer, include an extra ACL as in *Example 26.2, "Access Rules"*.

EXAMPLE 26.2: ACCESS RULES

```
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl webserver src 192.168.1.7/255.255.255.255 # webserver IP
```

Then add the rules in *Example 26.3, "Access Rules"* to permit access from the Web server.

EXAMPLE 26.3: ACCESS RULES

```
http_access allow manager localhost
http_access allow manager webserver
http_access deny manager
```

Configure a password for the manager for access to more options, like closing the cache remotely or viewing more information about the cache. For this, configure the entry `cachemgr_passwd` with a password for the manager and the list of options to view. This list appears as a part of the entry comments in `/etc/squid/squid.conf`.

Restart Squid every time the configuration file is changed. Do this easily with `systemctl reload squid`.

26.6.3 Viewing the Statistics

Go to the corresponding Web site—<http://webserver.example.org/cgi-bin/cachemgr.cgi>. Press *continue* and browse through the different statistics.

26.7 squidGuard

This section is not intended to explain an extensive configuration of squidGuard, only to introduce it and give some advice for using it. For more in-depth configuration issues, refer to the squidGuard Web site at <http://www.squidguard.org>.

squidGuard is a free (GPL), flexible, and fast filter, redirector, and access controller plug-in for Squid. It lets you define multiple access rules with different restrictions for different user groups on a Squid cache. squidGuard uses Squid's standard redirector interface. squidGuard can do the following:

- Limit Web access for some users to a list of accepted or well-known Web servers or URLs.
- Block access to some listed or blacklisted Web servers or URLs for some users.
- Block access to URLs matching a list of regular expressions or words for some users.

- Redirect blocked URLs to an “intelligent” CGI-based information page.
- Redirect unregistered users to a registration form.
- Redirect banners to an empty GIF.
- Use different access rules based on time of day, day of the week, date, etc.
- Use different rules for different user groups.

squidGuard and Squid cannot be used to:

- Edit, filter, or censor text inside documents.
- Edit, filter, or censor HTML-embedded script languages, such as JavaScript or VBScript.

Before it can be used, install `squidGuard`. Provide a minimal configuration file as `/etc/squid-guard.conf`. Find configuration examples in <http://www.squidguard.org/Doc/examples.html>. Experiment later with more complicated configuration settings.

Next, create a dummy “access denied” page or a more or less complex CGI page to redirect Squid if the client requests a blacklisted Web site. Using Apache is strongly recommended.

Now, configure Squid to use squidGuard. Use the following entry in the `/etc/squid/squid.conf` file:

```
redirect_program /usr/bin/squidGuard
```

Another option called `redirect_children` configures the number of “redirect” (in this case squidGuard) processes running on the machine. The more processes you set, the more RAM is required. Try low numbers first, for example `4`.

```
redirect_children 4
```

Last, have Squid load the new configuration by running `systemctl reload squid`. Now, test your settings with a browser.

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://Calamaris.Cord.de/>. 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

```
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:

```
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.

Following the installation, a small HOWTO about transparent proxies is available in how-toenh as /usr/share/doc/howto/en/txt/TransparentProxy.gz. 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 406

28 Using NetworkManager 417

29 Power Management 427

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 PCMCIA (Personal Computer Memory Card International Association), 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 towards conservation of power is at least as important as that of the operating system. openSUSE® Leap supports various methods that influence 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 towards 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, unused PCMCIA cards, 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.

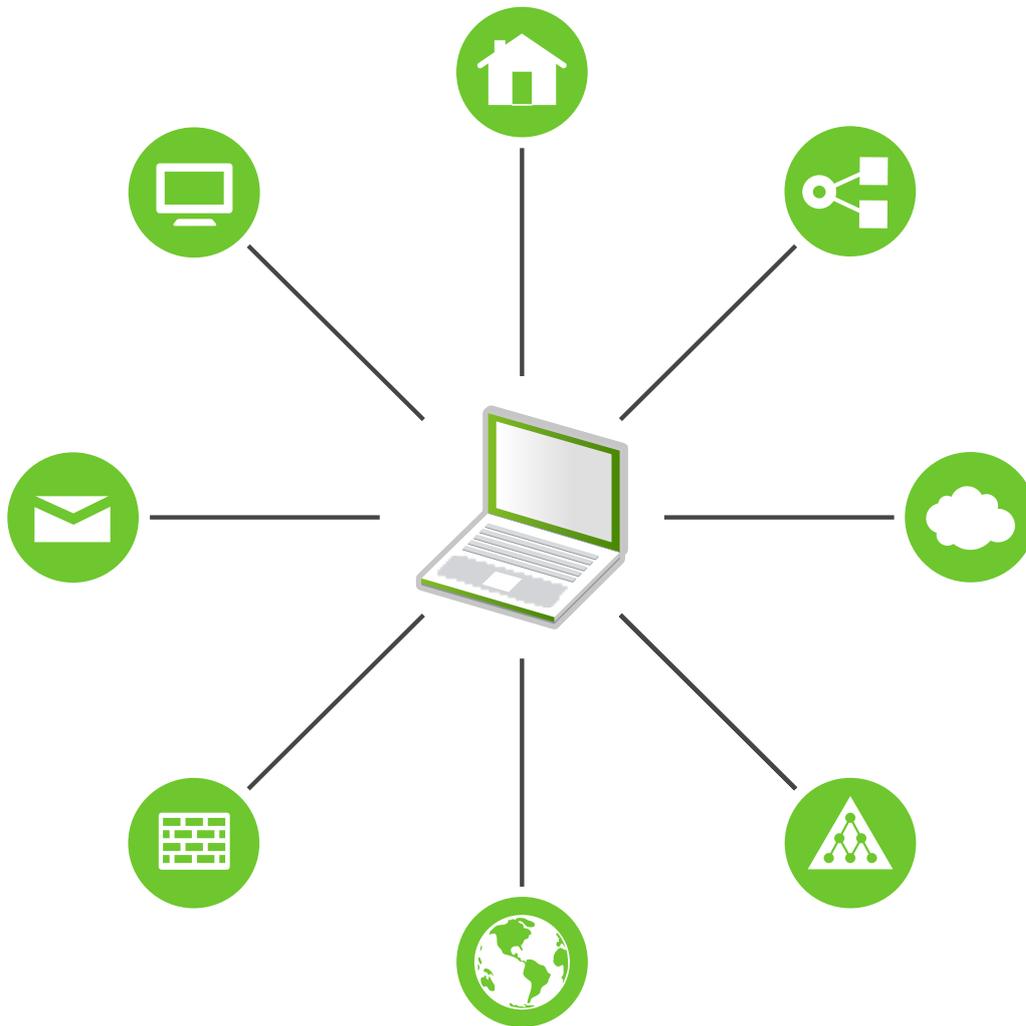


FIGURE 27.1: INTEGRATING A MOBILE COMPUTER IN AN EXISTING ENVIRONMENT

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, backwards-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 SUSE Linux Enterprise Server

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. If you intend to use WPA security, you should not use ad-hoc mode. Be aware, 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 makes use of 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 at all.

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 Guide", Chapter 11 "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 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 as soon as 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)

As soon as 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 from the name it was given by the system, select the corresponding menu item from the menu that opens when the icon is right-clicked. This name change is limited to display in the file manager. The descriptor by which the device is mounted in /media remains unaffected by this.

USB Flash Disks

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 Shotwell. For advanced photo processing use The GIMP. For a short introduction to The GIMP, see *Book "GNOME User Guide", Chapter 18 "GIMP: Manipulating Graphics"*.

27.3 Cellular Phones and PDAs

A desktop system or a laptop can communicate with a cellular phone via Bluetooth or IrDA. Some models support both protocols and some only one of the two. The usage areas for the two protocols and the corresponding extended documentation has already been mentioned in [Section 27.1.3.3, "Wireless Communication: Wi-Fi"](#). The configuration of these protocols on the cellular phones themselves is described in their manuals.

27.4 For More Information

The central point of reference for all questions regarding mobile devices and Linux is <http://tuxmobil.org/>. Various sections of that Web site deal with the hardware and software aspects of laptops, PDAs, cellular phones and other mobile hardware.

A similar approach to that of <http://tuxmobil.org/> is made by <http://www.linux-on-laptops.com/>. Information about laptops and handhelds can be found here.

SUSE maintains a mailing list in German dedicated to the subject of laptops. See <http://lists.opensuse.org/opensuse-mobile-de/>. On this list, users and developers discuss all aspects of mobile computing with openSUSE Leap. Postings in English are answered, but the majority of the archived information is only available in German. Use <http://lists.opensuse.org/opensuse-mobile/> for English postings.

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 moving. 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 having enabled NetworkManager in YaST, configure your network connections with the NetworkManager front-end available in GNOME. It shows tabs for all types of network connections, such as wired, wireless, mobile broadband, DSL, and VPN connections.

To open the network configuration dialog in GNOME, open the settings menu via the status menu and click the *Network* entry.



Note: Availability of Options

Depending on your system setup, you may not be allowed to configure connections. In a secured environment, some options may be locked or require root permission. Ask your system administrator for details.

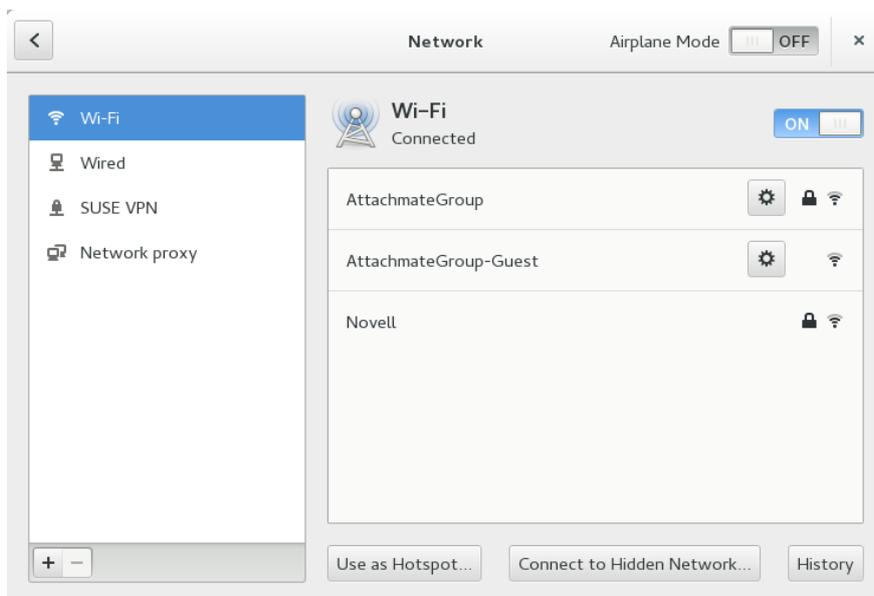


FIGURE 28.1: GNOME NETWORK CONNECTIONS DIALOG

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 having confirmed your changes, the newly configured network connection appears in the list of available networks you get by opening 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 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 and
- NetworkManager-openvpn-gnome.

vpnc (Cisco)

To use this VPN technology, install

- NetworkManager-vpnc and
- NetworkManager-vpnc-gnome.

PPTP (Point-to-Point Tunneling Protocol)

To use this VPN technology, install

- NetworkManager-pptp and
- NetworkManager-pptp-gnome.

After you have installed the packages, configure your VPN connection as described in *Procedure 28.1, "Adding and Editing Connections"*.

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 that 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 are connections that become available to NetworkManager when the first user logs in. Any required credentials are asked from the user and when the user logs out, the connections are disconnected and removed from NetworkManager. Connections that are defined as system connection can be shared by all users and are made available right after NetworkManager is started—before any users log in. In case of system connections, all credentials must be provided at the time the connection is created. Such system connections can be used to automatically connect to networks that require authorization. For information 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 Guide", Chapter 12 "Certificate Store"*.

28.5 Frequently Asked Questions

In the following, find some frequently asked questions about configuring special network options with NetworkManager.

28.5.1. 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 so 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.

28.5.2. 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.

28.5.3. How to share network connections to 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.

28.5.4. 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.

28.5.5. 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. 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 (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 in 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 out 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 page exceeds 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 makes use of 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 that 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

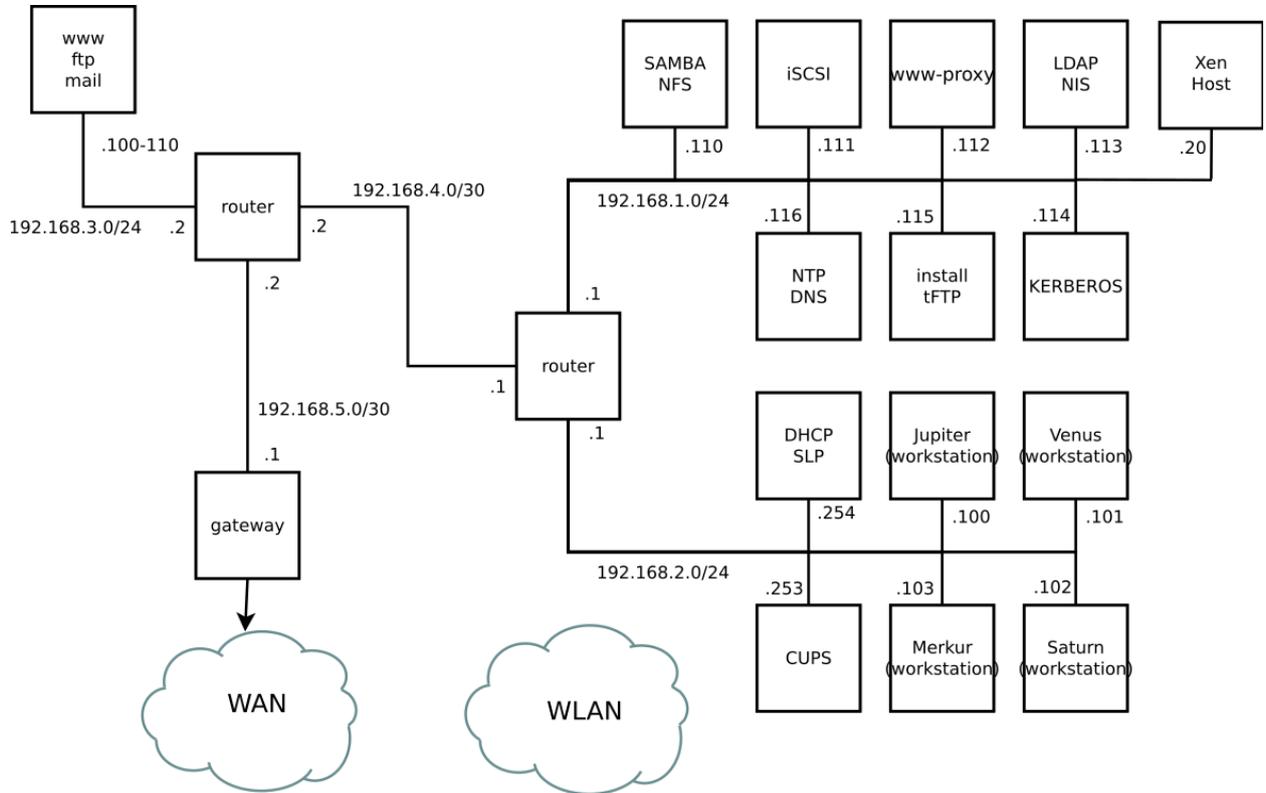
Refer to the kernel sources to see if your processor is supported. You may need a special kernel module or module option to activate CPU frequency control. If the `kernel-source` package is installed, this information is available in </usr/src/linux/Documentation/cpu-freq/>.

29.5 For More Information

- http://en.opensuse.org/SDB:Suspend_to_RAM —How to get Suspend to RAM working
- <http://old-en.opensuse.org/Pm-utils> —How to modify the general suspend framework

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.