The Media Streaming Journal

July 2019

Covering Audio and Video Internet Broadcasting

Brought To You By

RADIOSOLUTION

www.radiosolution.info

publicdomainvectors.org/en/free-clipart/Vintage-microphone-vector-graphics/6111.html

The Media Streaming Journal Staff

Derek Bullard
Publication Director
info@radiosolution.info

David Childers
Editor In Chief
editor@radiosolution.info

Advertising
advertising@radiosolution.info

www.radiosolution.info

Welcome to The Media Streaming Journal

Greetings,

Functionality and stability are one of the core elements associated with the Linux community of operating systems. Suse Linux was originated in the early 90's as one of the original Linux distributions. It has undergone various changes in proprietorships and direction but has maintained a strong community support. The openSUSE Linux distribution continues the tradition of functionality and stability; through the vision and guidance of the openSUSE community. If you are looking for a quality Linux distribution that can be used in a comprehensive production environment, look no further than openSUSE.

http://www.opensuse.org

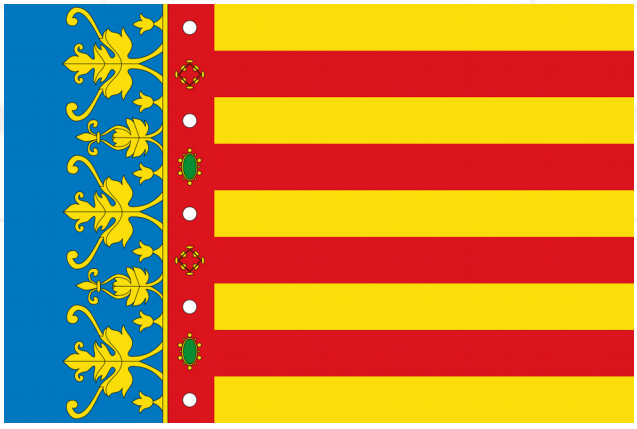
Please feel free to contact either the Publication Director (Derek Bullard) or myself if you have any questions or comments regarding The Media Streaming Journal.

Namaste

David Childers

www.linkedin.com/pub/david-childers/4/736/72a

The Grand Master of Digital Disaster
(Editor In Chief)



L'amistat es mesura per moments, mai per temps.

David Childers

The Grand Master of Digital Disaster

Current Member: International Association Of Internet Broadcasters

Former Member: Society of Motion Picture and Television Engineers

Published Author

Introduction To Internet Broadcasting
Amazon Publishing

Numerous Creative Commons Computer, Technical and Internet Broadcasting Guides
http://www.ScenicRadio.com/Library/BroadGuide/index.html

Newspaper Interviews

New York Times

Internet TV: Don't Touch That Mouse!
Tim Gnatek
July 1, 2004

Lagniappe - "Something Extra for Mobile"

Mobile Gets Hoaxed
Rob Holbert
Mar 16, 2016

Cited By

Five Essays on Copyright In the Digital Era
Ville Oksanen
2009

Turre Publishing
Helsinki Finland

Open Source Developer

Developed software architecture to continuously source multimedia content to Youtube Live servers.
Scenic Television – The sights and sounds of nature on the Internet.
http://www.ScenicRadio.com

Projects

Researched and developed documentation for Peercast P2P multimedia streaming project.
http://en.wikipedia.org/wiki/PeerCast

Researched and developed technical documentation for NSV / Winamp Television.
http://web.archive.org/web/20080601000000*/http://www.scvi.net

MidSummer Eve Webfest

A virtual International festival focusing on Digital art and Free Software that was coordinated by OrganicaDTM Design Studio.

Presentation and discussion regarding Internet multimedia content distribution.
http://web.archive.org/web/20061104230522/http://www.organicadtm.com/index.php?module=articles&func=display&catid=37&aid=61

LinkedIn Contact Information

http://www.linkedin.com/pub/david-childers/4/736/72a

The Media Streaming Journal

What is in this edition of the Media Streaming Journal

Start-Up openSUSE Leap 15.1

Security Guide openSUSE Leap 15.1

Reference openSUSE Leap 15.1

Join our technical discussion on Facebook

http://www.facebook.com/groups/internetradiosupport/

Magazine cover:

https://commons.wikimedia.org/wiki/File:Laptop-hard-drive-exposed.jpg

Flag of Valencia:

http://commons.wikimedia.org/wiki/File:Flag_of_the_Valencian_Community_(2x3).svg

RADIOSOLUTION

www.radiosolution.info

Our Mission

Let our friendly, knowledgeable staff assist you to build your project, such as an online radio station using our high end reliable video and audio streaming technologies. We want to become your partner for all your hosting needs, as well as your one stop shop for radio products such as custom DJ drops and radio ID's.

Start An Internet Radio Station

Whatever you need to start Internet radio station, we will deliver! We provide high quality Internet Radio services to make your music radio project a success. We can provide Wowza, Icecast, SHOUTcast hosting and internet radio services to hobbyists, deejays, amateurs and established professionals. No radio station client is too big or too small for Radiosolution.

Choose between complete hassle-free service packages or new features to add to start internet radio station. Benefit from customized services and the latest in internet radio technology. You will receive professional, personalized and better Internet Radio Station services than you have received up till now. If you already have an Icecast or SHOUTcast hosting provider, we can still help you transfer your radio server over to us with no hassle and at no charge.

Internet Radio Station Services

Launch your internet, digital, satellite or AM/FM radio station anywhere in the world with all of the right tools. A broadcasting specialist is on standby to help you get started with an SHOUTcast or Icecast hosting package. We have servers ready for reliable streaming in North America and Europe. Our hosting packages have all the features you need to make your radio station project a success.

If you stream live or with an Auto DJ, we can provide you with the latest in web-based Cloud technology. You will love the simple to use control panel. Discover how easy it is to manage live deejays, upload fresh music and create custom scheduled programming. You will be able to track your listeners by getting real time statistics.

Starting your own Internet radio has never been easier. Get in touch with us anytime to start your Internet radio station.

Radiosolution is a SHOUTcast hosting provider located in Quebec Canada. We also offer Icecast, Wowza and Web Hosting services. Contact us to discuss the best option available as you start internet radio station. Radiosolution can provide personalized service in English, Dutch, and French. Starting an internet radio station can be intimidating, many people want to start one, but have no idea where to start. Radiosolution will be there for you every step of the way. Everyday people are searching the internet for free SHOUTcast servers. With Radiosolution SHOUTcast hosting we will allow you to try our services for FREE. By trying our services, you can be confident that you have chosen the best radio server hosting provider. You have nothing to loose because we offer a 30 day satisfaction guarantee. What are you waiting for? Contact us now! Radiosolution offers everything you need to start internet radio station. You will not need to go anywhere else. We can create your website, market your station and help you submit your station to online directories. We also feature the voice of Derek Bullard aka Dibblebee He can create affordable commercials, DJ intros, sweepers, jingles, ids and so much more.

Relax With The Sights And Sounds Of Nature

Scenic Television

Your Window To The World

Scenic Television is an Internet television station that broadcasts the sights and sounds of nature 24 hours a day. Savor exotic tropical beaches, or relax in a remote rain forest. Meditate at a bubbling stream, or relish the view of soft rolling waves at a lake. We have beautiful nature video from locations all around the world.

Scenic Television originates from the Gulf coast of South Alabama and broadcasts to a global audience. The television broadcast is accessible on any device with an Internet connection. Such electronic devices include desktop computers, laptops, tablets, smartphones, game platforms, and Internet-connected televisions.

http://www.scenictelevision.com

all-free-download.com/free-vector/download/magnifying_glass_clip_art_23181.html

We Are Your Information Resource

Are you looking for specialized data?

Are you swamped with information overload?

Do you need help finding the right information?

We Can Help You
Find The Information
That You Need

Our experienced data research analysts can wade through the vast information wasteland and find the information that you need.

We can save you both time and money.

We can streamline data requirement planning.

We can provide business critical information acquisition.

Contact us today

info@radiosolution.info

Start-Up openSUSE Leap 15.1

This manual will see you through your initial contact with openSUSE® Leap.

Security Guide openSUSE Leap 15.1

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.

Reference openSUSE Leap 15.1

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 con-guration scenarios.

Start-Up

openSUSE Leap 15.1

Start-Up

openSUSE Leap 15.1

Publication Date: May 27, 2019

SUSE LLC
10 Canal Park Drive
Suite 200
Cambridge MA 02141
USA
https://www.suse.com/documentation

Contents

About This Guide

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.

Installation

Guides you through the installation process and the basic configuration of your system. The Quick Start section shows a quick walk through the installation using default values. The second part of this chapter provides details for every installation step.

Administration

Introduces YaST, the central tool for installation and configuration of your system. Learn how to initially set up your system and how to modify key components of your system.

Managing and Updating Software

Understand how to install or remove software with either YaST or using the command line, how to use the 1-Click Install feature, and how to keep your system up-to-date.

The Bash Shell

Learn how to work with the bash shell, the default command line interpreter on openSUSE Leap. Get to know the most commonly used Linux commands and understand basic concepts of a Linux system.

Help and Troubleshooting

Provides an overview of where to find help and additional documentation in case you need more information or want to perform specific tasks with your system. Also find a compilation of the most frequent problems and annoyances and learn how to solve these problems on your own.

1 Available Documentation

Note: Online Documentation and Latest Updates

Documentation for our products is available at http://doc.opensuse.org/, where you can also find the latest updates, and browse or download the documentation in various formats.

In addition, the product documentation is usually available in your installed system under `/usr/share/doc/manual`.

The following documentation is available for this product:

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.

***Book* "Reference"**

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

***Book* "Virtualization Guide"**

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

***Book* "AutoYaST Guide"**

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

***Book* "Security 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.

2 Feedback

Several feedback channels are available:

Bug Reports

To report bugs for openSUSE Leap, go to https://bugzilla.opensuse.org/, log in, and click *New*.

Mail

For feedback on the documentation of this product, you can also send a mail to `doc-team@suse.com`. 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 notices and typographical conventions are used in this documentation:

- `/etc/passwd`: directory names and file names
- *`PLACEHOLDER`*: replace *`PLACEHOLDER`* with the actual value
- `PATH`: the environment variable PATH
- **`ls`**, `--help`: commands, options, and parameters
- `user`: users or groups
- `package name` : name of a package
- Alt, Alt–F1: a key to press or a key combination; keys are shown in uppercase as on a keyboard
- *File*, *File* › *Save As*: menu items, buttons
- *Dancing Penguins* (Chapter *Penguins*, ↑Another Manual): This is a reference to a chapter in another manual.
- Commands that must be run with `root` privileges. Often you can also prefix these commands with the **`sudo`** command to run them as non-privileged user.

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

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

```
tux > command
```

- Notices

Warning: Warning Notice

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

Important: Important Notice

Important information you should be aware of before proceeding.

Note: Note Notice

Additional information, for example about differences in software versions.

Tip: Tip Notice

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

4 About the Making of This Documentation

This documentation is written in GeekoDoc (https://github.com/openSUSE/geekodoc), 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 Installation

1 *Installation Quick Start*

Use the following procedures to install a new version of openSUSE® Leap 15.1. This document gives a quick overview on how to run through a default installation of openSUSE Leap on the x86_64 architecture.

1.1 Welcome to openSUSE Leap

For more detailed installation instructions see *Chapter 3, Installation Steps*.

1.1.1 Minimum System Requirements

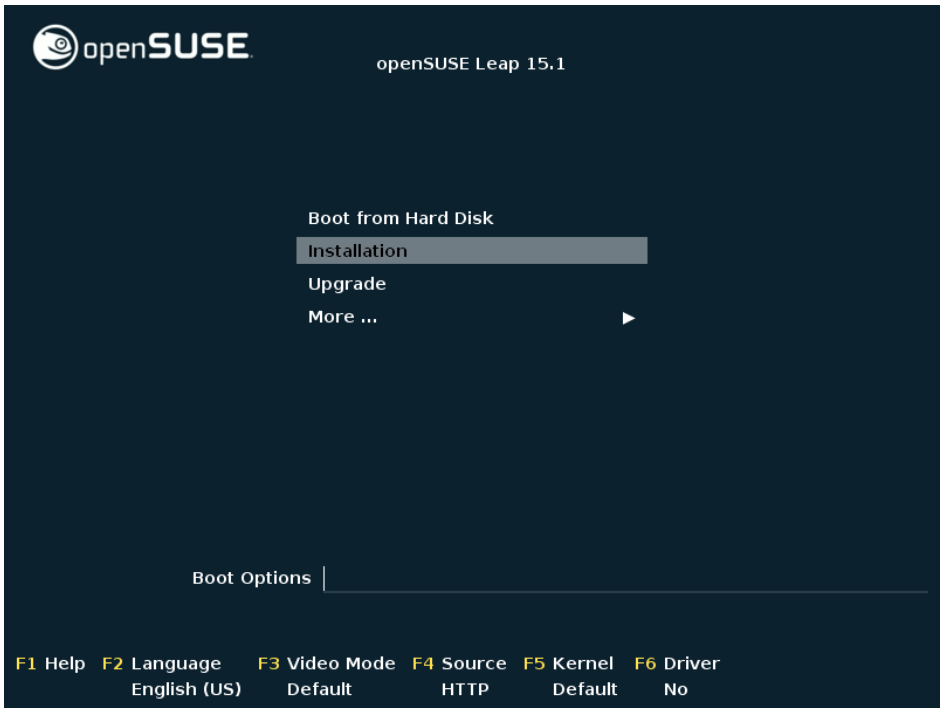
- any AMD64/Intel* EM64T processor (32-bit processors are not supported)
- 1 GB physical RAM (4 GB or more strongly recommended)
- 10 GB available disk space for a minimal installation, 16 GB for a graphical desktop (more is recommended). In case you plan to use Btrfs snapshots a minimum of 40 GB for the root partition is recommended.
- Supports most modern sound and graphics cards, 1024 x 768 display resolution (higher recommended)

1.1.2 Installing openSUSE Leap

Use these instructions if there is no existing Linux system on your machine, or if you want to replace an existing Linux system.

1.1.2.1 Booting the Installation System

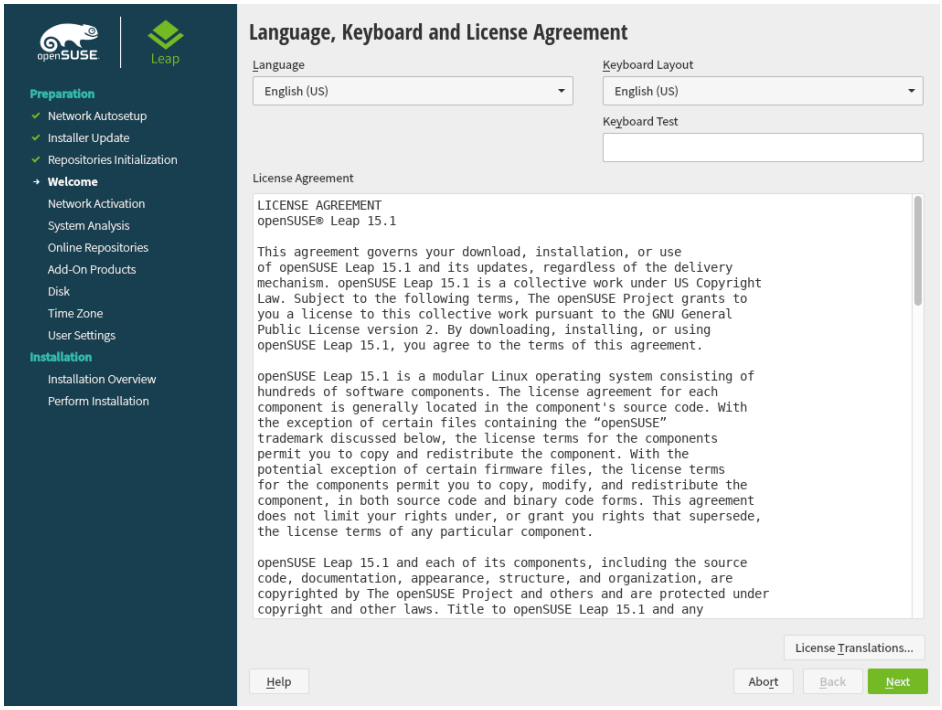
Insert a DVD or a bootable USB stick containing the installation image for openSUSE Leap, then reboot the computer to start the installation program. On machines with a traditional BIOS you will see the graphical boot screen shown below. On machines equipped with UEFI, a slightly different boot screen is used. Secure boot on UEFI machines is supported.



On BIOS machines, use F2 to change the language for the installer. A corresponding keyboard layout is chosen automatically. See *Section 2.2.1, "The Boot Screen on Machines Equipped with Traditional BIOS"* or *Section 2.2.2, "The Boot Screen on Machines Equipped with UEFI"* for more information about changing boot parameters. On UEFI machines adjust the language and keyboard settings in the next step.

Select *Installation* on the boot screen, then press Enter. This boots the system and loads the openSUSE Leap installer.

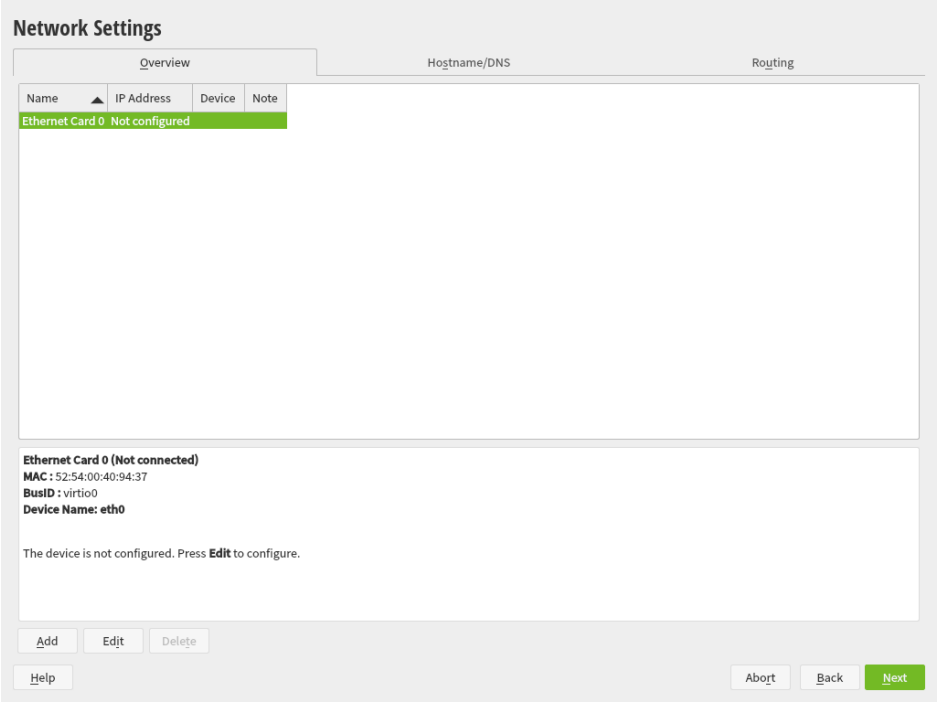
1.1.2.2 Language, Keyboard and License Agreement



On systems with a traditional BIOS the *Language* and *Keyboard Layout* settings are initialized with the language you chose at the boot screen. If you did not change the default, or are using a UEFI machine it will be English (US). Change the settings here, if necessary. Use the *Keyboard Test* text box to test the layout.

Read the License Agreement. It is presented in the language you have chosen. Other *License Translations* are available. Proceed with *Next*.

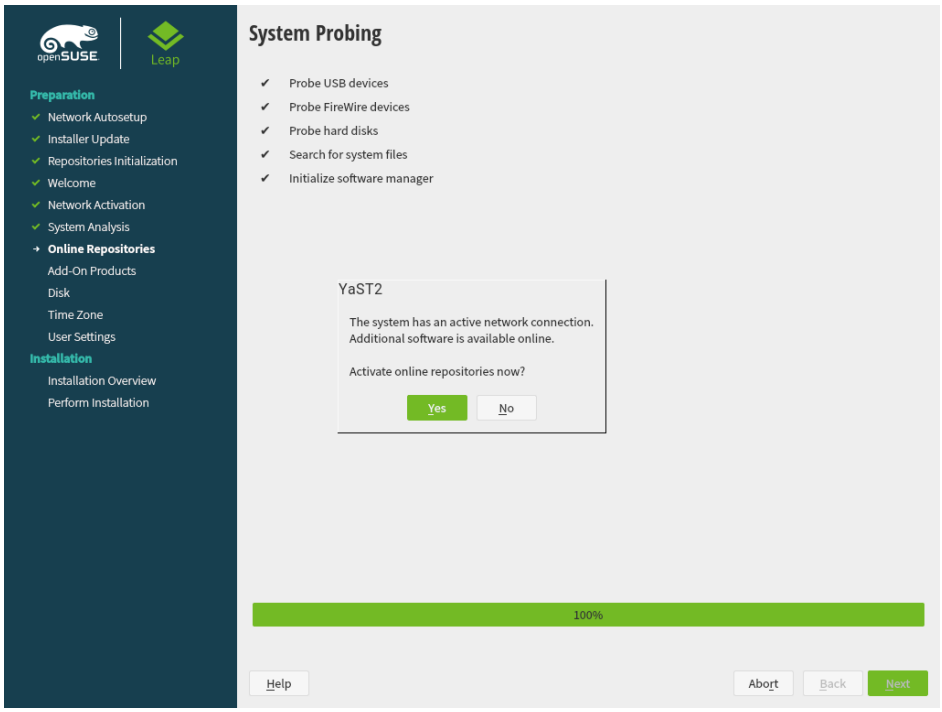
1.1.2.3 Network Settings



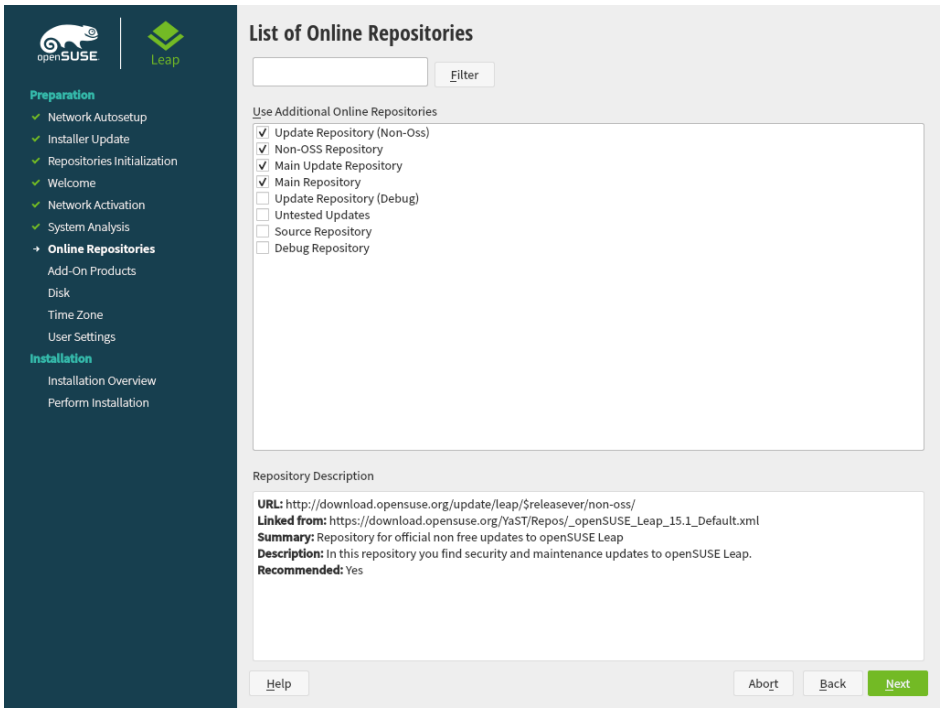
If the network can not be configured automatically, the *Network Settings* dialog opens. Choose a network interface from the list and configure it with *Edit*. Alternatively, *Add* an interface manually. See *Section 3.4, "Network Settings"* and *Book "*Reference*", Chapter 13 "Basic Networking", Section 13.4 "Configuring a Network Connection with YaST"* for more information. If you prefer to do an installation without network access, skip this step without making any changes and proceed with *Next*.

1.1.2.4 Online Repositories

A system analysis is performed, where the installer probes for storage devices, and tries to find other installed systems. If a network connection with Internet access is available, you will be asked to activate the online repositories. Answer with *Yes* to proceed. In case you do not have Internet access, this step will be skipped.



The online repositories are official openSUSE package sources. They not only offer additional packages not included on the installation media, but also the update repositories containing security and bug fixes. Using the default selection is recommended. Add at least the *Main Update Repository*, because it makes sure the system is installed with the latest security patches.

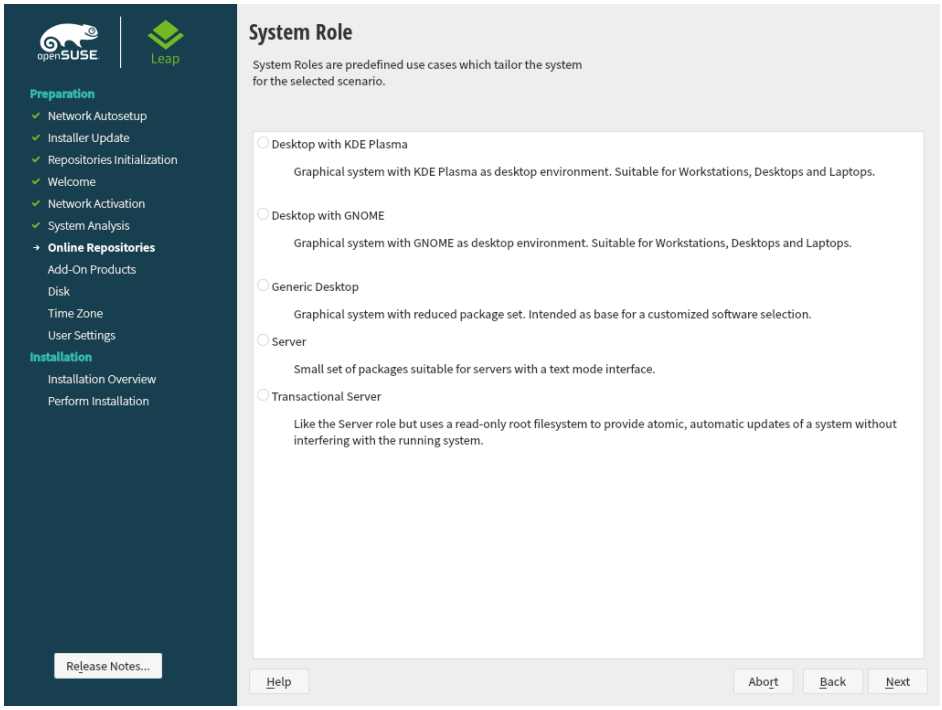


You have the following choices:

- The *Main Repository (OSS)* contains open source software (OSS). Compared to the DVD installation media, it contains many additional software packages, among them many additional desktop systems.
- The *Main Update Repository* contains security updates and fixes for packages from the *Main Repository (OSS)* and the DVD installation media. Choosing this repository is recommended for all installation scenarios.
- The *Main Repository (Non-OSS)* contains packages with a proprietary software license. Choosing it is not required for installing a custom desktop system.
- Choosing *Main Update Repository (Non-OSS)* is recommended when also having chosen the *Main Repository (Non-OSS)*. It contains the respective updates and security fixes.
- All other repositories are intended for experienced users and developers. Click on a repository name to get more information.

Confirm your selection with *Next*. Depending on your choice, you need to confirm one or more license agreements. Do so by choosing *Next* until you proceed to the *System Role* screen. Now choose *Next* to proceed.

1.1.2.5 System Role



Choose a general software and system configuration with this step by selecting a desktop or server configuration.

For a desktop installation, choose between *Desktop with KDE Plasma*, *Desktop with GNOME* and *Generic Desktop*. KDE is slightly similar to Windows, GNOME offers an alternative, innovative environment. In case you prefer an alterntive to the KDE or GNOME desktops, choose *Generic Desktop*. You will be able to choose between the XFCE, LXDE, MATE and others later in the installation process by selecting *Software* in the *Installation Settings dialog*.

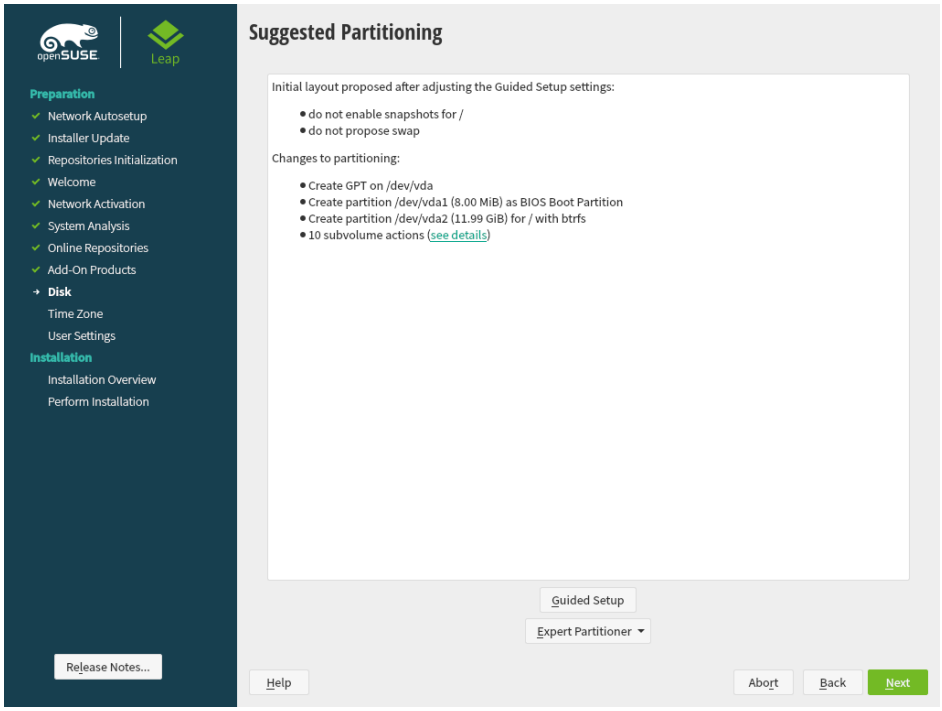
If setting up a server, you probably do not need a graphical user interface. Choose *Server (Text Mode)* in this case. Alternatively, set up a server system with a read-only root partition and transactional updates by choosing *Transactional Server*. This selection also is a prerequisite for setting up openSUSE Kubic. See https://kubic.opensuse.org/blog/2018-04-04-transactionalupdates/ for more information on transactional updates.

You can also manually choose the software configuration for your system. Select *Custom* and then *Next* to get to the *Software Selection and System Tasks* dialog. Choose one or more patterns for installation. By clicking *Details*, you can select individual packages.

Tip: Release Notes

From this point on, the Release Notes can be viewed from any screen during the installation process by selecting *Release Notes*.

1.1.2.6 Suggested Partitioning



Define a partition setup for openSUSE Leap in this step. Review the partition setup proposed by the system. If necessary, change it. You have the following options:

Guided Setup

Starts a wizard which lets you refine the partitioning proposal. Options available here depend on your system setup. In case it contains more than a single hard disk, you may choose which disk(s) to use and where to place the root partition. If the disk(s) already contain partitions, decide whether to remove or resize them.

In subsequent steps you may also add LVM support and disk encryption. You can change the file system for the root partition and decide whether to have a separate home partition or not.

Expert Partitioner

Opens the *Expert Partitioner* described in *Book "Reference", Chapter 5 "Expert Partitioner", Section 5.1 "Using the Expert Partitioner"*. This gives you full control over the partitioning setup and lets you create a custom setup. This option is intended for experts.

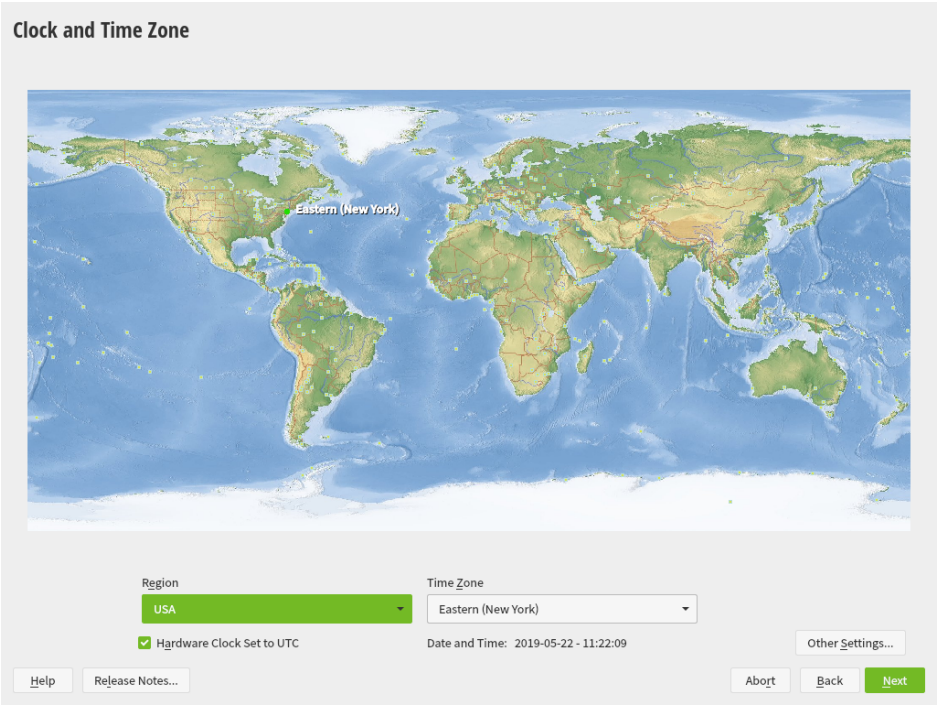
Note: Separate Home Partition

The default proposal no longer suggests to create a separate partition for `/home`. The `/home` directory contains the user's data and personal configuration files. Placing it on a separate directory makes it easier to rebuild the system in the future, or allows to share it with different Linux installations on the same machine.

In case you want to change the proposal to create a separate partition for `/home`, choose *Guided Setup* and click *Next* until you reach the *Filesystem Options* screen. Check *Propose Separate Home Partition*. By default it will be formatted with *XFS*, but you can choose to use a different file system. Close the dialog by clicking *Next* again.

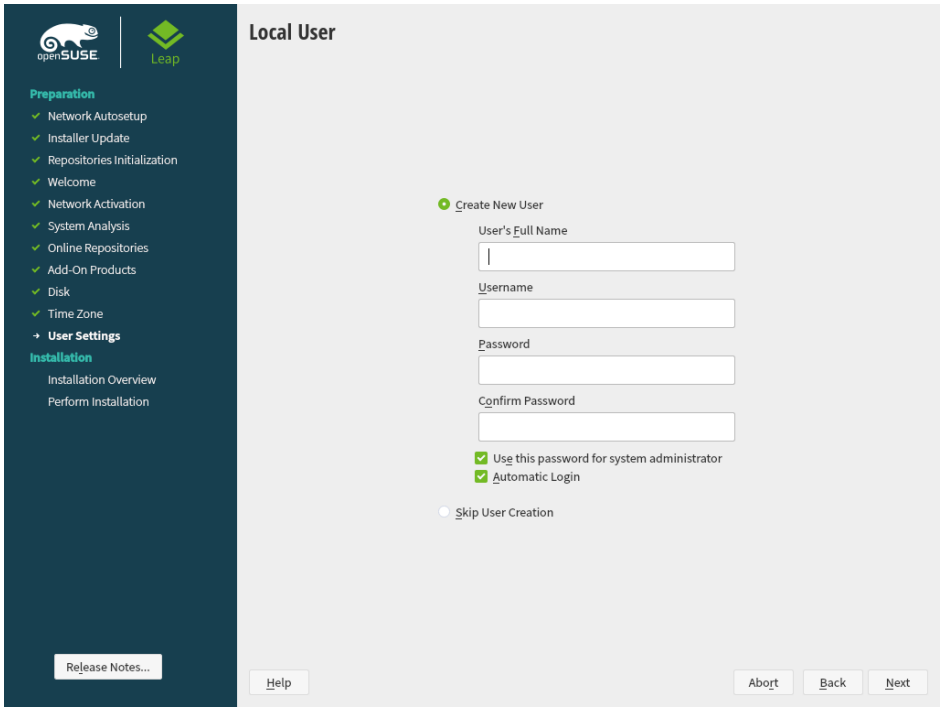
To accept the proposed setup without any changes, choose *Next* to proceed.

1.1.2.7 Clock and Time Zone



Select the clock and time zone to use in your system. To manually adjust the time or to configure an NTP server for time synchronization, choose *Other Settings*. See *Section 3.8, "Clock and Time Zone"* for detailed information. Proceed with *Next*.

1.1.2.8 Local User



To create a local user, type the first and last name in the *User's Full Name* field, the login name in the *Username* field, and the password in the *Password* field.

The password should be at least eight characters long and should contain both uppercase and lowercase letters and numbers. The maximum length for passwords is 72 characters, and passwords are case-sensitive.

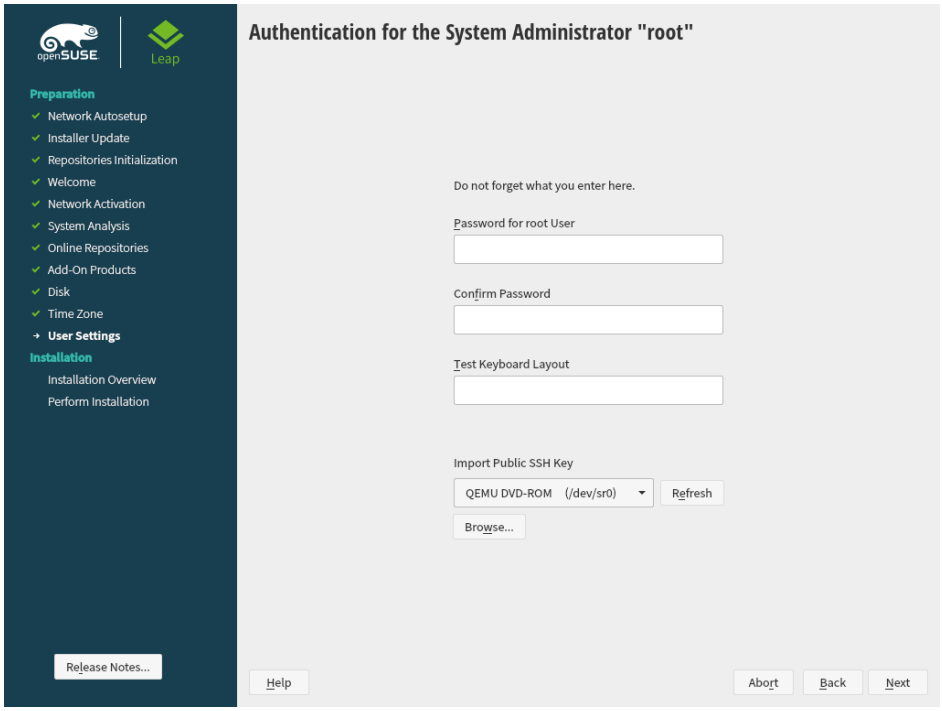
For security reasons it is also strongly recommended *not* to enable the *Automatic Login*. You should also *not Use this Password for the System Administrator* but rather provide a separate `root` password in the next installation step.

If you install on a system where a previous Linux installation was found, you may *Import User Data from a Previous Installation*. Click *Choose User* for a list of available user accounts. Select one or more user.

In an environment where users are centrally managed (for example by NIS or LDAP) you may want to skip the creation of local users. Select *Skip User Creation* in this case.

Proceed with *Next*.

1.1.2.9 Authentication for the System Administrator “root”



Provide a password for the system administrator account (called the `root` user).

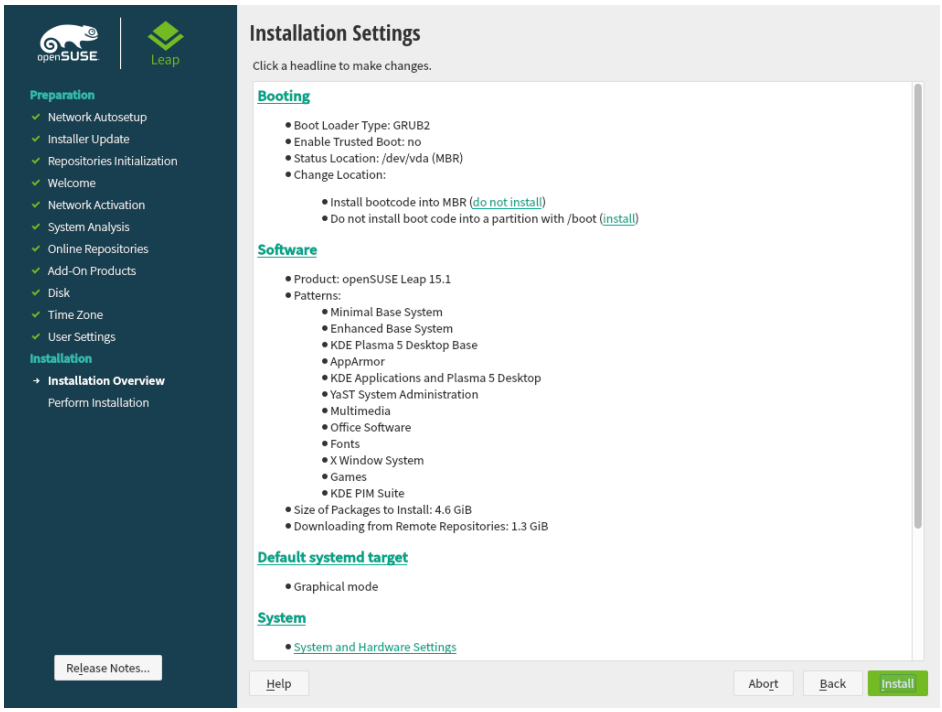
You should never forget the `root` password! After you entered it here, the password cannot be retrieved. See *Section 3.10, “Authentication for the System Administrator “root”” for more information.*
Proceed with *Next*.

Tip: Passwords and Keyboard Layout

It is recommended to only use characters that are available on an English keyboard. In case of a system error or when you need to start your system in rescue mode a localized keyboard might not be available.

In case you would like to enable password-less authentication via SSH login, you can import a key via *Import Public SSH Key*. If you want to completely disable `root` login via password, upload a key only and do not provide a root password. A login as system administrator will only be possible via SSH using the respective keyin this case.

1.1.2.10 Installation Settings



Use the *Installation Settings* screen to review and—if necessary—change several proposed installation settings. The current configuration is listed for each setting. To change it, click the headline. Some settings, such as firewall or SSH can directly be changed by clicking the respective links.

Tip: Remote System Access

Changes you can make in the *Installation Settings*, can also be made later at any time from the installed system. However, if you need remote access directly after the installation, you should adjust the *Firewall and SSH* settings by opening the SSH port and enabling the SSH server.

Booting

This section shows the boot loader configuration. Changing the defaults is only recommended if really needed. Refer to *Book "*Reference*", Chapter 12 "The Boot Loader GRUB 2"* for details.

Software

The default scope of software includes the base system and X Window with the selected desktop. Clicking *Software* opens the *Software Selection and System Tasks* screen, where you can change the software selection by selecting or deselecting patterns. Each pattern contains several software packages needed for specific functions (for example, Web and LAMP server or a print server). For a more detailed selection based on software packages to install, select *Details* to switch to the YaST *Software Manager*. See *Chapter 10, Installing or Removing Software* for more information.

Default Systemd Target

If you have chosen to install a desktop system, the system boots into the *graphical* target, with network, multiuser and display manager support. If you have not installed a desktop, the system boots into a login shell (*Text Mode*).

System

View detailed hardware information by clicking *System*. In the resulting screen you can also change *Kernel Settings*—see *Section 3.11.6, "System"* for more information.

Security

The *CPU Mitigations* refer to kernel boot command line parameters for software mitigations that have been deployed to prevent CPU side-channel attacks. Click the highlighted entry to choose a different option. For details, see *Book "*Reference*", Chapter 12 "The Boot Loader GRUB 2" CPU Mitigations*.

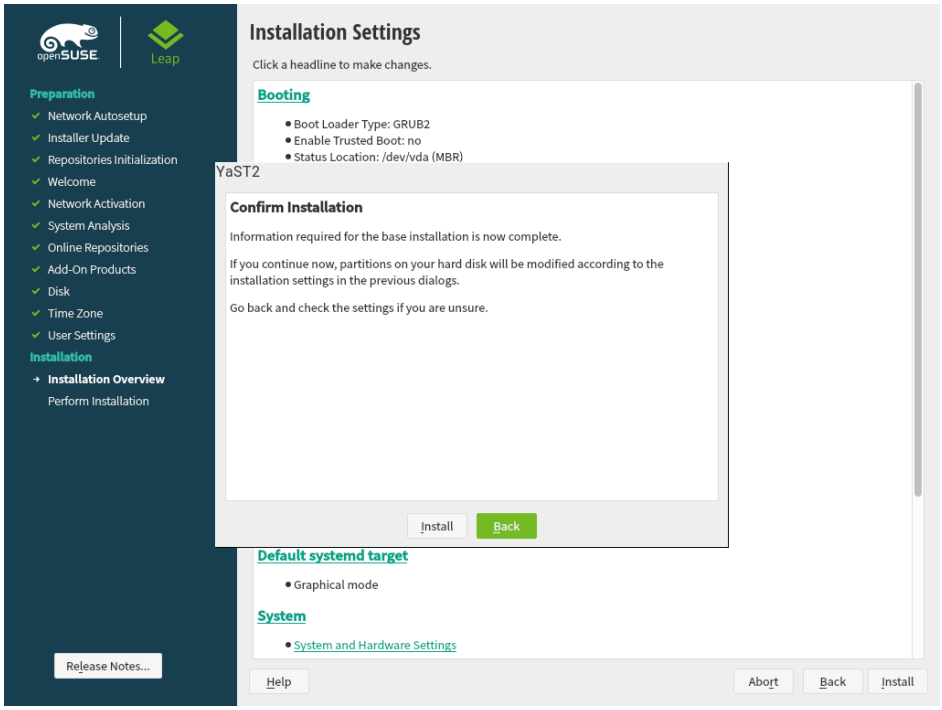
By default, the Firewall is enabled with all network interfaces configured for the public zone. See *Book "*Security Guide*", Chapter 16 "Masquerading and Firewalls", Section 16.4 "*`firewalld`*"* for configuration details.

The SSH service is disabled by default, its port (22) is closed. Therefore logging in from remote is not possible by default. Click *enable* and *open* to toggle these settings.

Network Configuration

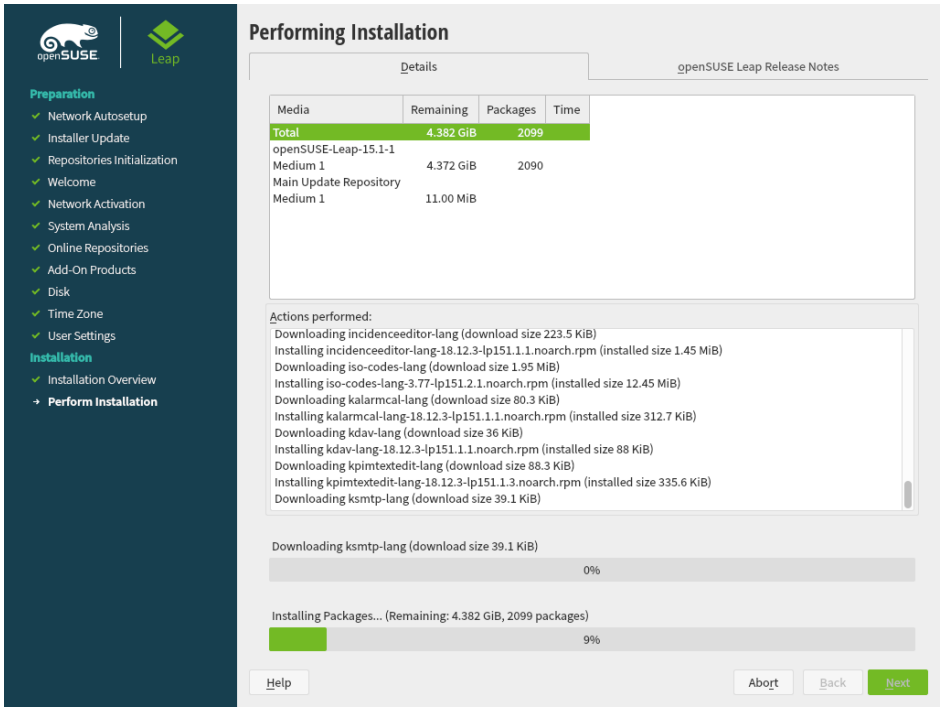
Displays the current network configuration. Click *Network Configuration* to change the settings. For details, see *Book "*Reference*", Chapter 13 "Basic Networking", Section 13.4 "Configuring a Network Connection with YaST"*.

1.1.2.11 Start the Installation



After you have finalized the system configuration on the *Installation Settings* screen, click *Install*. Depending on your software selection you may need to agree to license agreements before the installation confirmation screen pops up. Up to this point no changes have been made to your system. After you click *Install* a second time, the installation process starts.

1.1.2.12 The Installation Process



During the installation, the progress is shown in detail on the *Details* tab. The *openSUSE Leap Release Notes* tab shows important information; reading them is recommended.

After the installation routine has finished, the computer is rebooted into the installed system. Log in and start YaST to fine-tune the system. If you are not using a graphical desktop or are working from remote, refer to *Book "Reference", Chapter 1 "YaST in Text Mode"* for information on using YaST from a terminal.

2 Boot Parameters

openSUSE Leap allows setting several parameters during boot, for example choosing the source of the installation data or setting the network configuration.

Using the appropriate set of boot parameters helps simplify your installation procedure. Many parameters can also be configured later using the linuxrc routines, but using the boot parameters is easier. In some automated setups, the boot parameters can be provided with `initrd` or an `info` file.

The way the system is started for the installation depends on the architecture—system start-up is different for PC (AMD64/Intel 64) or mainframe, for example. If you install openSUSE Leap as a VM Guest on a KVM or Xen hypervisor, follow the instructions for the AMD64/Intel 64 architecture.

Note: Boot Options and Boot Parameters

The terms *Boot Parameters* and *Boot Options* are often used interchangeably. In this documentation, we mostly use the term *Boot Parameters*.

2.1 Using the Default Boot Parameters

The boot parameters are described in detail in *Chapter 3, Installation Steps*. Generally, selecting *Installation* starts the installation boot process.

If problems occur, use *Installation—ACPI Disabled* or *Installation—Safe Settings*. For more information about troubleshooting the installation process, refer to *Chapter 4, Troubleshooting*.

The menu bar at the bottom of the screen offers some advanced functionality needed in some setups. Using the function keys (`F1` ... `F12`), you can specify additional options to pass to the installation routines without having to know the detailed syntax of these parameters (see *Chapter 2, Boot Parameters*). A detailed description of the available function keys is available in *Section 2.2.1, “The Boot Screen on Machines Equipped with Traditional BIOS”*.

2.2 PC (AMD64/Intel 64/ARM AArch64)

This section describes changing the boot parameters for AMD64, Intel 64, and ARM AArch64.

2.2.1 The Boot Screen on Machines Equipped with Traditional BIOS

The boot screen displays several options for the installation procedure. *Boot from Hard Disk* boots the installed system and is selected by default, because the CD is often left in the drive. Select one of the other options with the arrow keys and press Enter to boot it. The relevant options are:

Installation

The normal installation mode. All modern hardware functions are enabled. In case the installation fails, see F5 *Kernel* for boot parameters that disable potentially problematic functions.

Upgrade

Perform a system upgrade. For more information refer to *Chapter 13, Upgrading the System and System Changes*.

More > Rescue System

Starts a minimal Linux system without a graphical user interface. For more information, see *Section 17.5.2, "Using the Rescue System"*. This option is not available on live CDs.

More > Boot Linux System

Boot a Linux system that is already installed. You will be asked from which partition to boot the system.

More > Check Installation Media

This option is only available when you install from media created from downloaded ISOs. In this case it is recommended to check the integrity of the installation medium. This option starts the installation system before automatically checking the media. In case the check was successful, the normal installation routine starts. If a corrupt media is detected, the installation routine aborts. Replace the broken medium and restart the installation process.

More > Memory Test

Tests your system RAM using repeated read and write cycles. Terminate the test by rebooting. For more information, see *Section 4.4, "Boot Failure"*.

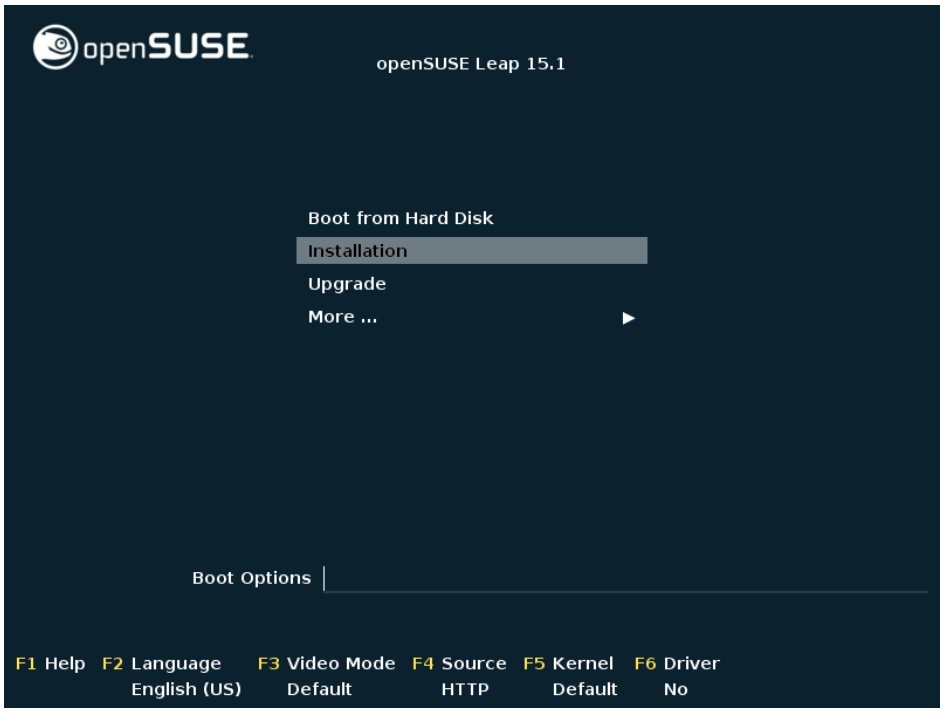


FIGURE 2.1: THE BOOT SCREEN ON MACHINES WITH A TRADITIONAL BIOS

Use the function keys shown at the bottom of the screen to change the language, screen resolution, installation source or to add an additional driver from your hardware vendor:

F1 *Help*

Get context-sensitive help for the active element of the boot screen. Use the arrow keys to navigate, Enter to follow a link, and Esc to leave the help screen.

F2 *Language*

Select the display language and a corresponding keyboard layout for the installation. The default language is English (US).

F3 *Video Mode*

Select various graphical display modes for the installation. By *Default* the video resolution is automatically determined using KMS ("Kernel Mode Setting"). If this setting does not work on your system, choose *No KMS* and, optionally, specify `vga=ask` on the boot command line to get prompted for the video resolution. Choose *Text Mode* if the graphical installation causes problems.

F4 *Source*

Normally, the installation is performed from the inserted installation medium. Here, select other sources, like FTP or NFS servers. If the installation is deployed on a network with an SLP server, select an installation source available on the server with this option.

F5 ***Kernel***

If you encounter problems with the regular installation, this menu offers to disable a few potentially problematic functions. If your hardware does not support ACPI (advanced configuration and power interface) select *No ACPI* to install without ACPI support. *No local APIC* disables support for APIC (Advanced Programmable Interrupt Controllers) which may cause problems with some hardware. *Safe Settings* boots the system with the DMA mode (for CD/DVD-ROM drives) and power management functions disabled.

If you are not sure, try the following options first: *Installation—ACPI Disabled* or *Installation —Safe Settings*. Experts can also use the command line (*Boot Options*) to enter or change kernel parameters.

F6 ***Driver***

Press this key to notify the system that you have an optional driver update for openSUSE Leap. With *File* or *URL*, load drivers directly before the installation starts. If you select *Yes*, you are prompted to insert the update disk at the appropriate point in the installation process.

2.2.2 The Boot Screen on Machines Equipped with UEFI

UEFI (Unified Extensible Firmware Interface) is a new industry standard which replaces and extends the traditional BIOS. The latest UEFI implementations contain the “Secure Boot” extension, which prevents booting malicious code by only allowing signed boot loaders to be executed. See *Book “*Reference*”, Chapter 14 “UEFI (Unified Extensible Firmware Interface)”* for more information.

The boot manager GRUB 2, used to boot machines with a traditional BIOS, does not support UEFI, therefore GRUB 2 is replaced with GRUB 2 for EFI. If Secure Boot is enabled, YaST will automatically select GRUB 2 for EFI for installation. From an administrative and user perspective, both boot manager implementations behave the same and are called `GRUB 2` in the following.

Tip: Using Additional Drivers with Secure Boot

When installing with Secure Boot enabled, you cannot load drivers that are not shipped with openSUSE Leap. This is also true of drivers shipped via SolidDriver, because their signing key is not trusted by default.

To load drivers not shipped with openSUSE Leap, do either of the following:

- Before the installation, add the needed keys to the firmware database via firmware/system management tools.
- Use a bootable ISO that will enroll the needed keys in the MOK list on the first boot.

For more information, see *Book "Reference", Chapter 14 "UEFI (Unified Extensible Firmware Interface)", Section 14.1 "Secure Boot"*.

The boot screen displays several options for the installation procedure. Change the selected option with the arrow keys and press Enter to boot it. The relevant options are:

Installation

The normal installation mode. All modern hardware functions are enabled. In case the installation fails, see F5 *Kernel* for boot parameters that disable potentially problematic functions.

Upgrade

Perform a system upgrade. For more information refer to *Chapter 13, Upgrading the System and System Changes*.

More › Rescue System

Starts a minimal Linux system without a graphical user interface. For more information, see *Section 17.5.2, "Using the Rescue System"*. This option is not available on Live CDs.

More › Boot Linux System

Boot a Linux system that is already installed. You will be asked from which partition to boot the system.

More › Check Installation Media

This option is only available when you install from media created from downloaded ISOs. In this case it is recommended to check the integrity of the installation medium. This option starts the installation system before automatically checking the media. In case the check was successful, the normal installation routine starts. If a corrupt media is detected, the installation routine aborts.

FIGURE 2.2: THE BOOT SCREEN ON MACHINES WITH UEFI

GRUB 2 for EFI on openSUSE Leap does not support a boot prompt or function keys for adding boot parameters. By default, the installation will be started with American English and the boot media as the installation source. A DHCP lookup will be performed to configure the network. To change these defaults or to add boot parameters you need to edit the respective boot entry. Highlight it using the arrow keys and press E. See the on-screen help for editing hints (note that only an English keyboard is available now). The *Installation* entry will look similar to the following:

```
setparams 'Installation'

   set gfxpayload=keep
   echo 'Loading kernel ...'
   linuxefi /boot/x86_64/loader/linux splash=silent
   echo 'Loading initial ramdisk ...'
   initrdefi /boot/x86_64/loader/initrd
```

Add space-separated parameters to the end of the line starting with `linuxefi`. To boot the edited entry, press F10. If you access the machine via serial console, press Esc–0. A complete list of parameters is available at http://en.opensuse.org/Linuxrc.

2.3 List of Important Boot Parameters

This section contains a selection of important boot parameters.

2.3.1 General Boot Parameters

`autoyast=` *URL*

The `autoyast` parameter specifies the location of the `autoinst.xml` control file for automatic installation.

`manual=<0|1>`

The `manual` parameter controls whether the other parameters are only default values that still must be acknowledged by the user. Set this parameter to `0` if all values should be accepted and no questions asked. Setting `autoyast` implies setting `manual` to `0`.

`Info=` *URL*

Specifies a location for a file from which to read additional options.

`upgrade=<0|1>`

To upgrade openSUSE Leap, specify `Upgrade=1`.

`dud=` *URL*

Load driver updates from *URL*.

Set `dud=ftp://ftp.example.com/PATH_TO_DRIVER` or `dud=http://www.example.com/PATH_TO_DRIVER` to load drivers from a URL. When `dud=1` you will be asked for the URL during boot.

`language=` *LANGUAGE*

Set the installation language. Some supported values are `cs_CZ`, `de_DE`, `es_ES`, `fr_FR`, `ja_JP`, `pt_BR`, `pt_PT`, `ru_RU`, `zh_CN`, and `zh_TW`.

`acpi=off`

Disable ACPI support.

`noapic`

No logical APIC.

`nomodeset`

Disable KMS.

`textmode=1`

Start installer in text mode.

`console=` *`SERIAL_DEVICE`*`[,`*`MODE`*`]`

`SERIAL_DEVICE` can be an actual serial or parallel device (for example `ttyS0`) or a virtual terminal (for example `tty1`). *`MODE`* is the baud rate, parity and stop bit (for example `9600n8`). The default for this setting is set by the mainboard firmware. If you do not see output on your monitor, try setting `console=tty1`. It is possible to define multiple devices.

2.3.2 Configuring the Network Interface

Important: Configuring the Network Interface

The settings discussed in this section apply only to the network interface used during installation. Configure additional network interfaces in the installed system by following the instructions given in *Book "*Reference*", Chapter 13 "Basic Networking", Section 13.6 "Configuring a Network Connection Manually"*.

The network will only be configured if it is required during the installation. To force the network to be configured, use the `netsetup` parameter.

`netsetup=`*`VALUE`*

`netsetup=dhcp` forces a configuration via DHCP. Set `netsetup=-dhcp` when configuring the network with the boot parameters `hostip`, `gateway` and `nameserver`. With the option `netsetup=hostip,netmask,gateway,nameserver` the installer asks for the network settings during boot.

`ifcfg=`*`INTERFACE`*`[.`*`VLAN`*`]=`*`SETTINGS`*

`INTERFACE` can be `*` to match all interfaces or, for example, `eth*` to match all interfaces that start with `eth`. It is also possible to use MAC addresses as values.
Optionally, a VLAN can be set behind the interface name, separated by a period.

If *SETTINGS* is `dhcp`, all matching interfaces will be configured with DHCP. It is possible to set static parameters. With static parameters, only the first matching interface will be configured. The syntax for the static configuration is:

```
ifcfg=*="IPS_NETMASK,GATEWAYS,NAMESERVERS,DOMAINS"
```

Each comma separated value can in turn contain a list of space character separated values. *IPS_NETMASK* is in the *CIDR notation*, for example `10.0.0.1/24`. The quotes are only needed when using space character separated lists. Example with two name servers:

```
ifcfg=*="10.0.0.10/24,10.0.0.1,10.0.0.1 10.0.0.2,example.com"
```

`hostname=host.example.com`

Enter the fully qualified host name.

`domain=example.com`

Domain search path for DNS. Allows you to use short host names instead of fully qualified ones.

`hostip=192.168.1.2[/24]`

Enter the IP address of the interface to configure. The IP can contain the subnet mask, for example `hostip=192.168.1.2/24`. This setting is only evaluated if the network is required during the installation.

`gateway=192.168.1.3`

Specify the gateway to use. This setting is only evaluated if the network is required during the installation.

`nameserver=192.168.1.4`

Specify the DNS server in charge. This setting is only evaluated if the network is required during the installation.

`domain=example.com`

Domain search path. This setting is only evaluated if the network is required during the installation.

2.3.3 Specifying the Installation Source

If you are not using the DVD for installation, specify an alternative installation source.

`install=SOURCE`

Specify the location of the installation source to use. Possible protocols are `cd`, `hd`, `slp`, `nfs`, `smb` (Samba/CIFS), `ftp`, `tftp` `http`, and `https`. Not all source types are available on all platforms.
The default option is `cd`.
If an `ftp`, `tftp` or `smb` URL is given, specify the user name and password with the URL. These parameters are optional and anonymous or guest login is assumed if they are not given. Example:

```
install=ftp://USER:PASSWORD@SERVER/DIRECTORY/DVD1/
```

To install over an encrypted connection, use an `https` URL. If the certificate cannot be verified, use the `sslcerts=0` boot parameter to disable certificate checking.
In case of a Samba or CIFS installation, you can also specify the domain that should be used:

```
install=smb://WORKDOMAIN;USER:PASSWORD@SERVER/DIRECTORY/DVD1/
```

To use `cd`, `hd` or `slp`, set them as the following example:

```
install=cd:/
install=hd:/?device=sda/PATH_TO_ISO
install=slp:/
```

2.3.4 Specifying Remote Access

Only one of the different remote control methods should be specified at a time. The different methods are: SSH, VNC, remote X server.

`display_ip=` *`IP_ADDRESS`*

`Display_IP` causes the installing system to try to connect to an X server at the given address.

> **Important: X Authentication Mechanism**
>
> The direct installation with the X Window System relies on a primitive authentication mechanism based on host names. This mechanism is disabled on current openSUSE Leap versions. Installation with SSH or VNC is preferred.

`vnc=1`

Enables a VNC server during the installation.

`vncpassword=` *`PASSWORD`*

Sets the password for the VNC server.

`ssh=1`

`ssh` enables SSH installation.

`ssh.password=` *`PASSWORD`*

Specifies an SSH password for the root user during installation.

2.4 Advanced Setups

To configure access to a local RMT or `supportconfig` server for the installation, you can specify boot parameters to set up these services during installation. The same applies if you need IPv6 support during the installation.

2.4.1 Using IPv6 for the Installation

By default you can only assign IPv4 network addresses to your machine. To enable IPv6 during installation, enter one of the following parameters at the boot prompt:

Accept IPv4 and IPv6

```
ipv6=1
```

Accept IPv6 only

```
ipv6only=1
```

2.4.2 Using a Proxy for the Installation

In networks enforcing the usage of a proxy server for accessing remote Web sites, registration during installation is only possible when configuring a proxy server.

To use a proxy during the installation, press F4 on the boot screen and set the required parameters in the *HTTP Proxy* dialog. Alternatively provide the kernel parameter `proxy` at the boot prompt:

```
proxy=http://USER:PASSWORD@proxy.example.com:PORT
```

Specifying *USER* and *PASSWORD* is optional—if the server allows anonymous access, the following data is sufficient: `http://proxy.example.com:`*PORT*.

2.4.3 Enabling SELinux Support

Enabling SELinux upon installation start-up enables you to configure it after the installation has been finished without having to reboot. Use the following parameters:

```
security=selinux selinux=1
```

2.4.4 Enabling the Installer Self-Update

During installation and upgrade, YaST can update itself as described in *Section 3.2, "Installer Self-Update"* to solve potential bugs discovered after release. The `self_update` parameter can be used to modify the behavior of this feature.

To enable the installer self-update, set the parameter to `1`:

```
self_update=1
```

To use a user-defined repository, specify a URL:

```
self_update=https://updates.example.com/
```

2.4.5 Scale User Interface for High DPI

If your screen uses a very high DPI, use the boot parameter `QT_AUTO_SCREEN_SCALE_FACTOR`. This scales font and user interface elements to the screen DPI.

```
QT_AUTO_SCREEN_SCALE_FACTOR=1
```

2.4.6 Using CPU Mitigations

The boot parameter `mitigations` lets you control mitigation options for side-channel attacks on affected CPUs. Its possible values are:

`auto`. Enables all mitigations required for your CPU model, but does not protect against cross-CPU thread attacks. This setting may impact performance to some degree, depending on the workload.

`nosmt`. Provides the full set of available security mitigations. Enables all mitigations required for your CPU model. In addition, it disables Simultaneous Multithreading (SMT) to avoid side-channel attacks across multiple CPU threads. This setting may further impact performance, depending on the workload.

`off`. Disables all mitigations. Side-channel attacks against your CPU are possible, depending on the CPU model. This setting has no impact on performance.

Each value comes with a set of specific parameters, depending on the CPU architecture, the kernel version, and on the vulnerabilities that need to be mitigated. Refer to the kernel documentation for details.

2.5 More Information

You can find more information about boot parameters in the openSUSE wiki at https://en.opensuse.org/SDB:Linuxrc#Parameter_Reference.

3 Installation Steps

This chapter describes the procedure in which the data for openSUSE Leap is copied to the target device. Some basic configuration parameters for the newly installed system are set during the procedure. A graphical user interface will guide you through the installation. The text mode installation has the same steps and only looks different. For information about performing non-interactive automated installations, see *Book "AutoYaST Guide"*.

If you are a first-time user of openSUSE Leap, you should follow the default YaST proposals in most parts, but you can also adjust the settings as described here to fine-tune your system according to your preferences. Help for each installation step is provided by clicking *Help*.

Tip: Installation Without a Mouse

If the installer does not detect your mouse correctly, use [→|] for navigation, arrow keys to scroll, and [Enter] to confirm a selection. Various buttons or selection fields contain a letter with an underscore. Use [Alt]–[Letter] to select a button or a selection directly instead of navigating there with [→|].

3.1 Overview

This section provides an overview of all installation steps. Each step contains a link to a more detailed description.

1. Before the installation starts, the installer can update itself. For details, see *Section 3.2, "Installer Self-Update"*.
2. The actual installation starts with choosing the language and accepting the license agreement. For details, see *Section 3.3, " Language, Keyboard, and License Agreement "*.
3. Configure the network. This is only required when you need network access during the installation and the automatic network configuration via DHCP failed. If the automatic network configuration succeeded, this step is skipped. For details, see *Section 3.4, "Network Settings"*.

4. Configure the online repositories. By adding official openSUSE repositories, you get access to more software and get the latest security updates already during installation. For details, see *Section 3.5, "Online Repositories"*. This step is optional and can be skipped.

5. Select a desktop or a role for your system. Among other things, this defines the default list of packages to install and makes a suggestion for partitioning the hard disks. For details, see *Section 3.6, "System Role"*.

6. Partition the hard disks of your system. For details, see *Section 3.7, "Partitioning"*.

7. Choose a time zone. For details, see *Section 3.8, "Clock and Time Zone"*.

8. Create a user. For details, see *Section 3.9, "Create New User"*.

9. Optionally, set a different password for the system administrator `root`. For details, see *Section 3.10, "Authentication for the System Administrator "root""*.

10. In a final step, the installer presents an overview of all settings. If required, you can change them. For details, see *Section 3.11, "Installation Settings"*.

11. The installer copies all required data and informs you about the progress. For details, see *Section 3.12, "Performing the Installation"*.

3.2 Installer Self-Update

During the installation and upgrade process, YaST can update itself to solve bugs in the installer that were discovered after the release. This functionality is enabled by default; to disable it, set the boot parameter `self_update` to `0`. For more information, see *Section 2.4.4, "Enabling the Installer Self-Update"*.

Important: Networking during Self-Update

To download installer updates, YaST needs network access. By default, it tries to use DHCP on all network interfaces. If there is a DHCP server in the network, it will work automatically.

If you need a static IP setup, you can use the `ifcfg` boot argument. For more details, see the linuxrc documentation at https://en.opensuse.org/Linuxrc.

Tip: Language Selection

The installer self-update is executed before the language selection step. This means that progress and errors which happen during this process are displayed in English by default.

To use another language for this part of the installer, use the `language` boot parameter if available for your architecture, for example, `language=de_DE`. On machines equipped with a traditional BIOS, alternatively, press F2 in the boot menu and select the language from the list.

Although this feature was designed to run without user intervention, it is worth knowing how it works. If you are not interested, you can jump directly to *Section 3.3, "Language, Keyboard, and License Agreement"* and skip the rest of this section.

3.2.1 Self-Update Process

The process can be broken down into two different parts:

1. Determine the update repository location.
2. Download and apply the updates to the installation system.

3.2.1.1 Determining the Update Repository Location

Installer Self-Updates are distributed as regular RPM packages via a dedicated repository, so the first step is to find out the repository URL.

Important: Installer Self-Update Repository Only

No matter which of the following options you use, only the installer self-update repository URL is expected, for example:

```
self_update=https://www.example.com/my_installer_updates/
```

Do not supply any other repository URL—for example the URL of the software update repository.

YaST will try the following sources of information:

1. The `self_update` boot parameter. (For more details, see *Section 2.4.4, "Enabling the Installer Self-Update"*.) If you specify a URL, it will take precedence over any other method.
2. The `/general/self_update_url` profile element in case you are using AutoYaST.
3. If none of the previous attempts worked, the fallback URL (defined in the installation media) will be used.

3.2.1.2 Downloading and Applying the Updates

When the updates repository is determined, YaST will check whether an update is available. If so, all the updates will be downloaded and applied to the installation system.

Finally, YaST will be restarted to load the new version and the welcome screen will be shown. If no updates were available, the installation will continue without restarting YaST.

Note: Update Integrity

Update signatures will be checked to ensure integrity and authorship. If a signature is missing or invalid, you will be asked whether you want to apply the update.

3.2.1.3 Temporary Self-Update Add-on Repository

Some packages distributed in the self-update repository provide additional data for the installer, like the installation defaults, system role definitions and similar. If the installer finds such packages in the self-update repository, a local temporary repository is created, to which those packages are copied. They are used during the installation process, but at the end of the installation, the temporary local repository is removed. Its packages are *not* installed onto the target system.

This additional repository is not displayed in the list of add-on products, but during installation it may still be visible as `SelfUpdate0` repository in the package management.

3.2.2 Custom Self-Update Repositories

YaST can use a user-defined repository instead of the official one by specifying a URL through the `self_update` boot parameter. However, the following points should be considered:

- Only HTTP/HTTPS and FTP repositories are supported.
- Only RPM-MD repositories are supported (required by RMT).
- Packages are not installed in the usual way: They are uncompressed only and no scripts are executed.
- No dependency checks are performed. Packages are installed in alphabetical order.
- Files from the packages override the files from the original installation media. This means that the update packages might not need to contain all files, only files that have changed. Unchanged files are omitted to save memory and download bandwidth.

Note: Only One Repository

Currently, it is not possible to use more than one repository as source for installer self-updates.

3.3 Language, Keyboard, and License Agreement

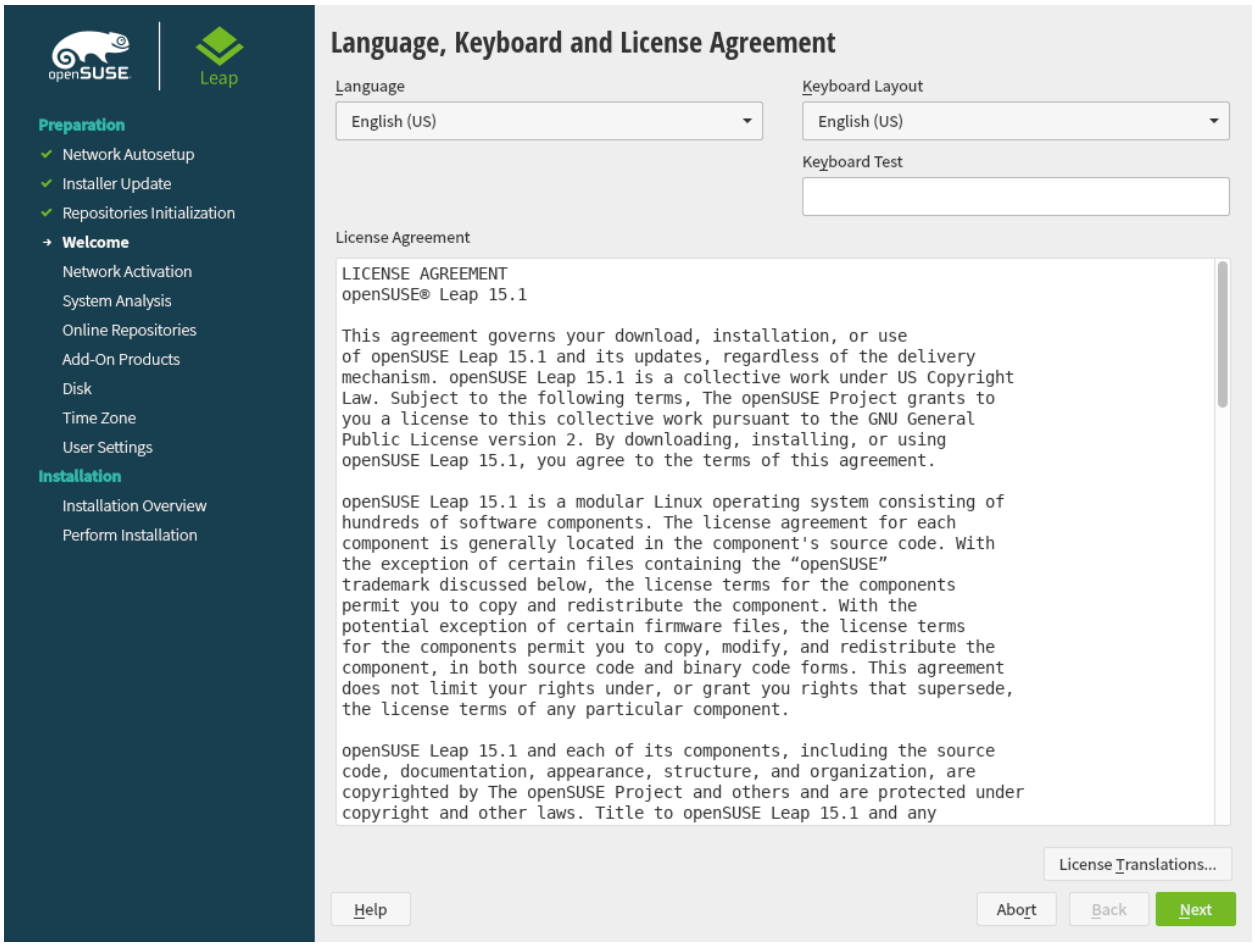


FIGURE 3.1: LANGUAGE, KEYBOARD, AND LICENSE AGREEMENT

The *Language* and *Keyboard Layout* settings are initialized with the language you chose on the boot screen. If you did not change the default, it will be English (US). Change the settings here, if necessary.

Changing the language will automatically preselect a corresponding keyboard layout. Override this proposal by selecting a different keyboard layout from the drop-down box. Use the *Keyboard Test* text box to test the layout. The language selected here is also used to assume a time zone for the system clock. This setting can be modified later in the installed system as described in *Chapter 6, Changing Language and Country Settings with YaST*.

Read the license agreement. It is presented in the language you have. Translations are available via the *License Language* drop-down box. Proceed with *Next* if you agree to the terms and conditions. If you do not agree, click *Abort* to terminate the installation.

3.4 Network Settings

After booting into the installation, the installation routine is set up. During this setup, an attempt to configure at least one network interface with DHCP is made. In case this attempt has failed, the *Network Settings* dialog launches now.

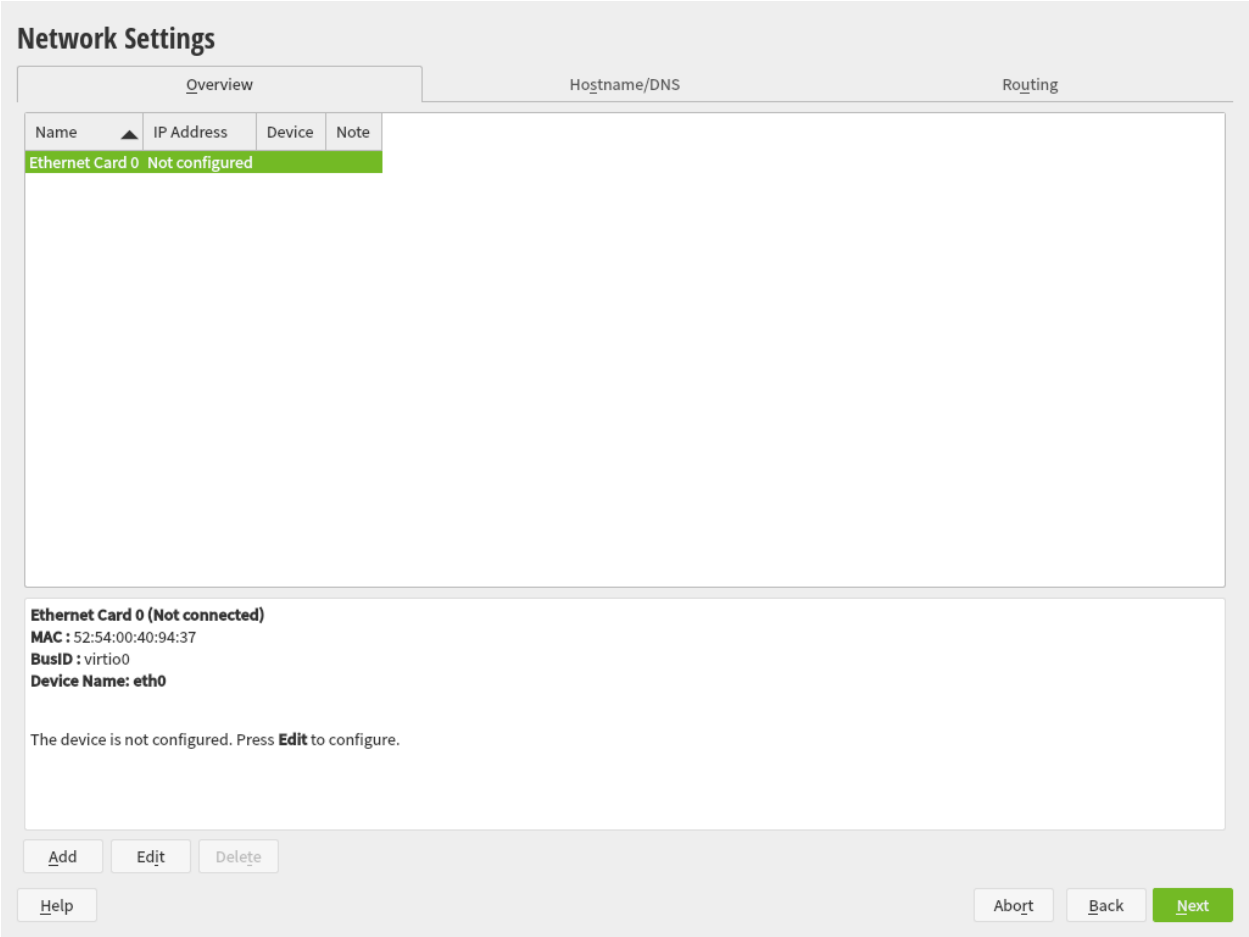


FIGURE 3.2: NETWORK SETTINGS

Choose a network interface from the list and click *Edit* to change its settings. Use the tabs to configure DNS and routing. See *Book "Reference", Chapter 13 "Basic Networking", Section 13.4 "Configuring a Network Connection with YaST"* for more details.

In case DHCP was successfully configured during installation setup, you can also access this dialog by clicking *Network Configuration* at the the *Installation Settings* step. It lets you change the automatically provided settings.

Note: Network Configuration with Boot Parameters

If at least one network interface has been configured via boot parameters (see *Section 2.3.2, "Configuring the Network Interface"*), automatic DHCP configuration is disabled and the boot parameter configuration is imported and used.

Tip: Accessing Network Storage or Local RAID

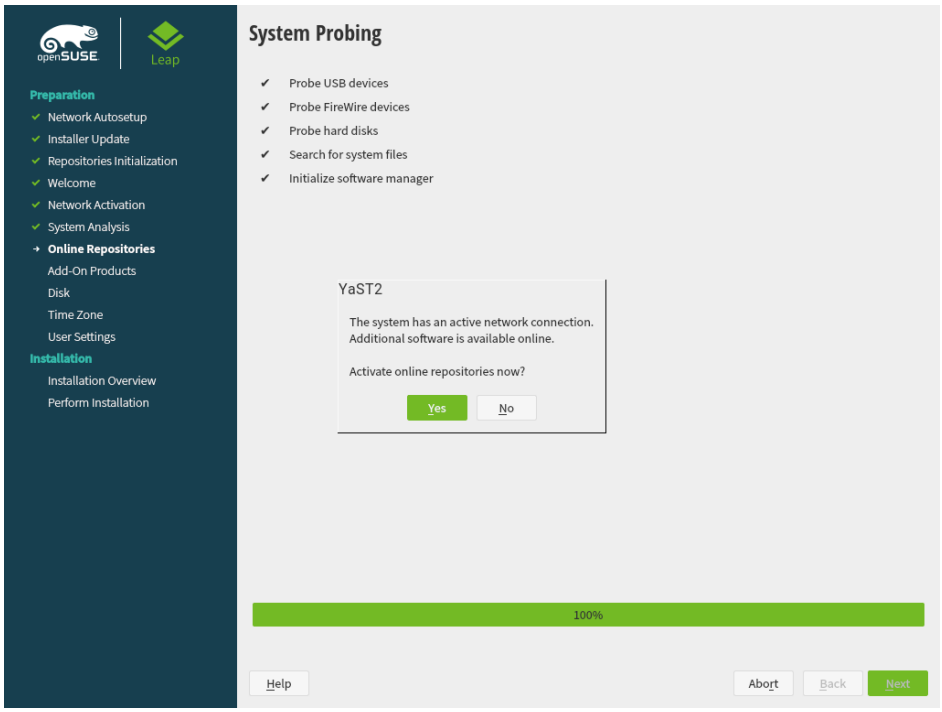
To access a SAN or a local RAID during the installation, you can use the libstorage command line client for this purpose:

1. Switch to a console with Ctrl–Alt–F2.
2. Install the libstoragemgmt extension by running **extend libstoragemgmt**.
3. Now you have access to the **lsmcli** command. For more information, run **lsmcli --help**.
4. To return to the installer, press Alt–F7

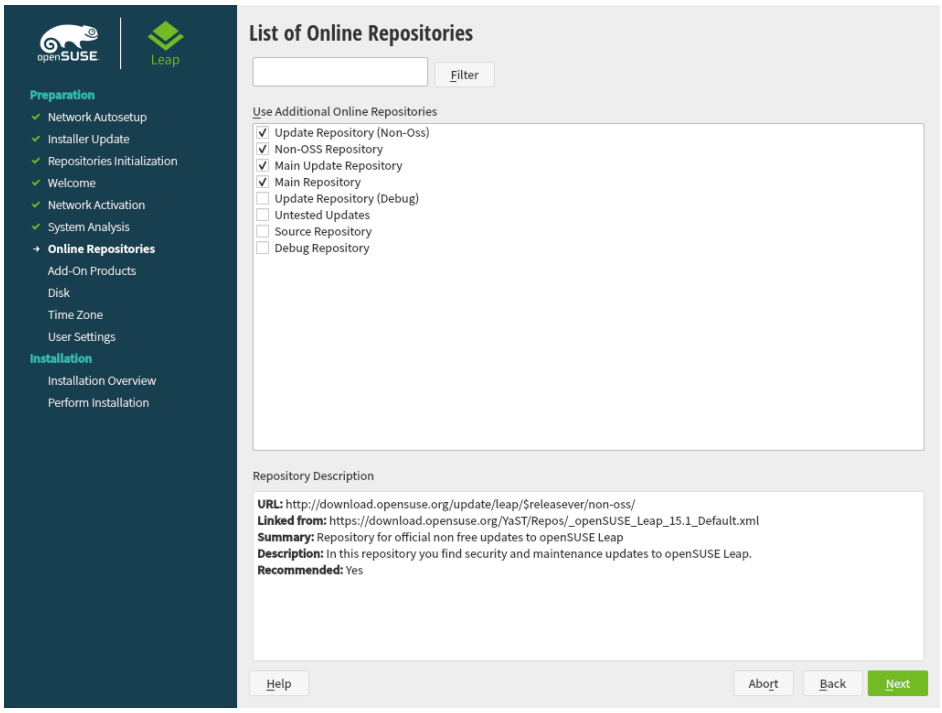
Supported are Netapp Ontap, all SMI-S compatible SAN providers, and LSI MegaRAID.

3.5 Online Repositories

A system analysis is performed, where the installer probes for storage devices, and tries to find other installed systems. If a network connection with Internet access is available, you will be asked to activate the online repositories. Answer with *Yes* to proceed. In case you do not have Internet access, this step will be skipped.



The online repositories are official openSUSE package sources. They not only offer additional packages not included on the installation media, but also the update repositories containing security and bug fixes. Using the default selection is recommended. Add at least the *Main Update Repository*, because it makes sure the system is installed with the latest security patches.



You have the following choices:

- The *Main Repository (OSS)* contains open source software (OSS). Compared to the DVD installation media, it contains many additional software packages, among them many additional desktop systems.
- The *Main Update Repository* contains security updates and fixes for packages from the *Main Repository (OSS)* and the DVD installation media. Choosing this repository is recommended for all installation scenarios.
- The *Main Repository (Non-OSS)* contains packages with a proprietary software license. Choosing it is not required for installing a custom desktop system.
- Choosing *Main Update Repository (Non-OSS)* is recommended when also having chosen the *Main Repository (Non-OSS)*. It contains the respective updates and security fixes.
- All other repositories are intended for experienced users and developers. Click on a repository name to get more information.

Confirm your selection with *Next*. Depending on your choice, you need to confirm one or more license agreements. Do so by choosing *Next* until you proceed to the *System Role* screen. Now choose *Next* to proceed.

3.6 System Role

openSUSE Leap supports a broad range of features. To simplify the installation, the installer offers predefined use cases which adjust the system to be installed so it is tailored for the selected scenario. Currently this affects the package set and the suggested partitioning scheme.

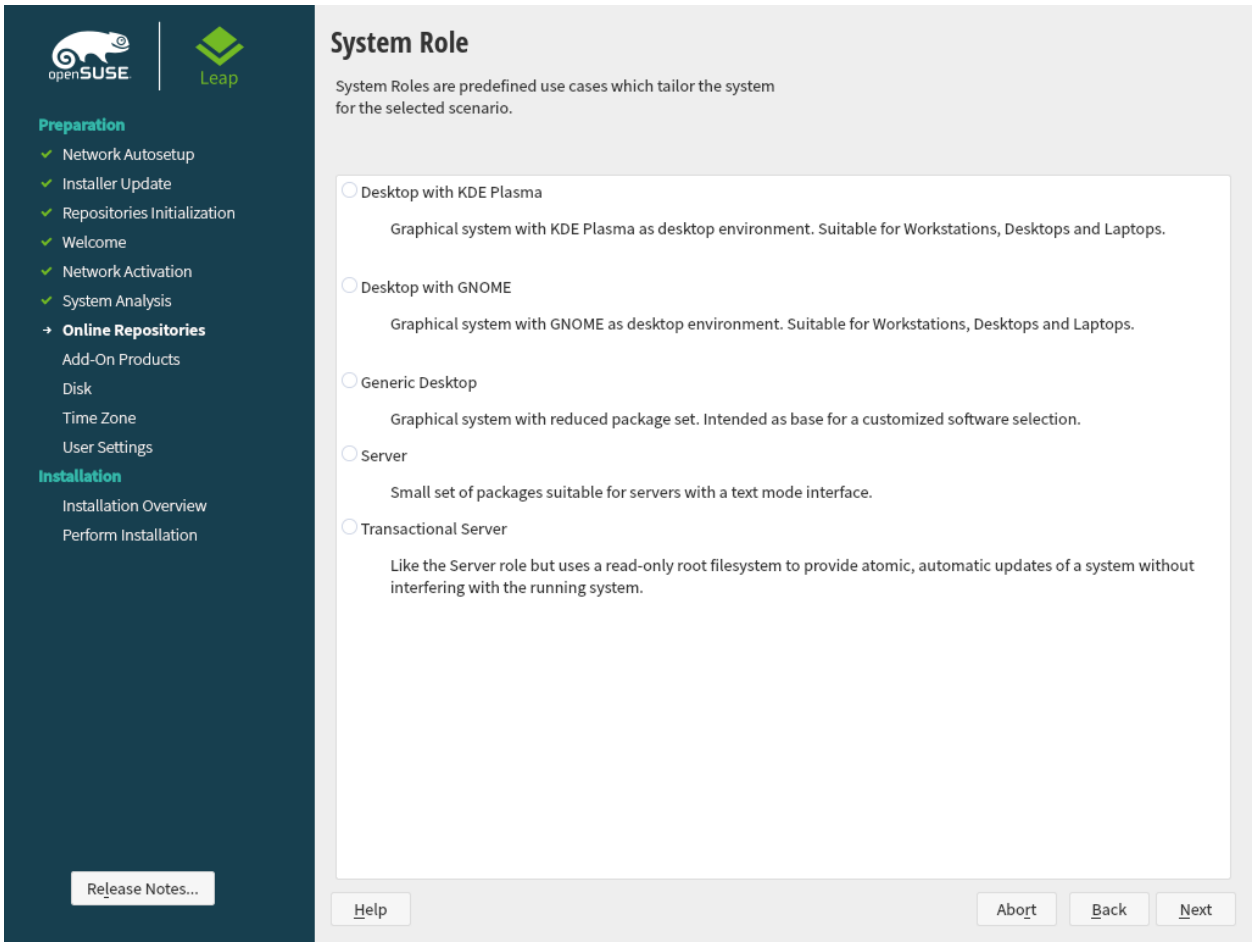


FIGURE 3.3: SYSTEM ROLE

Choose the *System Role* that meets your requirements best. The availability of system roles depends on your selection of modules and extensions. The following system roles are available with the default selection:

Desktop with KDE Plasma

A powerful desktop environment with a complete PIM suite (mail, calendar, tasks, notes, and feeds), widgets runnning on the desktop and many more features. If you are familiar with Windows, KDE is the recommended choice. For more information see https://kde.org/.

Desktop with GNOME

A desktop environment offering an alternative, innovative user experience. GNOME was designed with usability and productivity in mind. For more information see https://www.gnome.org/.

Generic Desktop

In case you prefer an alterntive to the KDE or GNOME desktops, choose this option. You will be able to choose between the following alternatives later in the installation process by selecting *Software* in the *Installation Settings dialog*:

Enlightenment (https://www.enlightenment.org/)
LXDE (https://lxde.org/)
LXQT (https://lxqt.org/)
MATE (https://mate-desktop.org/)
XFCE (https://xfce.org/)
Note that when installing from the DVD all these desktop systems except XFCE are only available when having enabled the MAIN Repository (OSS) in the *Online Repositories* step. You can still enable this repository now by using the *Back* button until you reach the welcome screen. From there, choose *Next* and then agree to add online repositories.

Server

If setting up a server, you probably do not need a graphical user interface and desktop applications such as an office suite. This option gives you a reduced set of packages suitable for servers.

Transactional Server

Similar to the server role, but with a read-only root partition and transactional updates. This selection also is a prerequisite for setting up openSUSE Kubic. See https://kubic.opensuse.org/blog/2018-04-04-transactionalupdates/ for more information on transactional updates.

3.7 Partitioning

3.7.1 Important Information

Warning: Read this Section Carefully

Read this section carefully before continuing with *Section 3.7.2, "Suggested Partitioning"*.

Custom Partitioning on UEFI Machines

A UEFI machine *requires* an EFI system partition that must be mounted to `/boot/efi`. This partition must be formatted with the `FAT32` file system.

If an EFI system partition is already present on your system (for example from a previous Windows installation) use it by mounting it to `/boot/efi` without formatting it.

If no EFI system partition is present on your UEFI machine, make sure to create it. The EFI system partition must be a physical partition or RAID 1. Other RAID levels, LVM and other technologies are not supported. It needs to be formatted with the FAT32 file system.

Custom Partitioning and Snapper

If the root partition is larger than 16 GB, openSUSE Leap by default enables file system snapshots.

openSUSE Leap uses Snapper together with Btrfs for this feature. Btrfs needs to be set up with snapshots enabled for the root partition.

If the disk is smaller than 16 GB, all Snapper features and automatic snapshots are disabled to prevent the system partition `/` from running out of space.

Being able to create system snapshots that enable rollbacks requires important system directories to be mounted on a single partition, for example `/usr` and `/var`. Only directories that are excluded from snapshots may reside on separate partitions, for example `/usr/local`, `/var/log`, and `/tmp`.

For details, see *Book "Reference", Chapter 3 "System Recovery and Snapshot Management with Snapper"*.

Important: Btrfs Snapshots and Root Partition Size

Snapshots occupy space on their partition. As a rule of thumb, the older a snapshot is, or the bigger the changeset they cover is, the bigger the snapshot. Plus, the more snapshots you keep, the more diskspace you need.

To prevent the root partition running full with snapshot data, you need to make sure it is big enough. In case you do frequent updates or other installations, consider at least 30 GB for the root partition. If you plan to keep snapshots activated for a system upgrade (to be able to roll back), you should consider 40 GB or more.

Btrfs Data Volumes

Using Btrfs for data volumes is supported on openSUSE Leap 15.1. For applications that require Btrfs as a data volume, consider creating a separate file system with quota groups disabled. This is already the default for non-root file systems.

Btrfs on an Encrypted Root Partition

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

Supported Software RAID Volumes

Installing to and booting from existing software RAID volumes is supported for Disk Data Format (DDF) volumes and Intel Matrix Storage Manager (IMSM) volumes. IMSM is also known by the following names:

- Intel Rapid Storage Technology
- Intel Matrix Storage Technology
- Intel Application Accelerator / Intel Application Accelerator RAID Edition

Mount Points for FCoE and iSCSI Devices

FCoE and iSCSI devices will appear asynchronously during the boot process. While the initrd guarantees that those devices are set up correctly for the root file system, there are no such guarantees for any other file systems or mount points like `/usr`. Hence any system mount points like `/usr` or `/var` are not supported. To use those devices, ensure correct synchronization of the respective services and devices.

Handling of Windows Partitions in Proposals

In case the disk selected for the suggested partitioning proposal contains a large Windows FAT or NTFS partition, it will automatically be resized to make room for the openSUSE Leap installation. To avoid data loss it is strongly recommended to

- make sure the partition is not fragmented (run a defragmentation program from Windows prior to the openSUSE Leap installation)
- double-check the suggested size for the Windows partition is big enough
- back up your data prior to the openSUSE Leap installation

To adjust the proposed size of the Windows partition, use the *Expert Partitioner*.

Proposal with a Separate Home Partition

The default proposal no longer suggests to create a separate partition for `/home`. The `/home` directory contains the user's data and personal configuration files. Placing it on a separate directory makes it easier to rebuild the system in the future, or allows to share it with different Linux installations on the same machine.

In case you want to change the proposal to create a separate partition for `/home`, choose *Guided Setup* and click *Next* until you reach the *Filesystem Options* screen. Check *Propose Separate Home Partition*. By default it will be formatted with *XFS*, but you can choose to use a different file system. Close the dialog by clicking *Next* again.

3.7.2 Suggested Partitioning

Define a partition setup for openSUSE Leap in this step.

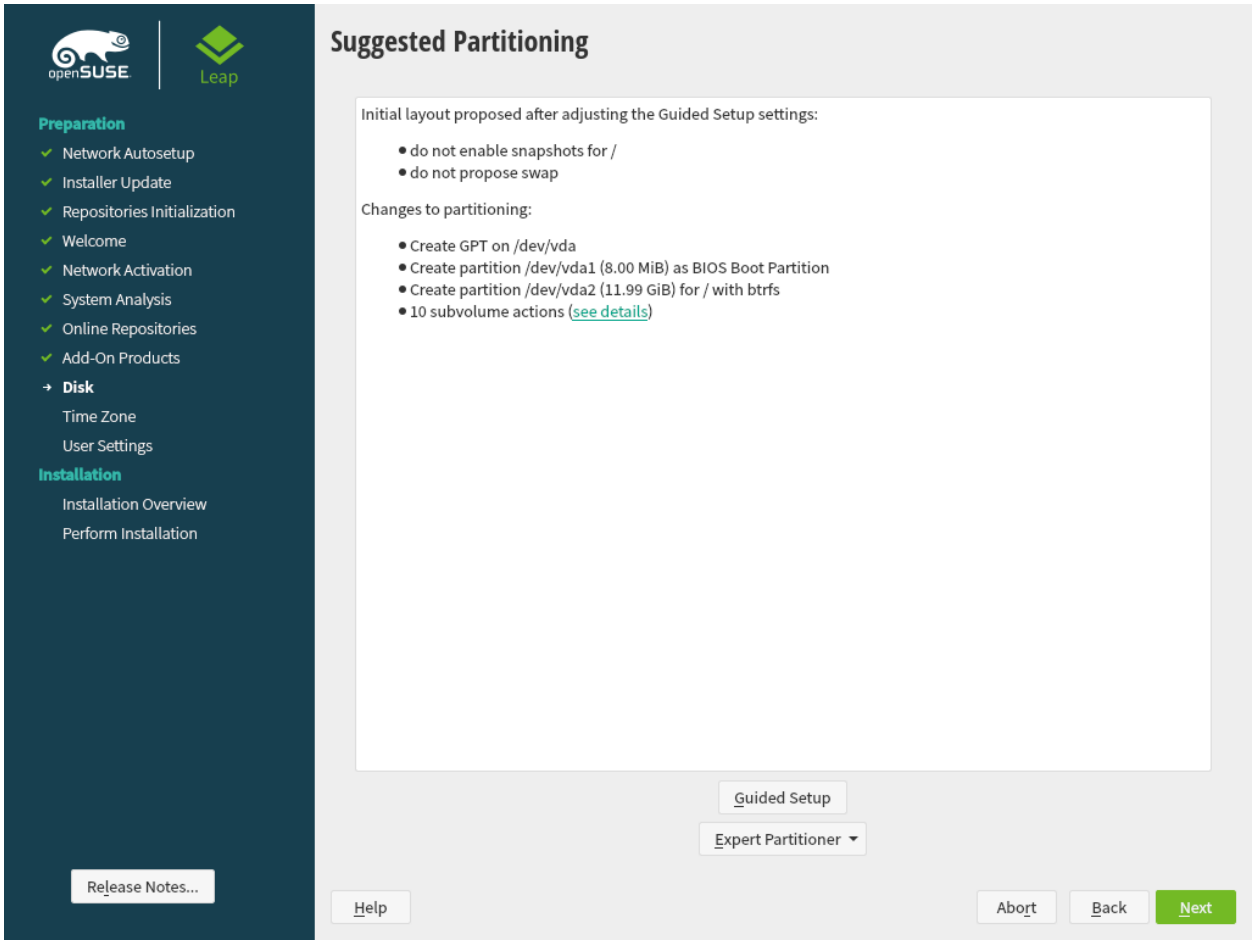


FIGURE 3.4: SUGGESTED PARTITIONING

The installer creates a proposal for one of the available disks containing a root partition formatted with Btrfs and a swap partition. If one or more swap partitions have been detected on the available hard disks, these partitions will be used. You have several options to proceed:

Next

To accept the proposal without any changes, click *Next* to proceed with the installation workflow.

Guided Setup

To adjust the proposal, choose *Guided Setup*. First, choose which hard disks and partitions to use. In the *Partitioning Scheme* screen, you can enable Logical Volume Management (LVM) and activate disk encryption. Afterwards specify the *Filesystem Options*. You can adjust the file system for the root partition and create a separate home and swap partitions. If you plan to suspend your machine, make sure to create a separate swap partition and check *Enlarge to RAM Size for Suspend*. If the root file system format is Btrfs, you can also enable or disable Btrfs snapshots here.

Expert Partitioner

To create a custom partition setup click *Expert Partitioner*. Select either *Start with Current Proposal* if you want start with the suggested disk layout, or *Start with Existing Partitions* to ignore the suggested layout and start with the existing layout on the disk. You can *Add*, *Edit*, *Resize*, or *Delete* partitions.

You can also set up Logical Volumes (LVM), configure software RAID and device mapping (DM), encrypt Partitions, mount NFS shares and manage tmpfs volumes with the Expert Partitioner. To fine-tune settings such as the subvolume and snapshot handling for each Btrfs partition, choose *Btrfs*. For more information about custom partitioning and configuring advanced features, refer to *Book "*Reference*", Chapter 5 "Expert Partitioner", Section 5.1 "Using the Expert Partitioner"*.

3.8 Clock and Time Zone

In this dialog, select your region and time zone. Both are preselected according to the installation language.

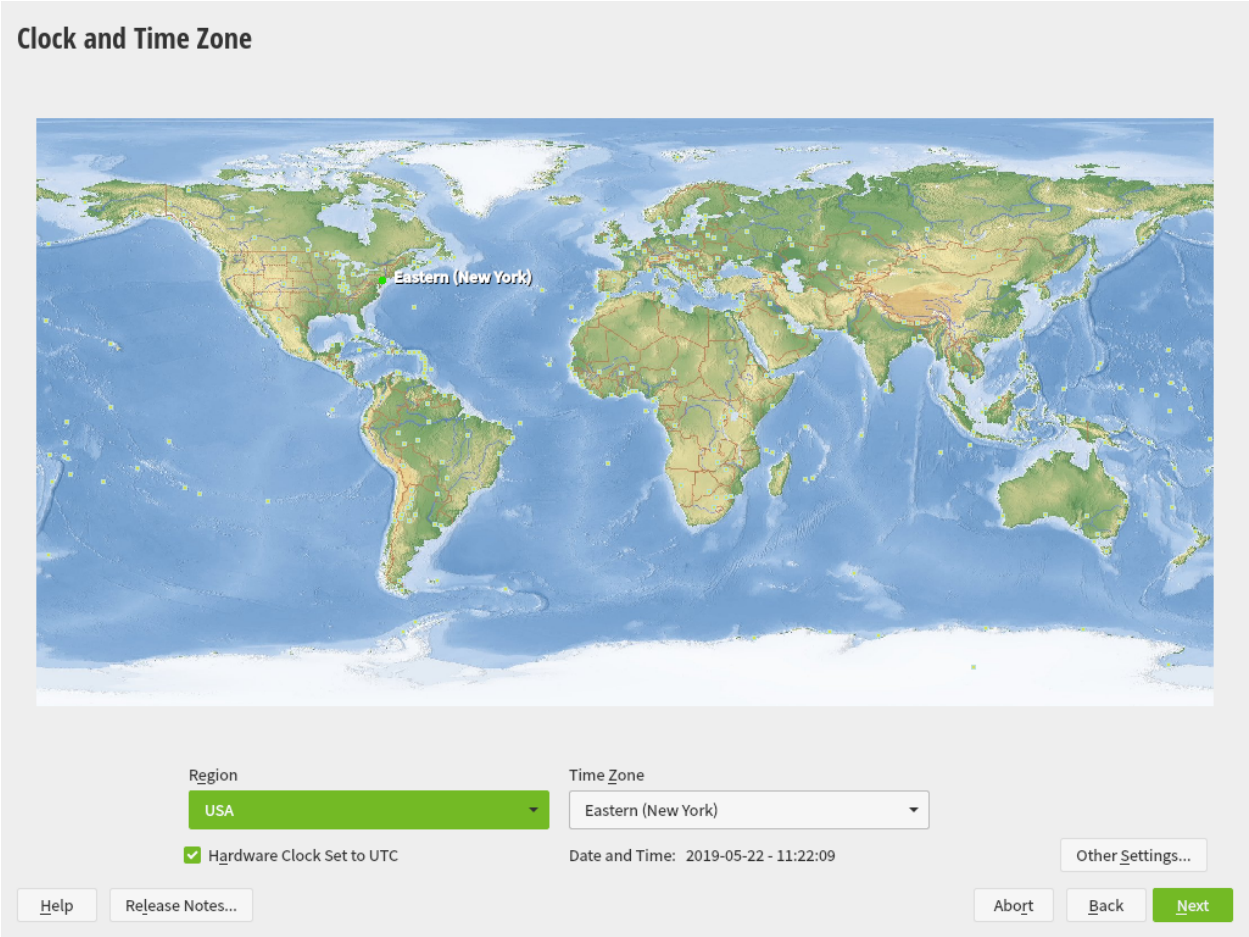


FIGURE 3.5: CLOCK AND TIME ZONE

To change the preselected values, either use the map or the drop-down boxes for *Region* and *Time Zone*. When using the map, point the cursor at the rough direction of your region and left-click to zoom. Now choose your country or region by left-clicking. Right-click to return to the world map.

To set up the clock, choose whether the *Hardware Clock is Set to UTC*. If you run another operating system on your machine, such as Microsoft Windows, it is likely your system uses local time instead. If you run Linux on your machine, set the hardware clock to UTC and have the switch from standard time to daylight saving time performed automatically.

Important: Set the Hardware Clock to UTC

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

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

If a network is already configured, you can configure time synchronization with an NTP server. Click *Other Settings* to either alter the NTP settings or to *Manually* set the time. See *Book "Reference", Chapter 18 "Time Synchronization with NTP"* for more information on configuring the NTP service. When finished, click *Accept* to continue the installation.

If running without NTP configured, consider setting `SYSTOHC=no` (`sysconfig` variable) to avoid saving unsynchronized time into the hardware clock.

3.9 Create New User

Create a local user in this step.

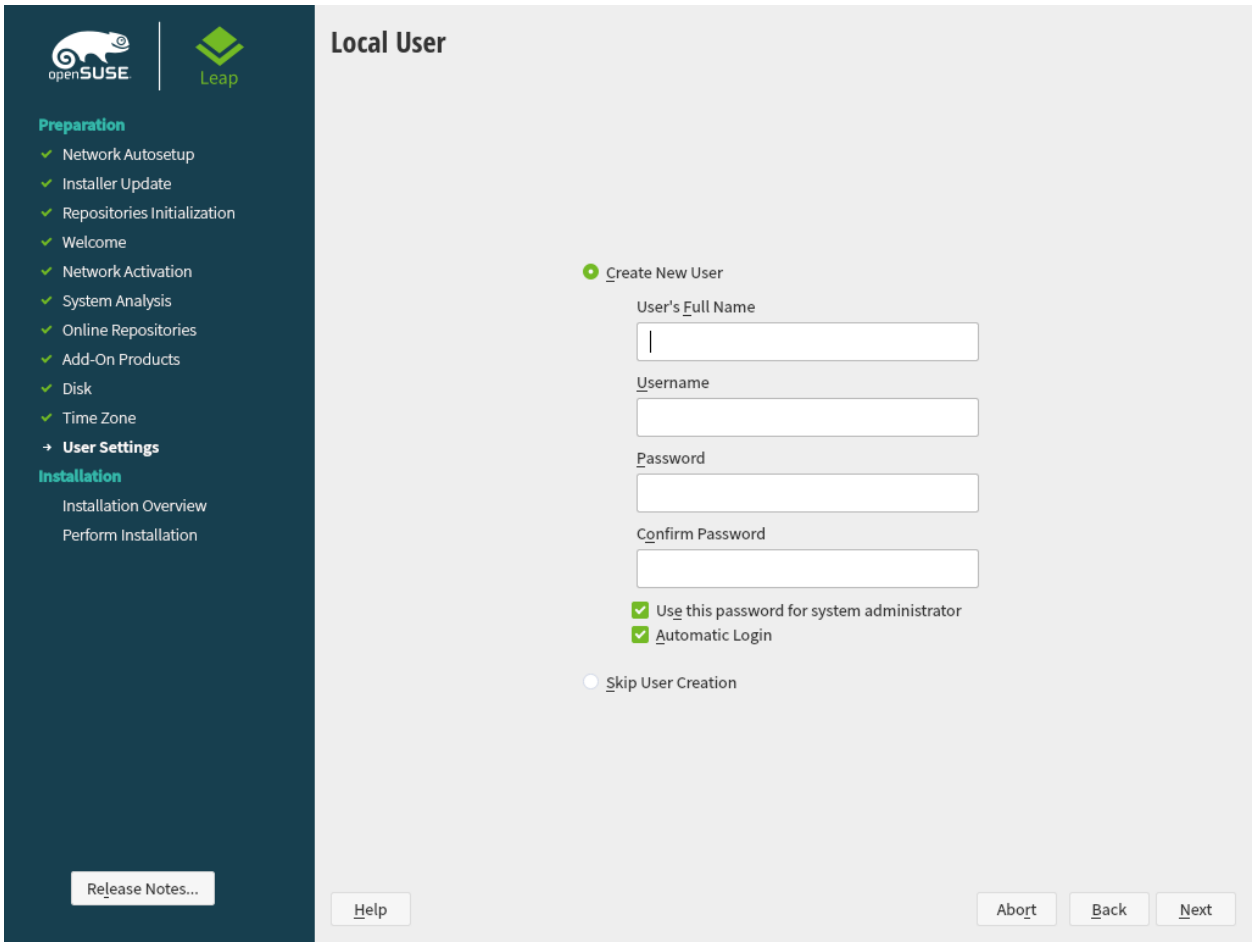


FIGURE 3.6: CREATE NEW USER

After entering the first name and last name, either accept the proposal or specify a new *User name* that will be used to log in. Only use lowercase letters (a-z), digits (0-9) and the characters _. (dot), _- (hyphen) and __ (underscore). Special characters, umlauts and accented characters are not allowed.

Finally, enter a password for the user. Re-enter it for confirmation (to ensure that you did not type something else by mistake). To provide effective security, a password should be at least six characters long and consist of uppercase and lowercase letters, numbers and special characters (7-bit ASCII). Umlauts or accented characters are not allowed. Passwords you enter are checked for weakness. When entering a password that is easy to guess (such as a dictionary word or a name) you will see a warning. It is a good security practice to use strong passwords.

Important: User Name and Password

Remember both your user name and the password because they are needed each time you log in to the system.

If you install openSUSE Leap on a machine with one or more existing Linux installations, YaST allows you to import user data such as user names and passwords. Select *Import User Data from a Previous Installation* and then *Choose Users* for import.

If you do not want to configure any local users (for example when setting up a client on a network with centralized user authentication), skip this step by choosing *Next* and confirming the warning. Network user authentication can be configured at any time later in the installed system; refer to *Chapter 5, Managing Users with YaST* for instructions.

Two additional options are available:

Use this Password for System Administrator

If checked, the same password you have entered for the user will be used for the system administrator `root`. This option is suitable for stand-alone workstations or machines in a home network that are administrated by a single user. When not checked, you are prompted for a system administrator password in the next step of the installation workflow (see *Section 3.10, "Authentication for the System Administrator "root""*).

Automatic Login

This option automatically logs the current user in to the system when it starts. This is mainly useful if the computer is operated by only one user.

Warning: Automatic Login

With the automatic login enabled, the system boots straight into your desktop with no authentication. If you store sensitive data on your system, you should not enable this option if the computer can also be accessed by others.

In an environment where users are centrally managed (for example by NIS or LDAP) you may want to skip the creation of local users. Select *Skip User Creation* in this case.

3.10 Authentication for the System Administrator “root”

If you have not chosen *Use this Password for System Administrator* in the previous step, you will be prompted to enter a password for the System Administrator `root` or provide a public SSH key. Otherwise this configuration step is skipped.

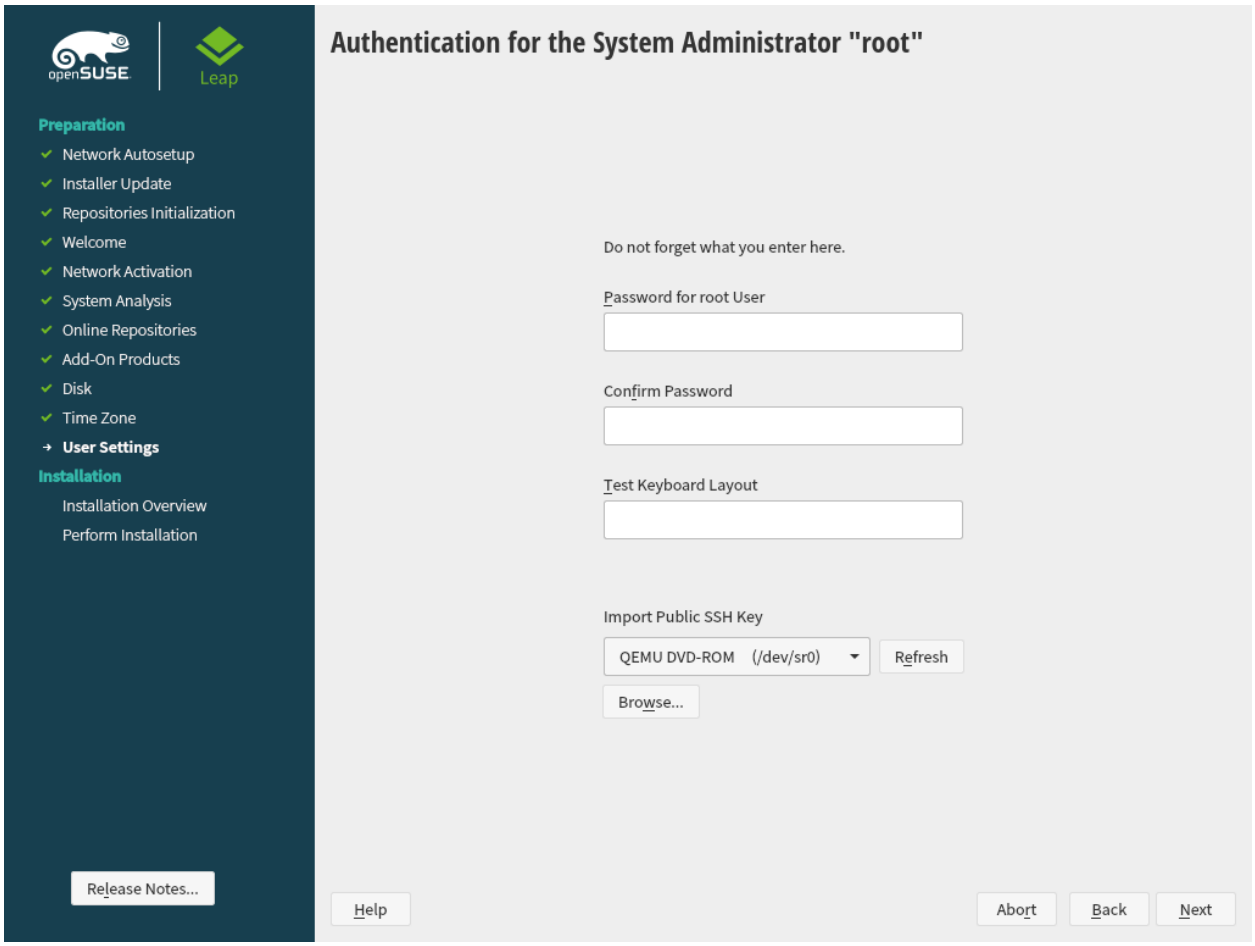


FIGURE 3.7: AUTHENTICATION FOR THE SYSTEM ADMINISTRATOR `root`

`root` is the name of the superuser, or the administrator of the system. Unlike regular users, `root` has unlimited rights to change the system configuration, install programs, and set up new hardware. If users forget their passwords or have other problems with the system, `root` can help. The `root` account should only be used for system administration, maintenance, and repair. Logging in as `root` for daily work is rather risky: a single mistake could lead to irretrievable loss of system files.

For verification purposes, the password for `root` must be entered twice. Do not forget the `root` password. After having been entered, this password cannot be retrieved.

Tip: Passwords and Keyboard Layout

It is recommended to only use characters that are available on an English keyboard. In case of a system error or when you need to start your system in rescue mode a localized keyboard might not be available.

The `root` password can be changed any time later in the installed system. To do so run YaST and start *Security and Users* › *User and Group Management*.

Important: The `root` User

The user `root` has all the permissions needed to make changes to the system. To carry out such tasks, the `root` password is required. You cannot carry out any administrative tasks without this password.

In some situations it is preferred to access the system remotely via SSH using a public key. This screen allows you to select a public key from a medium.

The following procedure describes how to add a public SSH key from a USB stick. It works the same way with CD/DVD-ROM or from an existing partition. Proceed as follows:

PROCEDURE 3.1: ADDING A PUBLIC SSH KEY FOR USER `root`

1. Insert into your computer the USB storage device containing the public SSH key. The public SSH key has the file extension `.pub`.
2. Click *Refresh*. You should see the device in the list selector under *Import Public Key*.
3. Click *Browse* and select the public SSH key.
4. Proceed with *Next*.
5. In the *Installation Settings* summary, make sure to check under *Firewall and SSH* the SSH port. Click *open* so it reads *SSH port will be open*.

After the installation is finished, you can log in through SSH using the provided public SSH key.

3.11 Installation Settings

On the last step before the real installation takes place, you can alter installation settings suggested by the installer. To modify the suggestions, click the respective headline. After having made changes to a particular setting, you are always returned to the Installation Settings window, which is updated accordingly.

If you have added an SSH key for your `root` as mentioned in *Procedure 3.1*, make sure to open the SSH port in the *Firewall and SSH* settings.

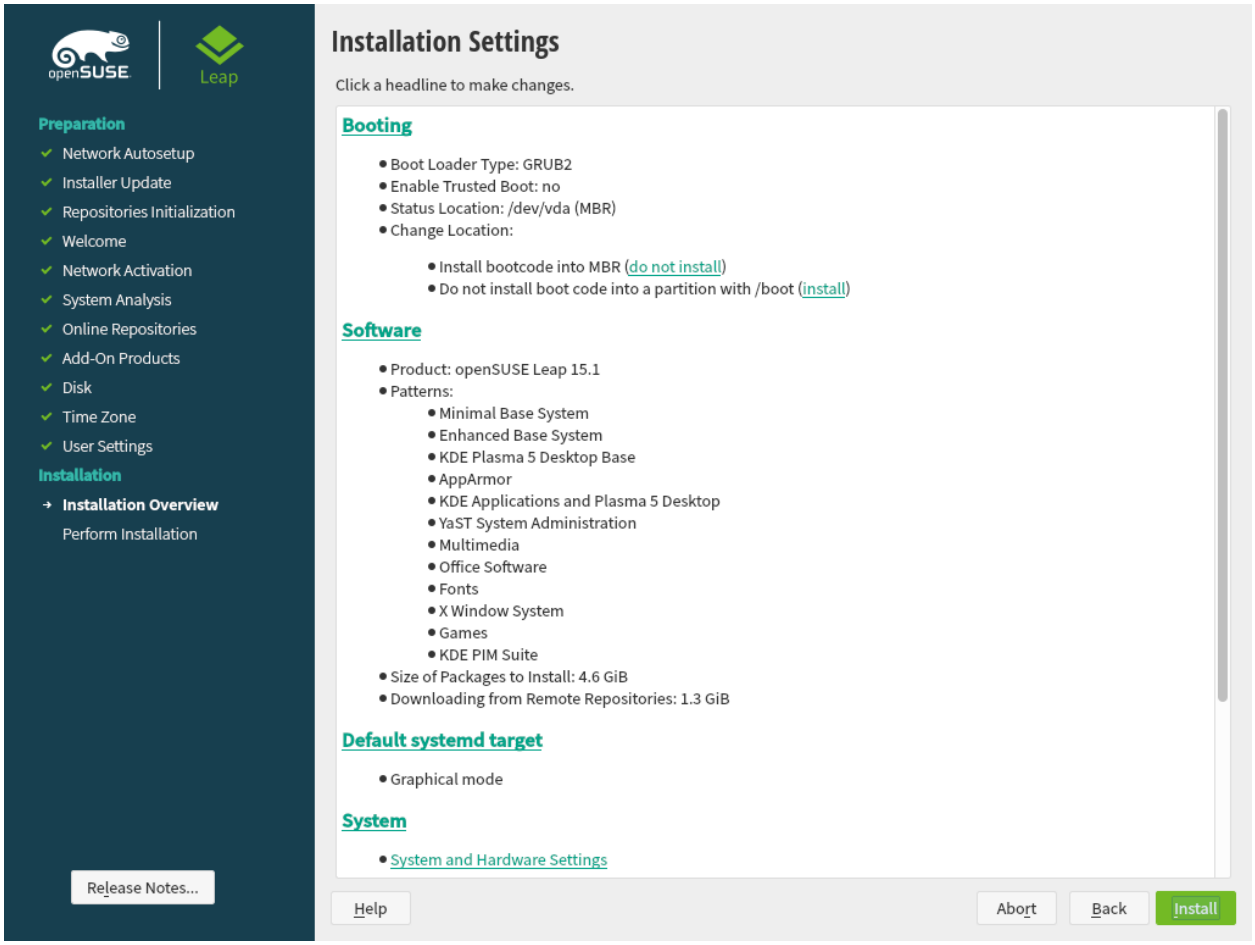


FIGURE 3.8: INSTALLATION SETTINGS

3.11.1 *Software*

openSUSE Leap contains several software patterns for various application purposes. The available choice of patterns and packages depends on your selection of modules and extensions.

Click *Software* to open the *Software Selection and System Tasks* screen where you can modify the pattern selection according to your needs. Select a pattern from the list and see a description in the right-hand part of the window.

Each pattern contains several software packages needed for specific functions (for example Multimedia or Office software). If you chose *Generic Desktop* in the *System Role* dialog choose a desktop environment from the list of available *Graphical Environments*. For a more detailed selection based on software packages to install, select *Details* to switch to the YaST Software Manager.

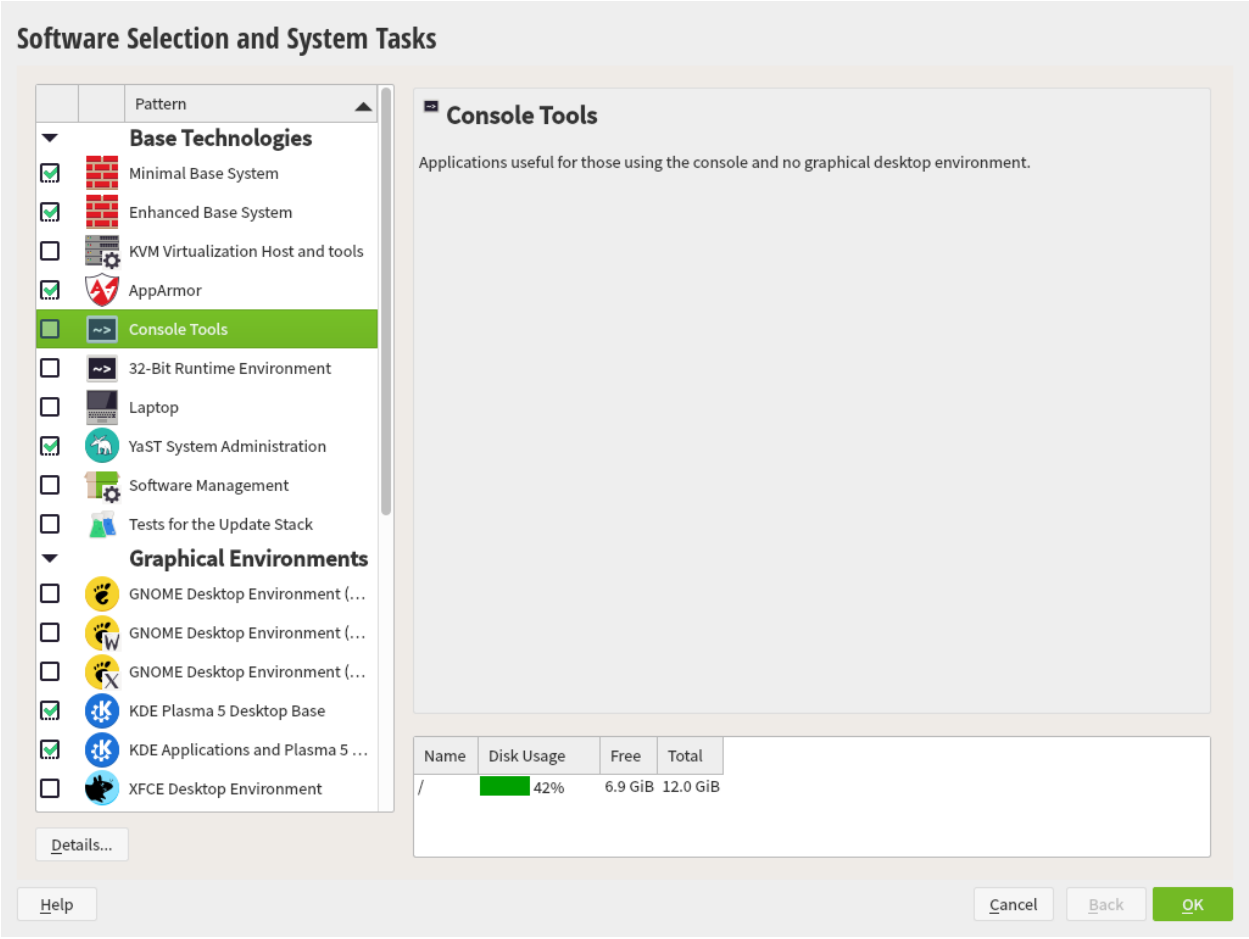


FIGURE 3.9: SOFTWARE SELECTION AND SYSTEM TASKS

You can also install additional software packages or remove software packages from your system at any later time with the YaST Software Manager. For more information, refer to *Chapter 10, Installing or Removing Software*.

By default, openSUSE Leap uses the Wayland display server protocol.

Tip: Adding Secondary Languages

The language you selected with the first step of the installation will be used as the primary (default) language for the system. You can add secondary languages from within the *Software* dialog by choosing *Details* › *View* › *Languages*.

3.11.2 *Booting*

The installer proposes a boot configuration for your system. Other operating systems found on your computer, such as Microsoft Windows or other Linux installations, will automatically be detected and added to the boot loader. However, openSUSE Leap will be booted by default. Normally, you can leave these settings unchanged. If you need a custom setup, modify the proposal according to your needs. For information, see *Book "*Reference*", Chapter 12 "The Boot Loader GRUB 2", Section 12.3 "Configuring the Boot Loader with YaST"*.

Important: Software RAID 1

Booting a configuration where `/boot` resides on a software RAID 1 device is supported, but it requires to install the boot loader into the MBR (*Boot Loader Location* › *Boot from Master Boot Record*). Having `/boot` on software RAID devices with a level other than RAID 1 is not supported.

3.11.3 *Firewall and SSH*

By default `firewalld` is enabled on all configured network interfaces. To globally disable the firewall for this computer, click *Disable* (not recommended).

Note: Firewall Settings

If the firewall is activated, all interfaces are configured to be in the "External Zone", where all ports are closed by default, ensuring maximum security. The only port you can open during the installation is port 22 (SSH), to allow remote access. All other services requiring network access (such as FTP, Samba, Web server, etc.) will only work after having adjusted the firewall settings. Refer to *Book "*Security Guide*", Chapter 16 "Masquerading and Firewalls"* for more information.

To enable remote access via the secure shell (SSH), make sure the `SSH service` is enabled and the `SSH port` is open.

Tip: Existing SSH Host Keys

If you install openSUSE Leap on a machine with existing Linux installations, the installation routine imports an SSH host key. It chooses the host key with the most recent access time by default.

If you are performing a remote administration over VNC, you can also specify whether the machine should be accessible via VNC after the installation. Note that enabling VNC also requires you to set the *Default systemd Target* to *graphical*.

3.11.4 *Default systemd Target*

openSUSE Leap can boot into two different targets (formerly known as “runlevels”). The *graphical* target starts a display manager, whereas the *multi-user* target starts the command line interface.

The default target is *graphical*. In case you have not installed the *X Window System* patterns, you need to change it to *multi-user*. If the system should be accessible via VNC, you need to choose *graphical*.

3.11.5 *Import SSH Host Keys and Configuration*

If an existing Linux installation on your computer was detected, YaST will import the most recent SSH host key found in `/etc/ssh` by default, optionally including other files in the directory as well. This makes it possible to reuse the SSH identity of the existing installation, avoiding the `REMOTE HOST IDENTIFICATION HAS CHANGED` warning on the first connection. Note that this item is not shown in the installation summary if YaST has not discovered any other installations. You have the following choices:

I would like to import SSH keys from a previous install:

Select this option to import the SSH host key and optionally the configuration of an installed system. You can select the installation to import from in the option list below.

Import SSH Configuration

Enable this to copy other files in `/etc/ssh` to the installed system in addition to the host keys.

3.11.6 *System*

This screen lists all the hardware information the installer could obtain about your computer. When opened for the first time, the hardware detection is started. Depending on your system, this may take some time. Select any item in the list and click *Details* to see detailed information about the selected item. Use *Save to File* to save a detailed list to either the local file system or a removable device.

Advanced users can also change the *PCI ID Setup* and kernel settings by choosing *Kernel Settings*. A screen with two tabs opens:

PCI ID Setup

Each kernel driver contains a list of device IDs of all devices it supports. If a new device is not in any driver's database, the device is treated as unsupported, even if it can be used with an existing driver. You can add PCI IDs to a device driver here. Only advanced users should attempt to do so.

To add an ID, click *Add* and select whether to *Manually* enter the data, or whether to choose from a list. Enter the required data. The *SysFS Dir* is the directory name from `/sys/bus/pci/drivers`—if empty, the *driver* name is used as the directory name. Existing entries can be managed with *Edit* and *Delete*.

Kernel Settings

Change the *Global I/O Scheduler* here. If *Not Configured* is chosen, the default setting for the respective architecture will be used. This setting can also be changed at any time later from the installed system. Refer to *Book "*System Analysis and Tuning Guide*", Chapter 12 "Tuning I/O Performance"* for details on I/O tuning.

Also activate the *Enable SysRq Keys* here. These keys will let you issue basic commands (such as rebooting the system or writing kernel dumps) in case the system crashes. Enabling these keys is recommended when doing kernel development. Refer to https://www.kernel.org/doc/html/latest/admin-guide/sysrq.html for details.

3.12 Performing the Installation

After configuring all installation settings, click *Install* in the Installation Settings window to start the installation. Some software may require a license confirmation. If your software selection includes such software, license confirmation dialogs are displayed. Click *Accept* to install the software package. When not agreeing to the license, click *I Disagree* and the software package will not be installed. In the dialog that follows, confirm with *Install* again.

The installation usually takes between 15 and 30 minutes, depending on the system performance and the selected software scope. After having prepared the hard disk and having saved and restored the user settings, the software installation starts. Choose *Details* to switch to the installation log or *Release Notes* to read important up-to-date information that was not available when the manuals were printed.

After the software installation has completed, the system reboots into the new installation where you can log in. To customize the system configuration or to install additional software packages, start YaST.

4 Troubleshooting

This section highlights some typical problems you may run into during installation and offers possible solutions or workarounds.

4.1 Checking Media

If you encounter any problems using the openSUSE Leap installation media, check the integrity of your installation media. Boot from the media and choose*More* › *Check Installation Media* from the boot menu. A minimal system boots and lets you choose which device to check. Select the respective device and confirm with *OK* to perform the check.

In a running system, start YaST and choose *Software* › *Media Check*. Insert the medium click *Start Check*. Checking may take several minutes.

If errors are detected during the check, do not use this medium for installation. Media problems may, for example, occur when having burned the medium on DVD yourself. Burning the media at a low speed (4x) helps to avoid problems.

4.2 No Bootable DVD Drive Available

If your computer does not contain a built-in bootable DVD drive there are several alternatives. This is also an option if your drive is not supported by openSUSE Leap.

Using External DVD Device

Linux supports most existing DVD drives. If the system has no DVD drive, it is still possible that an external DVD drive, connected through USB, FireWire, or SCSI, can be used to boot the system. Sometimes a BIOS update may help if you encounter problems.

Network Boot via PXE

If a machine lacks a DVD drive, but provides a working Ethernet connection, perform a completely network-based installation.

USB Flash Drive

You can use a USB flash drive if your machine lacks a DVD drive and network connection.

4.3 Booting from Installation Media Fails

One reason a machine does not boot the installation media can be an incorrect boot sequence setting in BIOS. The BIOS boot sequence must have DVD drive set as the first entry for booting. Otherwise the machine would try to boot from another medium, typically the hard disk. Guidance for changing the BIOS boot sequence can be found in the documentation provided with your mainboard, or in the following paragraphs.

The BIOS is the software that enables the very basic functions of a computer. Motherboard vendors provide a BIOS specifically made for their hardware. Normally, the BIOS setup can only be accessed at a specific time—when the machine is booting. During this initialization phase, the machine performs several diagnostic hardware tests. One of them is a memory check, indicated by a memory counter. When the counter appears, look for a line, usually below the counter or somewhere at the bottom, mentioning the key to press to access the BIOS setup. Usually the key to press is one of Del, F1, or Esc. Press this key until the BIOS setup screen appears.

PROCEDURE 4.1: CHANGING THE BIOS BOOT SEQUENCE

1. Enter the BIOS using the proper key as announced by the boot routines and wait for the BIOS screen to appear.
2. To change the boot sequence in an AWARD BIOS, look for the *BIOS FEATURES SETUP* entry. Other manufacturers may have a different name for this, such as *ADVANCED CMOS SETUP*. When you have found the entry, select it and confirm with Enter.
3. In the screen that opens, look for a subentry called *BOOT SEQUENCE* or *BOOT ORDER*. Change the settings by pressing Page ↑ or Page ↓ until the DVD drive is listed first.
4. Leave the BIOS setup screen by pressing Esc. To save the changes, select *SAVE & EXIT SETUP*, or press F10. To confirm that your settings should be saved, press Y.

PROCEDURE 4.2: CHANGING THE BOOT SEQUENCE IN AN SCSI BIOS (ADAPTEC HOST ADAPTER)

1. Open the setup by pressing Ctrl–A.
2. Select *Disk Utilities*. The connected hardware components are now displayed.
   Make note of the SCSI ID of your DVD drive.
3. Exit the menu with Esc.
4. Open *Configure Adapter Settings*. Under *Additional Options*, select *Boot Device Options* and press Enter.

5. Enter the ID of the DVD drive and press Enter again.

6. Press Esc twice to return to the start screen of the SCSI BIOS.

7. Exit this screen and confirm with *Yes* to boot the computer.

Regardless of what language and keyboard layout your final installation will be using, most BIOS configurations use the US keyboard layout as shown in the following figure:

FIGURE 4.1: US KEYBOARD LAYOUT

4.4 Boot Failure

Some hardware types, mainly very old or very recent ones, fail to boot. Reasons can be missing support for hardware in the installation kernel or drivers causing problems on some specific hardware.

If your system fails to install using the standard *Installation* mode from the first installation boot screen, try the following:

1. With the DVD still in the drive, reboot the machine with Ctrl–Alt–Del or using the hardware reset button.

2. When the boot screen appears, press F5, use the arrow keys of your keyboard to navigate to *No ACPI* and press Enter to launch the boot and installation process. This option disables the support for ACPI power management techniques.

3. Proceed with the installation as described in *Chapter 3, Installation Steps*.

If this fails, proceed as above, but choose *Safe Settings* instead. This option disables ACPI and DMA support. Most hardware will boot with this option.

If both of these options fail, use the boot parameters prompt to pass any additional parameters needed to support this type of hardware to the installation kernel. For more information about the parameters available as boot parameters, refer to the kernel documentation located in `/usr/src/linux/Documentation/kernel-parameters.txt`.

Tip: Obtaining Kernel Documentation

Install the `kernel-source` package to view the kernel documentation.

There are other ACPI-related kernel parameters that can be entered at the boot prompt prior to booting for installation:

`acpi=off`

This parameter disables the complete ACPI subsystem on your computer. This may be useful if your computer cannot handle ACPI or if you think ACPI in your computer causes trouble.

`acpi=force`

Always enable ACPI even if your computer has an old BIOS dated before the year 2000. This parameter also enables ACPI if it is set in addition to `acpi=off`.

`acpi=noirq`

Do not use ACPI for IRQ routing.

`acpi=ht`

Run only enough ACPI to enable hyper-threading.

`acpi=strict`

Be less tolerant of platforms that are not strictly ACPI specification compliant.

`pci=noacpi`

Disable PCI IRQ routing of the new ACPI system.

`pnpacpi=off`

This option is for serial or parallel problems when your BIOS setup contains wrong interrupts or ports.

`notsc`

Disable the time stamp counter. This option can be used to work around timing problems on your systems. It is a recent feature, so if you see regressions on your machine, especially time related or even total hangs, this option is worth a try.

`nohz=off`

Disable the nohz feature. If your machine hangs, this option may help. Otherwise it is of no use.

When you have determined the right parameter combination, YaST automatically writes them to the boot loader configuration to make sure that the system boots properly next time.

If inexplicable errors occur when the kernel is loaded or during the installation, select *Memory Test* in the boot menu to check the memory. If *Memory Test* returns an error, it is usually a hardware error.

4.5 Fails to Launch Graphical Installer

After you insert the medium into your drive and reboot your machine, the installation screen comes up, but after you select *Installation*, the graphical installer does not start.

There are several ways to deal with this situation:

- Try to select another screen resolution for the installation dialogs.
- Select *Text Mode* for installation.
- Do a remote installation via VNC using the graphical installer.

PROCEDURE 4.3: CHANGE SCREEN RESOLUTION FOR INSTALLATION

1. Boot for installation.
2. Press F3 to open a menu from which to select a lower resolution for installation purposes.
3. Select *Installation* and proceed with the installation as described in *Chapter 3, Installation Steps*.

PROCEDURE 4.4: INSTALLATION IN TEXT MODE

1. Boot for installation.
2. Press F3 and select *Text Mode*.
3. Select *Installation* and proceed with the installation as described in *Chapter 3, Installation Steps*.

PROCEDURE 4.5: VNC INSTALLATION

1. Boot for installation.
2. Enter the following text at the boot parameters prompt:

   ```
   vnc=1 vncpassword=SOME_PASSWORD
   ```

   Replace *SOME_PASSWORD* with the password to use for VNC installation.
3. Select *Installation* then press Enter to start the installation.
   Instead of starting right into the graphical installation routine, the system continues to run in a text mode. The system then halts, displaying a message containing the IP address and port number at which the installer can be reached via a browser interface or a VNC viewer application.
4. If using a browser to access the installer, launch the browser and enter the address information provided by the installation routines on the future openSUSE Leap machine and press Enter:

   ```
   http://IP_ADDRESS_OF_MACHINE:5801
   ```

   A dialog opens in the browser window prompting you for the VNC password. Enter it and proceed with the installation as described in *Chapter 3, Installation Steps*.

   Important: Cross-platform Support

   Installation via VNC works with any browser under any operating system, provided Java support is enabled.

   Provide the IP address and password to your VNC viewer when prompted. A window opens, displaying the installation dialogs. Proceed with the installation as usual.

4.6 Only Minimalist Boot Screen Started

You inserted the medium into the drive, the BIOS routines are finished, but the system does not start with the graphical boot screen. Instead it launches a very minimalist text-based interface. This may happen on any machine not providing sufficient graphics memory for rendering a graphical boot screen.

Although the text boot screen looks minimalist, it provides nearly the same functionality as the graphical one:

Boot Options

Unlike the graphical interface, the different boot parameters cannot be selected using the cursor keys of your keyboard. The boot menu of the text mode boot screen offers some keywords to enter at the boot prompt. These keywords map to the options offered in the graphical version. Enter your choice and press `Enter` to launch the boot process.

Custom Boot Options

After selecting a boot parameter, enter the appropriate keyword at the boot prompt or enter some custom boot parameters as described in *Section 4.4, "Boot Failure"*. To launch the installation process, press `Enter`.

Screen Resolutions

Use the function keys (`F1` ... `F12`) to determine the screen resolution for installation. If you need to boot in text mode, choose `F3`.

II Administration

5 Managing Users with YaST

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

5.1 User and Group Administration Dialog

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

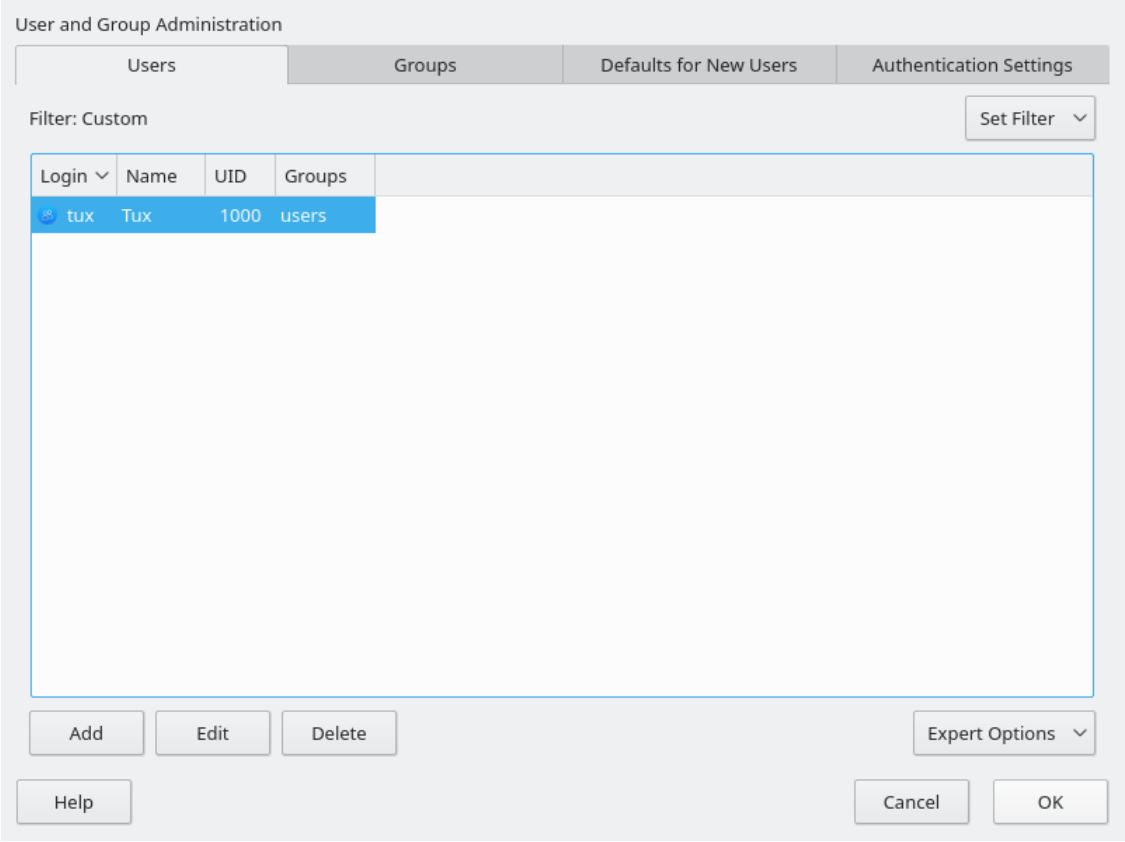


FIGURE 5.1: YAST USER AND GROUP ADMINISTRATION

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

Depending on the set of users you choose to view and modify with, the dialog (local users, network users, system users), the main window shows several tabs. These allow you to execute the following tasks:

Managing User Accounts

From the *Users* tab create, modify, delete or temporarily disable user accounts as described in *Section 5.2, "Managing User Accounts"*. Learn about advanced options like enforcing password policies, using encrypted home directories, or managing disk quotas in *Section 5.3, "Additional Options for User Accounts"*.

Changing Default Settings

Local users accounts are created according to the settings defined on the *Defaults for New Users* tab. Learn how to change the default group assignment, or the default path and access permissions for home directories in *Section 5.4, "Changing Default Settings for Local Users"*.

Assigning Users to Groups

Learn how to change the group assignment for individual users in *Section 5.5, "Assigning Users to Groups"*.

Managing Groups

From the *Groups* tab, you can add, modify or delete existing groups. Refer to *Section 5.6, "Managing Groups"* for information on how to do this.

Changing the User Authentication Method

When your machine is connected to a network that provides user authentication methods like NIS or LDAP, you can choose between several authentication methods on the *Authentication Settings* tab. For more information, refer to *Section 5.7, "Changing the User Authentication Method"*.

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

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

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

5.2 Managing User Accounts

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

Note: Changing User IDs of Existing Users

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

In the following, learn how to set up default user accounts. For further options, refer to *Section 5.3, "Additional Options for User Accounts"*.

PROCEDURE 5.1: ADDING OR MODIFYING USER ACCOUNTS

1. Open the YaST *User and Group Administration* dialog and click the *Users* tab.
2. With *Set Filter* define the set of users you want to manage. The dialog lists users in the system and the groups the users belong to.
3. To modify options for an existing user, select an entry and click *Edit*.
   To create a new user account, click *Add*.
4. Enter the appropriate user data on the first tab, such as *Username* (which is used for login) and *Password*. This data is sufficient to create a new user. If you click *OK* now, the system will automatically assign a user ID and set all other values according to the default.
5. Activate *Receive System Mail* if you want any kind of system notifications to be delivered to this user's mailbox. This creates a mail alias for `root` and the user can read the system mail without having to first log in as `root`.
   The mails sent by system services are stored in the local mailbox `/var/spool/mail/` *`USERNAME`*, where *`USERNAME`* is the login name of the selected user. To read e-mails, you can use the **`mail`** command.
6. To adjust further details such as the user ID or the path to the user's home directory, do so on the *Details* tab.
   If you need to relocate the home directory of an existing user, enter the path to the new home directory there and move the contents of the current home directory with *Move to New Location*. Otherwise, a new home directory is created without any of the existing data.

7. To force users to regularly change their password or set other password options, switch to *Password Settings* and adjust the options. For more details, refer to *Section 5.3.2, "Enforcing Password Policies"*.

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

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

Tip: Matching User IDs

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

PROCEDURE 5.2: DISABLING OR DELETING USER ACCOUNTS

1. Open the YaST *User and Group Administration* dialog and click the *Users* tab.

2. To temporarily disable a user account without deleting it, select the user from the list and click *Edit*. Activate *Disable User Login*. The user cannot log in to your machine until you enable the account again.

3. To delete a user account, select the user from the list and click *Delete*. Choose if you also want to delete the user's home directory or to retain the data.

5.3 Additional Options for User Accounts

In addition to the settings for a default user account, openSUSE® Leap offers further options. For example, options to enforce password policies, use encrypted home directories or define disk quotas for users and groups.

5.3.1 Automatic Login and Passwordless Login

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

Warning: Security Risk

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

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

5.3.2 Enforcing Password Policies

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

PROCEDURE 5.3: CONFIGURING PASSWORD SETTINGS

1. Open the YaST *User and Group Administration* dialog and select the *Users* tab.
2. Select the user for which to change the password options and click *Edit*.
3. Switch to the *Password Settings* tab. The user's last password change is displayed on the tab.
4. To make the user change their password at next login, activate *Force Password Change*.
5. To enforce password rotation, set a *Maximum Number of Days for the Same Password* and a *Minimum Number of Days for the Same Password*.
6. To remind the user to change their password before it expires, set the number of *Days before Password Expiration to Issue Warning*.

7. To restrict the period of time the user can log in after their password has expired, change the value in *Days after Password Expires with Usable Login*.
8. You can also specify a certain expiration date for the complete account. Enter the *Expiration Date* in *YYYY-MM-DD* format. Note that this setting is not password-related but rather applies to the account itself.
9. For more information about the options and about the default values, click *Help*.
10. Apply your changes with *OK*.

5.3.3 Managing Quotas

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

openSUSE Leap allows usage of `soft` and `hard` quotas. Additionally, grace intervals can be defined that allow users or groups to temporarily violate their quotas by certain amounts.

Soft Quota

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

Hard Quota

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

Grace Period

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

PROCEDURE 5.4: ENABLING QUOTA SUPPORT FOR A PARTITION

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

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

   ```
   tux > sudo systemctl status quotaon
   ```

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

PROCEDURE 5.5: SETTING UP QUOTAS FOR USERS OR GROUPS

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

1. In the YaST *User and Group Administration*, select the user or the group you want to set the quotas for and click *Edit*.
2. On the *Plug-Ins* tab, select the *Manage User Quota* entry and click *Launch* to open the *Quota Configuration* dialog.
3. From *File System*, select the partition to which the quota should apply.
4. Below *Size Limits*, restrict the amount of disk space. Enter the number of 1 KB blocks the user or group may have on this partition. Specify a *Soft Limit* and a *Hard Limit* value.
5. Additionally, you can restrict the number of inodes the user or group may have on the partition. Below *Inodes Limits*, enter a *Soft Limit* and *Hard Limit*.
6. You can only define grace intervals if the user or group has already exceeded the soft limit specified for size or inodes. Otherwise, the time-related text boxes are not activated. Specify the time period for which the user or group is allowed to exceed the limits set above.
7. Confirm your settings with *OK*.
8. Click *OK* to close the administration dialog and save the changes.
   Alternatively, to save all changes without exiting the *User and Group Administration* dialog, click *Expert Options* › *Write Changes Now*.

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

5.4 Changing Default Settings for Local Users

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

1. Open the YaST *User and Group Administration* dialog and select the *Defaults for New Users* tab.
2. To change the primary group the new users should automatically belong to, select another group from *Default Group*.
3. To modify the secondary groups for new users, add or change groups in *Secondary Groups*. The group names must be separated by commas.
4. If you do not want to use `/home/USERNAME` as default path for new users' home directories, modify the *Path Prefix for Home Directory*.
5. To change the default permission modes for newly created home directories, adjust the umask value in *Umask for Home Directory*. For more information about umask, refer to *Book "*Security Guide*", Chapter 10 "Access Control Lists in Linux"* and to the **`umask`** man page.
6. For information about the individual options, click *Help*.
7. Apply your changes with *OK*.

5.5 Assigning Users to Groups

Local users are assigned to several groups according to the default settings which you can access from the *User and Group Administration* dialog on the *Defaults for New Users* tab. In the following, learn how to modify an individual user's group assignment. If you need to change the default group assignments for new users, refer to *Section 5.4, "Changing Default Settings for Local Users"*.

PROCEDURE 5.6: CHANGING A USER'S GROUP ASSIGNMENT

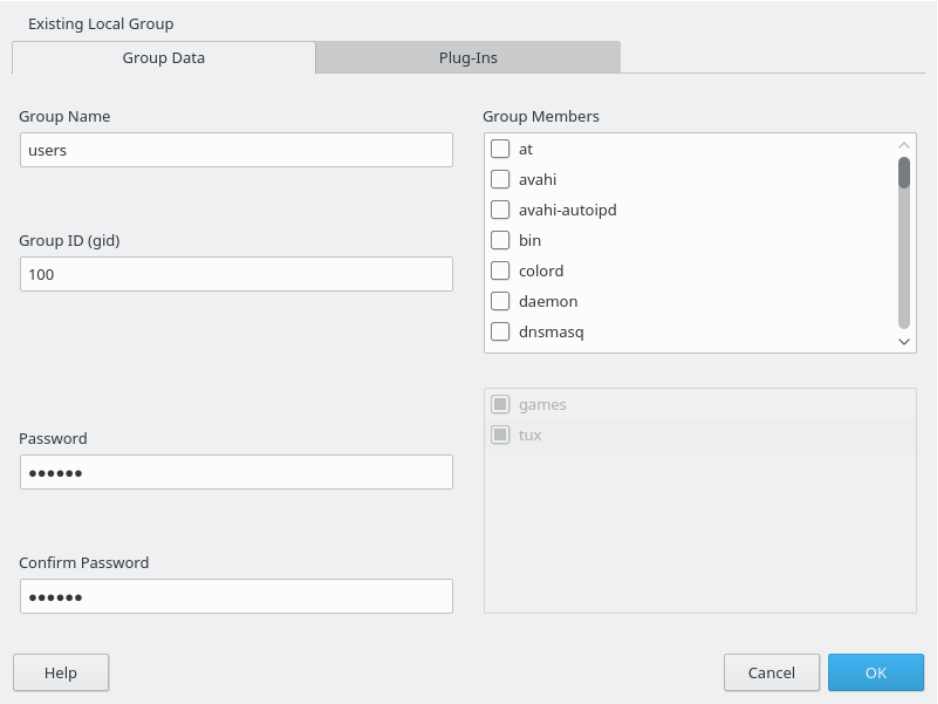
1. Open the YaST *User and Group Administration* dialog and click the *Users* tab. It lists users and the groups the users belong to.
2. Click *Edit* and switch to the *Details* tab.
3. To change the primary group the user belongs to, click *Default Group* and select the group from the list.
4. To assign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

5.6 Managing Groups

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

PROCEDURE 5.7: CREATING AND MODIFYING GROUPS

1. Open the YaST *User and Group Management* dialog and click the *Groups* tab.
2. With *Set Filter* define the set of groups you want to manage. The dialog lists groups in the system.
3. To create a new group, click *Add*.
4. To modify an existing group, select the group and click *Edit*.
5. In the following dialog, enter or change the data. The list on the right shows an overview of all available users and system users which can be members of the group.



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

7. Click *OK* to apply your changes.

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

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

5.7 Changing the User Authentication Method

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

NIS

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

SSSD

The *System Security Services Daemon* (SSSD) can locally cache user data and then allow users to use the data, even if the real directory service is (temporarily) unreachable. For details, see *Book "*Security Guide*", Chapter 4 "Setting Up Authentication Servers and Clients Using YaST", Section 4.3 "SSSD"*.

Samba

SMB authentication is often used in mixed Linux and Windows networks. For details, see *Book "*Reference*", Chapter 21 "Samba"* and *Book "*Security Guide*", Chapter 7 "Active Directory Support"*.

To change the authentication method, proceed as follows:

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

   NIS: *Book "*Security Guide*", Chapter 3 "Using NIS", Section 3.2 "Configuring NIS Clients"*

   LDAP: *Book "*Security Guide*", Chapter 4 "Setting Up Authentication Servers and Clients Using YaST", Section 4.2 "Configuring an Authentication Client with YaST"*

   Samba: *Book "*Reference*", Chapter 21 "Samba", Section 21.5.1 "Configuring a Samba Client with YaST"*

   SSSD: *Book "*Security Guide*", Chapter 4 "Setting Up Authentication Servers and Clients Using YaST", Section 4.3 "SSSD"*
4. After accepting the configuration, return to the *User and Group Administration* overview.
5. Click *OK* to close the administration dialog.

5.8 Default System Users

By default, openSUSE Leap creates user names which cannot be deleted. These users are typically defined in the Linux Standard Base. The following list provides the common user names and their purpose:

COMMON USER NAMES INSTALLED BY DEFAULT

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

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

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

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

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

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

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

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

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

`postfix`
: Used by the Postfix mailer.

`pulse`
: Used by the Pulseaudio sound server.

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

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

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

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

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

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

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

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

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

`systemd-network`
: Used by the **`/usr/lib/systemd/systemd-networkd`** command to manage networks.

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

6 Changing Language and Country Settings with YaST

Working in different countries or having to work in a multilingual environment requires your computer to be set up to support this. openSUSE® Leap can handle different `locales` in parallel. A locale is a set of parameters that defines the language and country settings reflected in the user interface.

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

For those tasks, use the YaST language module as described in *Section 6.1, "Changing the System Language"*. Install secondary languages to get optional localization if you need to start applications or desktops in languages other than the primary one.

Apart from that, the YaST timezone module allows you to adjust your country and timezone settings accordingly. It also lets you synchronize your system clock against a time server. For details, refer to *Section 6.2, "Changing the Country and Time Settings"*.

6.1 Changing the System Language

Depending on how you use your desktop and whether you want to switch the entire system to another language or only the desktop environment itself, there are several ways to do this:

Changing the System Language Globally

Proceed as described in *Section 6.1.1, "Modifying System Languages with YaST"* and *Section 6.1.2, "Switching the Default System Language"* to install additional localized packages with YaST and to set the default language. Changes are effective after the next login. To ensure that the entire system reflects the change, reboot the system or close and restart all running services, applications, and programs.

Changing the Language for the Desktop Only

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

Temporarily Switching Languages for One Application Only

You can also run a single application in another language (that has already been installed with YaST). To do so, start it from the command line by specifying the language code as described in *Section 6.1.3, "Switching Languages for Standard X and GNOME Applications"*.

6.1.1 Modifying System Languages with YaST

YaST knows two different language categories:

Primary Language

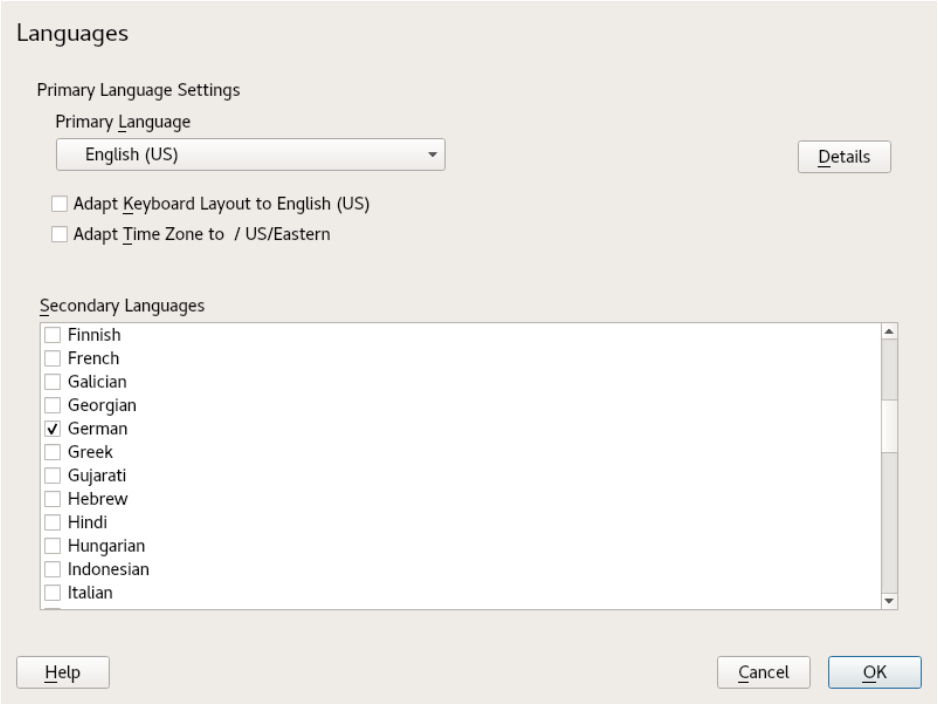
The primary language set in YaST applies to the entire system, including YaST and the desktop environment. This language is used whenever available unless you manually specify another language.

Secondary Languages

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

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

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



PROCEDURE 6.1: INSTALLING ADDITIONAL LANGUAGES

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

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

Tip: Advanced Settings

For advanced keyboard or time zone settings, select *Hardware* › *System Keyboard Layout* or *System* › *Date and Time* in YaST to start the respective dialogs. For more information, refer to *Section 7.1, "Setting Up Your System Keyboard Layout"* and *Section 6.2, "Changing the Country and Time Settings"*.

4. To change language settings specific to the user `root`, click *Details*.

   a. Set *Locale Settings for User root* to the desired value. For more information, click *Help*.

   b. Decide if you want to *Use UTF-8 Encoding* for `root` or not.

5. If your locale was not included in the list of primary languages available, try specifying it with *Detailed Locale Setting*. However, some localization may be incomplete.

6. Confirm your changes in the dialogs with *OK*. If you have selected secondary languages, YaST installs the localized software packages for the additional languages.

The system is now multilingual. However, to start an application in a language other than the primary one, you need to set the desired language explicitly as explained in *Section 6.1.3, "Switching Languages for Standard X and GNOME Applications"*.

6.1.2 Switching the Default System Language

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

1. Start the YaST language module.

2. Select the desired new system language as *Primary Language*.

Important: Deleting Former System Languages

If you switch to a different primary language, the localized software packages for the former primary language will be removed from the system. To switch the default system language but keep the former primary language as additional language, add it as *Secondary Language* by enabling the respective check box.

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

4. Confirm your changes with *OK*.

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

6.1.3 Switching Languages for Standard X and GNOME Applications

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

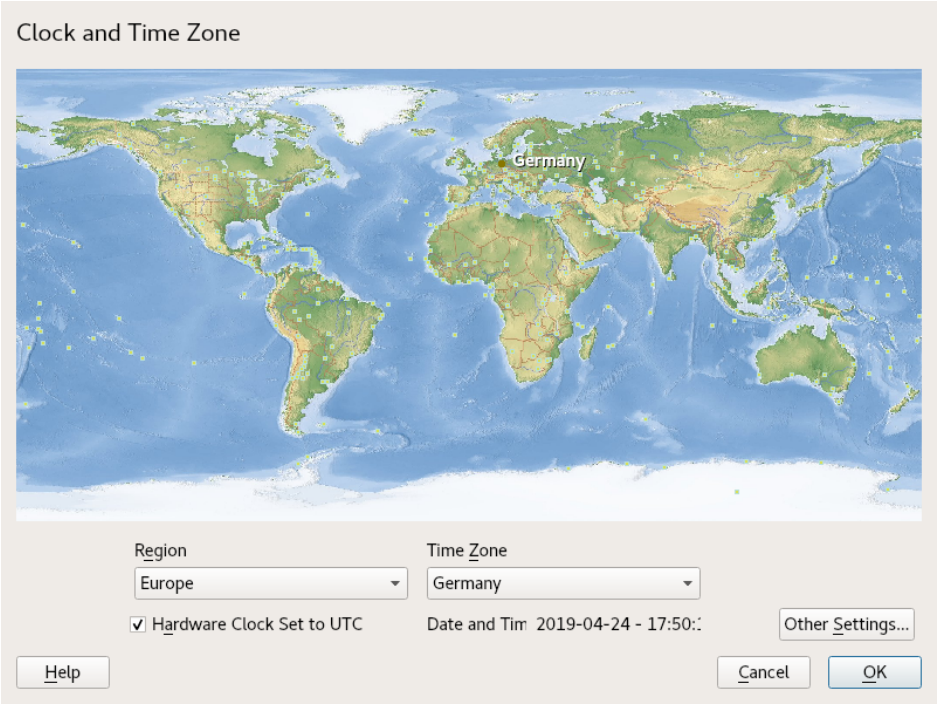
Start the application from the command line by using the following command:

```
LANG=LANGUAGE application
```

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

6.2 Changing the Country and Time Settings

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



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

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

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

Important: Set the Hardware Clock to UTC

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

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

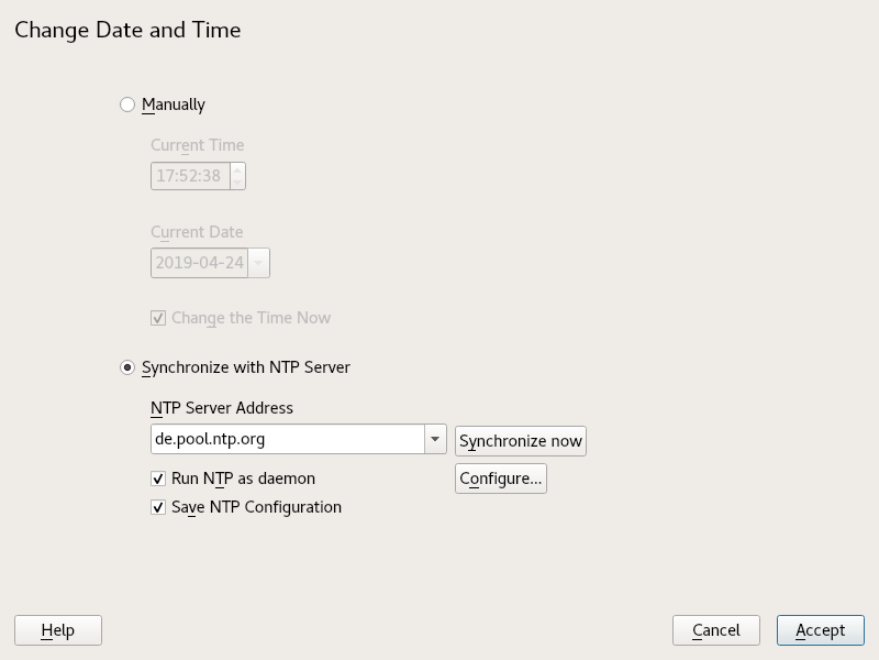
You can change the date and time manually or opt for synchronizing your machine against an NTP server, either permanently or only for adjusting your hardware clock.

PROCEDURE 6.2: MANUALLY ADJUSTING TIME AND DATE

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

PROCEDURE 6.3: SETTING DATE AND TIME WITH NTP SERVER

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



4. Click *Synchronize Now* to get your system time set correctly.
5. To use NTP permanently, enable *Save NTP Configuration*.

6. With the *Configure* button, you can open the advanced NTP configuration. For details, see *Book "*Reference*", Chapter 18 "Time Synchronization with NTP", Section 18.1 "Configuring an NTP Client with YaST"*.

7. Confirm your changes.

7 Setting Up Hardware Components with YaST

YaST allows you to configure hardware items such as audio hardware, your system keyboard layout or printers.

Note: Graphics Card, Monitor, Mouse and Keyboard Settings

Graphics card, monitor, mouse and keyboard can be configured with GNOME tools. See *Book "*GNOME User Guide*", Chapter 3 "Customizing Your Settings"* for details.

7.1 Setting Up Your System Keyboard Layout

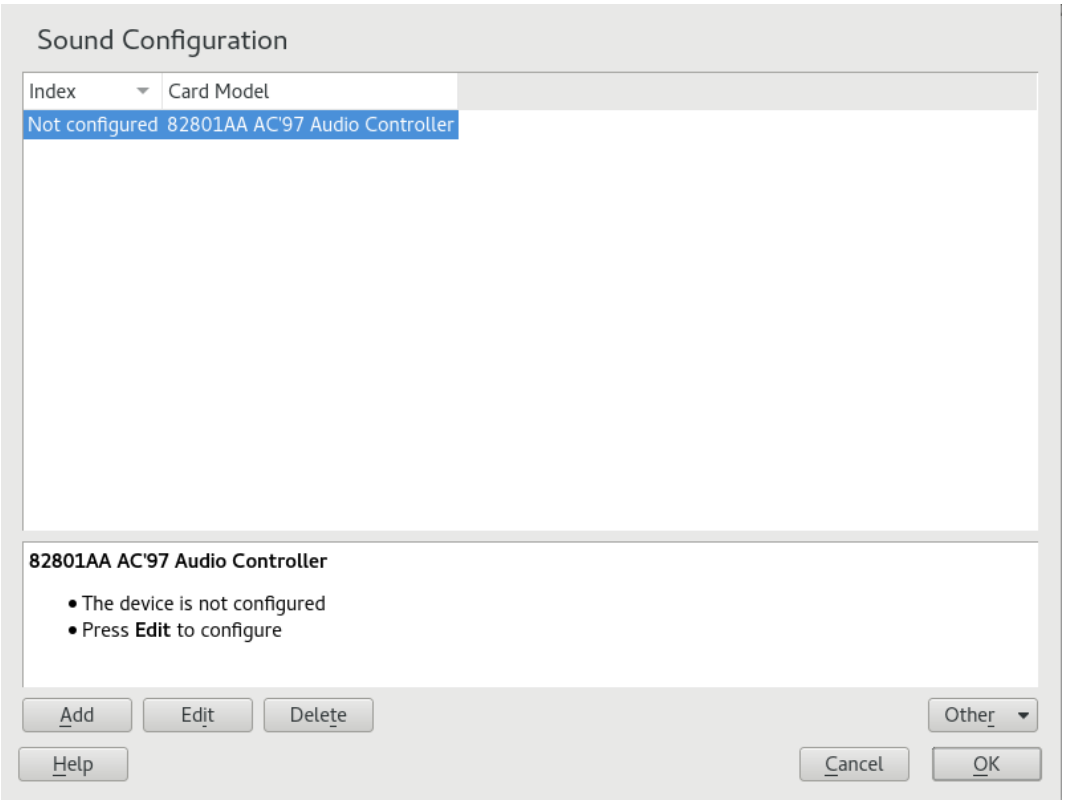
The YaST *System Keyboard Layout* module lets you define the default keyboard layout for the system (also used for the console). Users can modify the keyboard layout in their individual X sessions, using the desktop's tools.

1. Start the YaST *System Keyboard Configuration* dialog by clicking *Hardware* › *System Keyboard Layout* in YaST. Alternatively, start the module from the command line with `sudo yast2 keyboard`.
2. Select the desired *Keyboard Layout* from the list.
3. Optionally, you can also define the keyboard repeat rate or keyboard delay rate in the *Expert Settings*.
4. Try the selected settings in the *Test* text box.
5. If the result is as expected, confirm your changes and close the dialog. The settings are written to `/etc/sysconfig/keyboard`.

7.2 Setting Up Sound Cards

YaST detects most sound cards automatically and configures them with the appropriate values. To change the default settings, or to set up a sound card that could not be configured automatically, use the YaST sound module. There, you can also set up additional sound cards or switch their order.

To start the sound module, start YaST and click *Hardware* › *Sound*. Alternatively, start the *Sound Configuration* dialog directly by running `yast2 sound &` as user `root` from a command line.



The dialog shows all sound cards that were detected.

PROCEDURE 7.1: CONFIGURING SOUND CARDS

If you have added a new sound card or YaST could not automatically configure an existing sound card, follow the steps below. For configuring a new sound card, you need to know your sound card vendor and model. If in doubt, refer to your sound card documentation for the required information. For a reference list of sound cards supported by ALSA with their corresponding sound modules, see http://www.alsa-project.org/main/index.php/Matrix:Main.

During configuration, you can choose between the following setup options:

Quick Automatic Setup

You are not required to go through any of the further configuration steps—the sound card is configured automatically. You can set the volume or any options you want to change later.

Normal Setup

Allows you to adjust the output volume and play a test sound during the configuration.

Advanced setup with possibility to change options

For experts only. Allows you to customize all parameters of the sound card.

Important: Advanced Configuration

Only use this option if you know exactly what you are doing. Otherwise leave the parameters untouched and use the normal or the automatic setup options.

1. Start the YaST sound module.
2. To configure a detected, but *Not Configured* sound card, select the respective entry from the list and click *Edit*.
   To configure a new sound card, click *Add*. Select your sound card vendor and model and click *Next*.
3. Choose one of the setup options and click *Next*.
4. If you have chosen *Normal Setup*, you can now *Test* your sound configuration and make adjustments to the volume. You should start at about ten percent volume to avoid damage to your hearing or the speakers.
5. If all options are set according to your wishes, click *Next*.
   The *Sound Configuration* dialog shows the newly configured or modified sound card.
6. To remove a sound card configuration that you no longer need, select the respective entry and click *Delete*.
7. Click *OK* to save the changes and leave the YaST sound module.

PROCEDURE 7.2: MODIFYING SOUND CARD CONFIGURATIONS

1. To change the configuration of an individual sound card (for experts only!), select the sound card entry in the *Sound Configuration* dialog and click *Edit*.
   This takes you to the *Sound Card Advanced Options* where you can fine-tune several parameters. For more information, click *Help*.
2. To adjust the volume of an already configured sound card or to test the sound card, select the sound card entry in the *Sound Configuration* dialog and click *Other*. Select the respective menu item.

Note: YaST Mixer

The YaST mixer settings provide only basic options. They are intended for troubleshooting (for example, if the test sound is not audible). Access the YaST mixer settings from *Other* › *Volume*. For everyday use and fine-tuning of sound options, use the mixer applet provided by your desktop or the **`alsasound`** command line tool.

3. For playback of MIDI files, select *Other* › *Start Sequencer*.
4. When a supported sound card is detected, you can install SoundFonts for playback of MIDI files:
    a. Insert the original driver CD-ROM into your CD or DVD drive.
    b. Select *Other* › *Install SoundFonts* to copy SF2 SoundFonts™ to your hard disk. The SoundFonts are saved in the directory `/usr/share/sfbank/creative/`.
5. If you have configured more than one sound card in your system you can adjust the order of your sound cards. To set a sound card as primary device, select the sound card in the *Sound Configuration* and click *Other* › *Set as the Primary Card*. The sound device with index `0` is the default device and thus used by the system and the applications.
6. By default, openSUSE Leap uses the PulseAudio sound system. This is an abstraction layer that helps to mix multiple audio streams, bypassing any restrictions the hardware may have. To enable or disable the PulseAudio sound system, click *Other* › *PulseAudio Configuration*. If enabled, PulseAudio daemon is used to play sounds. Disable *PulseAudio Support* to use something else system-wide.

The volume and configuration of all sound cards are saved when you click *OK* and leave the YaST sound module. The mixer settings are saved to the file `/etc/asound.state`. The ALSA configuration data is appended to the end of the file `/etc/modprobe.d/sound` and written to `/etc/sysconfig/sound`.

7.3 Setting Up a Printer

YaST can be used to configure a local printer connected to your machine via USB and to set up printing with network printers. It is also possible to share printers over the network. Further information about printing (general information, technical details, and troubleshooting) is available in *Chapter 8, Printer Operation*.

In YaST, click *Hardware* › *Printer* to start the printer module. By default it opens in the *Printer Configurations* view, displaying a list of all printers that are available and configured. This is especially useful when having access to a lot of printers via the network. From here you can also *Print a Test Page* and configure printers.

Note: Starting CUPS

To print from your system, CUPS must be running. In case it is not running, you are asked to start it. Answer with *Yes*, or you cannot configure printing. In case CUPS is not started at boot time, you will also be asked to enable this feature. It is recommended to say *Yes*, otherwise CUPS would need to be started manually after each reboot.

7.3.1 Configuring Printers

Usually a USB printer is automatically detected. There are two possible reasons it is not automatically detected:

- The USB printer is switched off.
- Communication between printer and computer is not possible. Check the cable and the plugs to make sure that the printer is properly connected. If this is the case, the problem may not be printer-related, but rather a USB-related problem.

Configuring a printer is a three-step process: specify the connection type, choose a driver, and name the print queue for this setup.

For many printer models, several drivers are available. When configuring the printer, YaST defaults to those marked `recommended` as a general rule. Normally it is not necessary to change the driver. However, if you want a color printer to print only in black and white, you can use a driver that does not support color printing. If you experience performance problems with a PostScript printer when printing graphics, try to switch from a PostScript driver to a PCL driver (provided your printer understands PCL).

If no driver for your printer is listed, try to select a generic driver with an appropriate standard language from the list. Refer to your printer's documentation to find out which language (the set of commands controlling the printer) your printer understands. If this does not work, refer to *Section 7.3.1.1, "Adding Drivers with YaST"* for another possible solution.

A printer is never used directly, but always through a print queue. This ensures that simultaneous jobs can be queued and processed one after the other. Each print queue is assigned to a specific driver, and a printer can have multiple queues. This makes it possible to set up a second queue on a color printer that prints black and white only, for example. Refer to *Section 8.1, "The CUPS Workflow"* for more information about print queues.

PROCEDURE 7.3: ADDING A NEW PRINTER

1. Start the YaST printer module with *Hardware* › *Printer*.
2. In the *Printer Configurations* screen click *Add*.
3. If your printer is already listed under `Specify the Connection`, proceed with the next step. Otherwise, try to *Detect More* or start the *Connection Wizard*.
4. In the text box under `Find and Assign a Driver` enter the vendor name and the model name and click *Search for*.
5. Choose a driver that matches your printer. It is recommended to choose the driver listed first. If no suitable driver is displayed:
    a. Check your search term.
    b. Broaden your search by clicking *Find More*.
    c. Add a driver as described in *Section 7.3.1.1, "Adding Drivers with YaST"*.
6. Specify the `Default paper size`.
7. In the *Set Arbitrary Name* field, enter a unique name for the print queue.
8. The printer is now configured with the default settings and ready to use. Click *OK* to return to the *Printer Configurations* view. The newly configured printer is now visible in the list of printers.

7.3.1.1 Adding Drivers with YaST

Not all printer drivers available for openSUSE Leap are installed by default. If no suitable driver is available in the *Find and Assign a Driver* dialog when adding a new printer install a driver package containing drivers for your printers:

PROCEDURE 7.4: INSTALLING ADDITIONAL DRIVER PACKAGES

1. Start the YaST printer module with *Hardware* › *Printer*.
2. In the *Printer Configurations* screen, click *Add*.
3. In the `Find and Assign a Driver` section, click *Driver Packages*.
4. Choose one or more suitable driver packages from the list. Do *not* specify the path to a printer description file.
5. Choose *OK* and confirm the package installation.
6. To directly use these drivers, proceed as described in *Procedure 7.3, "Adding a New Printer"*.

PostScript printers do not need printer driver software. PostScript printers need only a PostScript Printer Description (PPD) file which matches the particular model. PPD files are provided by the printer manufacturer.

If no suitable PPD file is available in the *Find and Assign a Driver* dialog when adding a PostScript printer, install a PPD file for your printer:

Several sources for PPD files are available. It is recommended to first try additional driver packages that are shipped with openSUSE Leap but not installed by default (see below for installation instructions). If these packages do not contain suitable drivers for your printer, get PPD files directly from your printer vendor or from the driver CD of a PostScript printer. For details, see *Section 8.8.2, "No Suitable PPD File Available for a PostScript Printer"*. Alternatively, find PPD files at http://www.linuxfoundation.org/collaborate/workgroups/openprinting/database/databaseintro, the "OpenPrinting.org printer database". When downloading PPD files from OpenPrinting, keep in mind that it always shows the latest Linux support status, which is not necessarily met by openSUSE Leap.

PROCEDURE 7.5: ADDING A PPD FILE FOR POSTSCRIPT PRINTERS

1. Start the YaST printer module with *Hardware* › *Printer*.
2. In the *Printer Configurations* screen, click *Add*.

3. In the `Find and Assign a Driver` section, click *Driver Packages*.
4. Enter the full path to the PPD file into the text box under `Make a Printer Description File Available`.
5. Click *OK* to return to the `Add New Printer Configuration` screen.
6. To directly use this PPD file, proceed as described in *Procedure 7.3, "Adding a New Printer"*.

7.3.1.2 Editing a Local Printer Configuration

By editing an existing configuration for a printer you can change basic settings such as connection type and driver. It is also possible to adjust the default settings for paper size, resolution, media source, etc. You can change identifiers of the printer by altering the printer description or location.

1. Start the YaST printer module with *Hardware* › *Printer*.
2. In the *Printer Configurations* screen, choose a local printer configuration from the list and click *Edit*.
3. Change the connection type or the driver as described in *Procedure 7.3, "Adding a New Printer"*. This should only be necessary in case you have problems with the current configuration.
4. Optionally, make this printer the default by checking *Default Printer*.
5. Adjust the default settings by clicking *All Options for the Current Driver*. To change a setting, expand the list of options by clicking the relative `+` sign. Change the default by clicking an option. Apply your changes with *OK*.

7.3.2 Configuring Printing via the Network with YaST

Network printers are not detected automatically. They must be configured manually using the YaST printer module. Depending on your network setup, you can print to a print server (CUPS, LPD, SMB, or IPX) or directly to a network printer (preferably via TCP). Access the configuration view for network printing by choosing *Printing via Network* from the left pane in the YaST printer module.

7.3.2.1 Using CUPS

In a Linux environment CUPS is usually used to print via the network. The simplest setup is to only print via a single CUPS server which can directly be accessed by all clients. Printing via more than one CUPS server requires a running local CUPS daemon that communicates with the remote CUPS servers.

Important: Browsing Network Print Queues

CUPS servers announce their print queues over the network either via the traditional CUPS browsing protocol or via Bonjour/DNS-SD. Clients need to browse these lists, so users can select specific printers to send their print jobs to. To browse network print queues, the service `cups-browsed` provided by the package `cups-filters-cups-browsed` must run on all clients that print via CUPS servers. `cups-browsed` is started automatically when configuring network printing with YaST.

In case browsing does not work after having started `cups-browsed`, the CUPS server(s) probably announce the network print queues via Bonjour/DNS-SD. In this case you need to additionally install the package `avahi` and start the associated service with **`sudo systemctl start avahi-daemon`** on all clients.

PROCEDURE 7.6: PRINTING VIA A SINGLE CUPS SERVER

1. Start the YaST printer module with *Hardware* › *Printer*.
2. From the left pane, launch the *Print via Network* screen.
3. Check *Do All Your Printing Directly via One Single CUPS Server* and specify the name or IP address of the server.
4. Click *Test Server* to make sure you have chosen the correct name or IP address.
5. Click *OK* to return to the *Printer Configurations* screen. All printers available via the CUPS server are now listed.

PROCEDURE 7.7: PRINTING VIA MULTIPLE CUPS SERVERS

1. Start the YaST printer module with *Hardware* › *Printer*.
2. From the left pane, launch the *Print via Network* screen.
3. Check *Accept Printer Announcements from CUPS Servers*.

4. Under `General Settings` specify which servers to use. You may accept connections from all networks available or from specific hosts. If you choose the latter option, you need to specify the host names or IP addresses.

5. Confirm by clicking *OK* and then *Yes* when asked to start a local CUPS server. After the server has started YaST will return to the *Printer Configurations* screen. Click *Refresh list* to see the printers detected so far. Click this button again, in case more printers are available.

7.3.2.2 Using Print Servers other than CUPS

If your network offers print services via print servers other than CUPS, start the YaST printer module with *Hardware* › *Printer* and launch the *Print via Network* screen from the left pane. Start the *Connection Wizard* and choose the appropriate *Connection Type*. Ask your network administrator for details on configuring a network printer in your environment.

7.3.3 Sharing Printers over the Network

Printers managed by a local CUPS daemon can be shared over the network and so turn your machine into a CUPS server. Usually you share a printer by enabling so-called “browsing mode” in CUPS. If browsing is enabled, the local print queues are made available on the network for listening to remote CUPS daemons. It is also possible to set up a dedicated CUPS server that manages all print queues and can directly be accessed by remote clients. In this case it is not necessary to enable browsing.

PROCEDURE 7.8: SHARING PRINTERS

1. Start the YaST printer module with *Hardware* › *Printer*.

2. Launch the *Share Printers* screen from the left pane.

3. Select *Allow Remote Access*. Also check *For computers within the local network* and enable browsing mode by also checking *Publish printers by default within the local network*.

4. Click *OK* to restart the CUPS server and to return to the *Printer Configurations* screen.

5. Regarding CUPS and firewall settings, see http://en.opensuse.org/SDB:CUPS_and_SANE_Firewall_settings.

7.4 Setting Up a Scanner

You can configure a USB or SCSI scanner with YaST. The `sane-backends` package contains hardware drivers and other essentials needed to use a scanner. If you own an HP All-In-One device, see *Section 7.4.1, "Configuring an HP All-In-One Device"*, instructions on how to configure a network scanner are available at *Section 7.4.3, "Scanning over the Network"*.

PROCEDURE 7.9: CONFIGURING A USB OR SCSI SCANNER

1. Connect your USB or SCSI scanner to your computer and turn it on.
2. Start YaST and select *Hardware* › *Scanner*. YaST builds the scanner database and tries to detect your scanner model automatically.
   If a USB or SCSI scanner is not properly detected, try *Other* › *Restart Detection*.
3. To activate the scanner select it from the list of detected scanners and click *Edit*.
4. Choose your model form the list and click *Next* and *Finish*.
5. Use *Other* › *Test* to make sure you have chosen the correct driver.
6. Leave the configuration screen with *OK*.

7.4.1 Configuring an HP All-In-One Device

An HP All-In-One device can be configured with YaST even if it is made available via the network. If you own a USB HP All-In-One device, start configuring as described in *Procedure 7.9, "Configuring a USB or SCSI Scanner"*. If it is detected properly and the *Test* succeeds, it is ready to use.

If your USB device is not properly detected, or your HP All-In-One device is connected to the network, run the HP Device Manager:

1. Start YaST and select *Hardware* › *Scanner*. YaST loads the scanner database.
2. Start the HP Device Manager with *Other* › *Run hp-setup* and follow the on-screen instructions. After having finished the HP Device Manager, the YaST scanner module automatically restarts the auto detection.
3. Test it by choosing *Other* › *Test*.
4. Leave the configuration screen with *OK*.

7.4.2 Sharing a Scanner over the Network

openSUSE Leap allows the sharing of a scanner over the network. To do so, configure your scanner as follows:

1. Configure the scanner as described in *Section 7.4, "Setting Up a Scanner"*.
2. Choose *Other* › *Scanning via Network*.
3. Enter the host names of the clients (separated by a comma) that should be allowed to use the scanner under *Server Settings* › *Permitted Clients for saned* and leave the configuration dialog with *OK*.

7.4.3 Scanning over the Network

To use a scanner that is shared over the network, proceed as follows:

1. Start YaST and select *Hardware* › *Scanner*.
2. Open the network scanner configuration menu by *Other* › *Scanning via Network*.
3. Enter the host name of the machine the scanner is connected to under *Client Settings* › *Servers Used for the net Metadriver*
4. Leave with *OK*. The network scanner is now listed in the Scanner Configuration window and is ready to use.

8 Printer Operation

openSUSE® Leap supports printing with many types of printers, including remote network printers. Printers can be configured manually or with YaST. For configuration instructions, refer to *Section 7.3, "Setting Up a Printer"*. Both graphical and command line utilities are available for starting and managing print jobs. If your printer does not work as expected, refer to *Section 8.8, "Troubleshooting"*.

CUPS (Common Unix Printing System) is the standard print system in openSUSE Leap.

Printers can be distinguished by interface, such as USB or network, and printer language. When buying a printer, make sure that the printer has an interface that is supported (USB, Ethernet, or Wi-Fi) and a suitable printer language. Printers can be categorized on the basis of the following three classes of printer languages:

PostScript Printers

PostScript is the printer language in which most print jobs in Linux and Unix are generated and processed by the internal print system. If PostScript documents can be processed directly by the printer and do not need to be converted in additional stages in the print system, the number of potential error sources is reduced.

Currently PostScript is being replaced by PDF as the standard print job format. PostScript + PDF printers that can directly print PDF (in addition to PostScript) already exist. For traditional PostScript printers PDF needs to be converted to PostScript in the printing workflow.

Standard Printers (Languages Like PCL and ESC/P)

In the case of known printer languages, the print system can convert PostScript jobs to the respective printer language with Ghostscript. This processing stage is called interpreting. The best-known languages are PCL (which is mostly used by HP printers and their clones) and ESC/P (which is used by Epson printers). These printer languages are usually supported by Linux and produce an adequate print result. Linux may not be able to address some special printer functions. Except for HP and Epson, there are currently no printer manufacturers who develop Linux drivers and make them available to Linux distributors under an open source license.

Proprietary Printers (Also Called GDI Printers)

These printers do not support any of the common printer languages. They use their own undocumented printer languages, which are subject to change when a new edition of a model is released. Usually only Windows drivers are available for these printers. See *Section 8.8.1, "Printers without Standard Printer Language Support"* for more information.

Before you buy a new printer, refer to the following sources to check how well the printer you intend to buy is supported:

http://www.linuxfoundation.org/OpenPrinting/

The OpenPrinting home page with the printer database. The database shows the latest Linux support status. However, a Linux distribution can only integrate the drivers available at production time. Accordingly, a printer currently rated as "perfectly supported" may not have had this status when the latest openSUSE Leap version was released. Thus, the databases may not necessarily indicate the correct status, but only provide an approximation.

http://pages.cs.wisc.edu/~ghost/

The Ghostscript Web page.

`/usr/share/doc/packages/ghostscript/catalog.devices`

List of built-in Ghostscript drivers.

8.1 The CUPS Workflow

The user creates a print job. The print job consists of the data to print plus information for the spooler. This includes the name of the printer or the name of the print queue, and optionally, information for the filter, such as printer-specific options.

At least one dedicated print queue exists for every printer. The spooler holds the print job in the queue until the desired printer is ready to receive data. When the printer is ready, the spooler sends the data through the filter and back-end to the printer.

The filter converts the data generated by the application that is printing (usually PostScript or PDF, but also ASCII, JPEG, etc.) into printer-specific data (PostScript, PCL, ESC/P, etc.). The features of the printer are described in the PPD files. A PPD file contains printer-specific options with the parameters needed to enable them on the printer. The filter system makes sure that options selected by the user are enabled.

If you use a PostScript printer, the filter system converts the data into printer-specific PostScript. This does not require a printer driver. If you use a non-PostScript printer, the filter system converts the data into printer-specific data. This requires a printer driver suitable for your printer. The back-end receives the printer-specific data from the filter then passes it to the printer.

8.2 Methods and Protocols for Connecting Printers

There are various possibilities for connecting a printer to the system. The configuration of CUPS does not distinguish between a local printer and a printer connected to the system over the network. For more information about the printer connection, read the article *CUPS in a Nutshell* at http://en.opensuse.org/SDB:CUPS_in_a_Nutshell.

Warning: Changing Cable Connections in a Running System

When connecting the printer to the machine, do not forget that only USB devices can be plugged in or unplugged during operation. To avoid damaging your system or printer, shut down the system before changing any connections that are not USB.

8.3 Installing the Software

PPD (PostScript printer description) is the computer language that describes the properties, like resolution, and options, such as the availability of a duplex unit. These descriptions are required for using various printer options in CUPS. Without a PPD file, the print data would be forwarded to the printer in a "raw" state, which is usually not desired.

To configure a PostScript printer, the best approach is to get a suitable PPD file. Many PPD files are available in the packages `manufacturer-PPDs` and `OpenPrintingPPDs-postscript`. See *Section 8.7.3, "PPD Files in Various Packages"* and *Section 8.8.2, "No Suitable PPD File Available for a PostScript Printer"*.

New PPD files can be stored in the directory `/usr/share/cups/model/` or added to the print system with YaST as described in *Section 7.3.1.1, "Adding Drivers with YaST"*. Subsequently, the PPD file can be selected during the printer setup.

Be careful if a printer manufacturer wants you to install entire software packages. This kind of installation may result in the loss of the support provided by openSUSE Leap. Also, print commands may work differently and the system may no longer be able to address devices of other manufacturers. For this reason, the installation of manufacturer software is not recommended.

8.4 Network Printers

A network printer can support various protocols, some even concurrently. Although most of the supported protocols are standardized, some manufacturers modify the standard. Manufacturers then provide drivers for only a few operating systems. Unfortunately, Linux drivers are rarely provided. The current situation is such that you cannot act on the assumption that every protocol works smoothly in Linux. Therefore, you may need to experiment with various options to achieve a functional configuration.

CUPS supports the `socket`, `LPD`, `IPP` and `smb` protocols.

socket

Socket refers to a connection in which the plain print data is sent directly to a TCP socket. Some socket port numbers that are commonly used are `9100` or `35`. The device URI (uniform resource identifier) syntax is: socket://*`IP.OF.THE.PRINTER`*:*`PORT`*, for example: `socket://192.168.2.202:9100/`.

LPD (Line Printer Daemon)

The LPD protocol is described in RFC 1179. Under this protocol, some job-related data, such as the ID of the print queue, is sent before the actual print data is sent. Therefore, a print queue must be specified when configuring the LPD protocol. The implementations of diverse printer manufacturers are flexible enough to accept any name as the print queue. If necessary, the printer manual should indicate what name to use. LPT, LPT1, LP1 or similar names are often used. The port number for an LPD service is `515`. An example device URI is `lpd://192.168.2.202/LPT1`.

IPP (Internet Printing Protocol)

IPP is based on the HTTP protocol. With IPP, more job-related data is transmitted than with the other protocols. CUPS uses IPP for internal data transmission. The name of the print queue is necessary to configure IPP correctly. The port number for IPP is `631`. Example device URIs are `ipp://192.168.2.202/ps` and `ipp://192.168.2.202/printers/ps`.

SMB (Windows Share)

CUPS also supports printing on printers connected to Windows shares. The protocol used for this purpose is SMB. SMB uses the port numbers `137`, `138` and `139`. Example device URIs are `smb://user:password@workgroup/smb.example.com/printer`, `smb://user:password@smb.example.com/printer`, and `smb://smb.example.com/printer`.

The protocol supported by the printer must be determined before configuration. If the manufacturer does not provide the needed information, the command **`nmap`** (which comes with the `nmap` package) can be used to ascertain the protocol. **`nmap`** checks a host for open ports. For example:

```
tux > nmap -p 35,137-139,515,631,9100-10000 IP.OF.THE.PRINTER
```

8.5 Configuring CUPS with Command Line Tools

CUPS can be configured with command line tools like **`lpinfo`**, **`lpadmin`** and **`lpoptions`**. You need a device URI consisting of a back-end, such as USB, and parameters. To determine valid device URIs on your system use the command **`lpinfo -v | grep ":/"`**:

```
tux > sudo lpinfo -v | grep ":/"
direct usb://ACME/FunPrinter%20XL
network socket://192.168.2.253
```

With **`lpadmin`** the CUPS server administrator can add, remove or manage print queues. To add a print queue, use the following syntax:

```
tux > sudo lpadmin -p QUEUE -v DEVICE-URI -P PPD-FILE -E
```

Then the device (`-v`) is available as *`QUEUE`* (`-p`), using the specified PPD file (`-P`). This means that you must know the PPD file and the device URI to configure the printer manually.

Do not use `-E` as the first option. For all CUPS commands, `-E` as the first argument sets use of an encrypted connection. To enable the printer, `-E` must be used as shown in the following example:

```
tux > sudo lpadmin -p ps -v usb://ACME/FunPrinter%20XL -P \
/usr/share/cups/model/Postscript.ppd.gz -E
```

The following example configures a network printer:

```
tux > sudo lpadmin -p ps -v socket://192.168.2.202:9100/ -P \
/usr/share/cups/model/Postscript-level1.ppd.gz -E
```

For more options of **lpadmin**, see the man page of `lpadmin(8)`.

During printer setup, certain options are set as default. These options can be modified for every print job (depending on the print tool used). Changing these default options with YaST is also possible. Using command line tools, set default options as follows:

1. First, list all options:

   ```
   tux > sudo lpoptions -p QUEUE -l
   ```

   Example:

   ```
   Resolution/Output Resolution: 150dpi *300dpi 600dpi
   ```

   The activated default option is identified by a preceding asterisk (`*`).

2. Change the option with **lpadmin**:

   ```
   tux > sudo lpadmin -p QUEUE -o Resolution=600dpi
   ```

3. Check the new setting:

   ```
   tux > sudo lpoptions -p QUEUE -l

   Resolution/Output Resolution: 150dpi 300dpi *600dpi
   ```

When a normal user runs **lpoptions**, the settings are written to `~/.cups/lpoptions`. However, `root` settings are written to `/etc/cups/lpoptions`.

8.6 Printing from the Command Line

To print from the command line, enter **lp -d** *QUEUENAME* *FILENAME*, substituting the corresponding names for *QUEUENAME* and *FILENAME*.

Some applications rely on the **lp** command for printing. In this case, enter the correct command in the application's print dialog, usually without specifying *FILENAME*, for example, **lp -d** *QUEUENAME*.

8.7 Special Features in openSUSE Leap

Several CUPS features have been adapted for openSUSE Leap. Some of the most important changes are covered here.

8.7.1 CUPS and Firewall

After completing a default installation of openSUSE Leap, `firewalld` is active and the network interfaces are configured to be in the `public` zone, which blocks incoming traffic.

When `firewalld` is active, you may need to configure it to allow clients to browse network printers by allowing `mdns` and `ipp` through the internal network zone. The public zone should never expose printer queues.

(More information about the `firewalld` configuration is available in *Book "*Security Guide*", Chapter 16 "Masquerading and Firewalls", Section 16.4 "*`firewalld`*"* and at http://en.opensuse.org/SDB:CUPS_and_SANE_Firewall_settings.)

8.7.1.1 CUPS Client

Normally, a CUPS client runs on a regular workstation located in a trusted network environment behind a firewall. In this case it is recommended to configure the network interface to be in the `Internal Zone`, so the workstation is reachable from within the network.

8.7.1.2 CUPS Server

If the CUPS server is part of a trusted network environment protected by a firewall, the network interface should be configured to be in the `Internal Zone` of the firewall. It is not recommended to set up a CUPS server in an untrusted network environment unless you ensure that it is protected by special firewall rules and secure settings in the CUPS configuration.

8.7.2 Browsing for Network Printers

CUPS servers regularly announce the availability and status information of shared printers over the network. Clients can access this information to display a list of available printers in printing dialogs, for example. This is called "browsing".

CUPS servers announce their print queues over the network either via the traditional CUPS browsing protocol, or via Bonjour/DNS-SD. To enable browsing network print queues, the service `cups-browsed` needs to run on all clients that print via CUPS servers. `cups-browsed` is not started by default. To start it for the active session, use **`sudo systemctl start cups-browsed`**. To ensure it is automatically started after booting, enable it with **`sudo systemctl enable cups-browsed`** on all clients.

In case browsing does not work after having started `cups-browsed`, the CUPS server(s) probably announce the network print queues via Bonjour/DNS-SD. In this case you need to additionally install the package `avahi` and start the associated service with **`sudo systemctl start avahi-daemon`** on all clients.

See *Section 8.7.1, "CUPS and Firewall"* for information on allowing printer browsing through `firewalld`.

8.7.3 PPD Files in Various Packages

The YaST printer configuration sets up the queues for CUPS using the PPD files installed in `/usr/share/cups/model`. To find the suitable PPD files for the printer model, YaST compares the vendor and model determined during hardware detection with the vendors and models in all PPD files. For this purpose, the YaST printer configuration generates a database from the vendor and model information extracted from the PPD files.

The configuration using only PPD files and no other information sources has the advantage that the PPD files in `/usr/share/cups/model` can be modified freely. For example, if you have PostScript printers the PPD files can be copied directly to `/usr/share/cups/model` (if they do not already exist in the `manufacturer-PPDs` or `OpenPrintingPPDs-postscript` packages) to achieve an optimum configuration for your printers.

Additional PPD files are provided by the following packages:

- `gutenprint`: the Gutenprint driver and its matching PPDs
- `splix`: the SpliX driver and its matching PPDs
- `OpenPrintingPPDs-ghostscript`: PPDs for Ghostscript built-in drivers
- `OpenPrintingPPDs-hpijs`: PPDs for the HPIJS driver for non-HP printers

8.8 Troubleshooting

The following sections cover some of the most frequently encountered printer hardware and software problems and ways to solve or circumvent these problems. Among the topics covered are GDI printers, PPD files and port configuration. Common network printer problems, defective printouts, and queue handling are also addressed.

8.8.1 Printers without Standard Printer Language Support

These printers do not support any common printer language and can only be addressed with special proprietary control sequences. Therefore they can only work with the operating system versions for which the manufacturer delivers a driver. GDI is a programming interface developed by Microsoft* for graphics devices. Usually the manufacturer delivers drivers only for Windows, and since the Windows driver uses the GDI interface these printers are also called *GDI printers*. The actual problem is not the programming interface, but that these printers can only be addressed with the proprietary printer language of the respective printer model.

Some GDI printers can be switched to operate either in GDI mode or in one of the standard printer languages. See the manual of the printer whether this is possible. Some models require special Windows software to do the switch (note that the Windows printer driver may always switch the printer back into GDI mode when printing from Windows). For other GDI printers there are extension modules for a standard printer language available.

Some manufacturers provide proprietary drivers for their printers. The disadvantage of proprietary printer drivers is that there is no guarantee that these work with the installed print system or that they are suitable for the various hardware platforms. In contrast, printers that support a standard printer language do not depend on a special print system version or a special hardware platform.

Instead of spending time trying to make a proprietary Linux driver work, it may be more cost-effective to purchase a printer which supports a standard printer language (preferably PostScript). This would solve the driver problem once and for all, eliminating the need to install and configure special driver software and obtain driver updates that may be required because of new developments in the print system.

8.8.2 No Suitable PPD File Available for a PostScript Printer

If the `manufacturer-PPDs` or `OpenPrintingPPDs-postscript` packages do not contain a suitable PPD file for a PostScript printer, it should be possible to use the PPD file from the driver CD of the printer manufacturer or download a suitable PPD file from the Web page of the printer manufacturer.

If the PPD file is provided as a zip archive (.zip) or a self-extracting zip archive (`.exe`), unpack it with **`unzip`**. First, review the license terms of the PPD file. Then use the **`cupstestppd`** utility to check if the PPD file complies with "Adobe PostScript Printer Description File Format Specification, version 4.3." If the utility returns "FAIL," the errors in the PPD files are serious and are likely to cause major problems. The problem spots reported by **`cupstestppd`** should be eliminated. If necessary, ask the printer manufacturer for a suitable PPD file.

8.8.3 Network Printer Connections

Identifying Network Problems

Connect the printer directly to the computer. For test purposes, configure the printer as a local printer. If this works, the problems are related to the network.

Checking the TCP/IP Network

The TCP/IP network and name resolution must be functional.

Checking a Remote **`lpd`**

Use the following command to test if a TCP connection can be established to **`lpd`** (port `515`) on *`HOST`*:

```
tux > netcat -z HOST 515 && echo ok || echo failed
```

If the connection to **`lpd`** cannot be established, **`lpd`** may not be active or there may be basic network problems.

Provided that the respective **`lpd`** is active and the host accepts queries, run the following command as `root` to query a status report for *`QUEUE`* on remote *`HOST`*:

```
root # echo -e "\004queue" \
| netcat -w 2 -p 722 HOST 515
```

If **`lpd`** does not respond, it may not be active or there may be basic network problems. If **`lpd`** responds, the response should show why printing is not possible on the `queue` on `host`. If you receive a response like that shown in *Example 8.1, "Error Message from* **`lpd`***"*, the problem is caused by the remote **`lpd`**.

EXAMPLE 8.1: ERROR MESSAGE FROM `lpd`

```
lpd: your host does not have line printer access
lpd: queue does not exist
printer: spooling disabled
printer: printing disabled
```

Checking a Remote `cupsd`

A CUPS network server can broadcast its queues by default every 30 seconds on UDP port `631`. Accordingly, the following command can be used to test whether there is a broadcasting CUPS network server in the network. Make sure to stop your local CUPS daemon before executing the command.

```
tux > netcat -u -l -p 631 & PID=$! ; sleep 40 ; kill $PID
```

If a broadcasting CUPS network server exists, the output appears as shown in *Example 8.2, "Broadcast from the CUPS Network Server"*.

EXAMPLE 8.2: BROADCAST FROM THE CUPS NETWORK SERVER

```
ipp://192.168.2.202:631/printers/queue
```

The following command can be used to test if a TCP connection can be established to **`cupsd`** (port `631`) on *`HOST`*:

```
tux > netcat -z HOST 631 && echo ok || echo failed
```

If the connection to **`cupsd`** cannot be established, **`cupsd`** may not be active or there may be basic network problems. **`lpstat -h`** *`HOST`* -l -t returns a (possibly very long) status report for all queues on *`HOST`*, provided the respective **`cupsd`** is active and the host accepts queries.

The next command can be used to test if the *`QUEUE`* on *`HOST`* accepts a print job consisting of a single carriage-return character. Nothing should be printed. Possibly, a blank page may be ejected.

```
tux > echo -en "\r" \
| lp -d queue -h HOST
```

Troubleshooting a Network Printer or Print Server Machine

Spoolers running in a print server machine sometimes cause problems when they need to deal with multiple print jobs. Since this is caused by the spooler in the print server machine, there no way to resolve this issue. As a work-around, circumvent the spooler in the print server machine by addressing the printer connected to the print server machine directly with the TCP socket. See *Section 8.4, "Network Printers"*.

In this way, the print server machine is reduced to a converter between the various forms of data transfer (TCP/IP network and local printer connection). To use this method, you need to know the TCP port on the print server machine. If the printer is connected to the print server machine and turned on, this TCP port can usually be determined with the **nmap** utility from the `nmap` package some time after the print server machine is powered up. For example, **nmap** *IP-address* may deliver the following output for a print server machine:

```
Port       State       Service
23/tcp     open        telnet
80/tcp     open        http
515/tcp    open        printer
631/tcp    open        cups
9100/tcp   open        jetdirect
```

This output indicates that the printer connected to the print server machine can be addressed via TCP socket on port `9100`. By default, **nmap** only checks several commonly known ports listed in `/usr/share/nmap/nmap-services`. To check all possible ports, use the command **nmap -p** *FROM_PORT-TO_PORT IP_ADDRESS*. This may take some time. For further information, refer to the man page of **nmap**.

Enter a command like

```
tux > echo -en "\rHello\r\f" | netcat -w 1 IP-address port
cat file | netcat -w 1 IP-address port
```

to send character strings or files directly to the respective port to test if the printer can be addressed on this port.

8.8.4 Defective Printouts without Error Message

For the print system, the print job is completed when the CUPS back-end completes the data transfer to the recipient (printer). If further processing on the recipient fails (for example, if the printer is not able to print the printer-specific data) the print system does not notice this. If the printer cannot print the printer-specific data, select a PPD file that is more suitable for the printer.

8.8.5 Disabled Queues

If the data transfer to the recipient fails entirely after several attempts, the CUPS back-end, such as `USB` or `socket`, reports an error to the print system (to **`cupsd`**). The back-end determines how many unsuccessful attempts are appropriate until the data transfer is reported as impossible. As further attempts would be in vain, **`cupsd`** disables printing for the respective queue. After eliminating the cause of the problem, the system administrator must re-enable printing with the command **`cupsenable`**.

8.8.6 CUPS Browsing: Deleting Print Jobs

If a CUPS network server broadcasts its queues to the client hosts via browsing and a suitable local **`cupsd`** is active on the client hosts, the client **`cupsd`** accepts print jobs from applications and forwards them to the **`cupsd`** on the server. When **`cupsd`** on the server accepts a print job, it is assigned a new job number. Therefore, the job number on the client host is different from the job number on the server. As a print job is usually forwarded immediately, it cannot be deleted with the job number on the client host This is because the client **`cupsd`** regards the print job as completed when it has been forwarded to the server **`cupsd`**.

To delete the print job on the server, use a command such as **`lpstat -h cups.example.com -o`** to determine the job number on the server. This assumes that the server has not already completed the print job (that is, sent it completely to the printer). Use the obtained job number to delete the print job on the server as follows:

```
tux > cancel -h cups.example.com QUEUE-JOBNUMBER
```

8.8.7 Defective Print Jobs and Data Transfer Errors

If you switch the printer off or shut down the computer during the printing process, print jobs remain in the queue. Printing resumes when the computer (or the printer) is switched back on. Defective print jobs must be removed from the queue with **`cancel`**.

If a print job is corrupted or an error occurs in the communication between the host and the printer, the printer cannot process the data correctly and prints numerous sheets of paper with unintelligible characters. To fix the problem, follow these steps:

1. To stop printing, remove all paper from ink jet printers or open the paper trays of laser printers. High-quality printers have a button for canceling the current printout.

2. The print job may still be in the queue, because jobs are only removed after they are sent completely to the printer. Use **`lpstat -o`** or **`lpstat -h cups.example.com -o`** to check which queue is currently printing. Delete the print job with **`cancel`** *QUEUE*-*JOBNUMBER* or **`cancel -h cups.example.com`** *QUEUE*-*JOBNUMBER*.
3. Some data may still be transferred to the printer even though the print job has been deleted from the queue. Check if a CUPS back-end process is still running for the respective queue and terminate it.
4. Reset the printer completely by switching it off for some time. Then insert the paper and turn on the printer.

8.8.8 Debugging CUPS

Use the following generic procedure to locate problems in CUPS:

1. Set **`LogLevel debug`** in `/etc/cups/cupsd.conf`.
2. Stop **`cupsd`**.
3. Remove `/var/log/cups/error_log*` to avoid having to search through very large log files.
4. Start **`cupsd`**.
5. Repeat the action that led to the problem.
6. Check the messages in `/var/log/cups/error_log*` to identify the cause of the problem.

8.8.9 For More Information

In-depth information about printing on openSUSE Leap is presented in the openSUSE Support Database at http://en.opensuse.org/Portal:Printing.

9 Accessing File Systems with FUSE

FUSE is the acronym for *file system in user space*. This means you can configure and mount a file system as an unprivileged user. Normally, you need to be `root` for this task. FUSE alone is a kernel module. Combined with plug-ins, it allows you to extend FUSE to access almost all file systems like remote SSH connections, ISO images, and more.

9.1 Configuring FUSE

Before you can use FUSE, you need to install the package `fuse`. Depending which file system you want to use, you need additional plug-ins available as separate packages. For an overview, see *Section 9.5, "Available FUSE Plug-ins"*.

Generally you do not need to configure FUSE. However, it is a good idea to create a directory where all your mount points are combined. For example, you can create a directory `~/mounts` and insert your subdirectories for your different file systems there.

9.2 Mounting an NTFS Partition

NTFS, the *New Technology File System*, is the default file system of Windows. Since under normal circumstances the unprivileged user cannot mount NTFS block devices using the external FUSE library, the process of mounting a Windows partition described below requires root privileges.

1. Become `root` and install the package `ntfs-3g`.
2. Create a directory that is to be used as a mount point, for example `~/mounts/windows`.
3. Find out which Windows partition you need. Use YaST and start the partitioner module to see which partition belongs to Windows, but do not modify anything. Alternatively, become `root` and execute `/sbin/fdisk -l`. Look for partitions with a partition type of `HPFS/NTFS`.
4. Mount the partition in read-write mode. Replace the placeholder *`DEVICE`* with your respective Windows partition:

```
tux > ntfs-3g /dev/DEVICE MOUNT POINT
```

To use your Windows partition in read-only mode, append `-o ro`:

```
tux > ntfs-3g /dev/DEVICE MOUNT POINT -o ro
```

The command **`ntfs-3g`** uses the current user (UID) and group (GID) to mount the given device. If you want to set the write permissions to a different user, use the command **`id`** `USER` to get the output of the UID and GID values. Set it with:

```
root # id tux
uid=1000(tux) gid=100(users) groups=100(users),16(dialout),33(video)
ntfs-3g /dev/DEVICE MOUNT POINT -o uid=1000,gid=100
```

Find additional options in the man page.

To unmount the resource, run **`fusermount -u`** *MOUNT POINT*.

9.3 Mounting Remote File System with SSHFS

SSH, the secure shell network protocol, can be used to exchange data between two computers using a secure channel. To establish an SSH connection through FUSE, proceed as follows:

1. Install the package `sshfs`.
2. Create a directory that is to be used as a mount point. A good idea is to use `~/mounts/`*HOST*. Replace *HOST* with the name of your remote computer.
3. Mount the remote file system:

   ```
   root # sshfs USER@HOST MOUNT POINT
   ```

4. Enter your password for the remote computer.

To unmount the resource, run **`fusermount -u`** *MOUNT POINT*.

9.4 Mounting an ISO File System

To look into an ISO image, you can mount it with the `fuseiso` package:

1. Install the package `fuseiso`.

2. Create a directory that is to be used as a mount point, for example `~/mounts/iso`.

3. Mount the ISO image:

```
root # fuseiso ISO_IMAGE MOUNT POINT
```

You can only read content from the ISO image, but you can not write back. To unmount the resource, use **`fusermount -u`** *`MOUNT POINT`*.

9.5 Available FUSE Plug-ins

FUSE is dependent on plug-ins. The following table lists common plug-ins.

TABLE 9.1: AVAILABLE FUSE PLUG-INS

`curlftpfs`	mount FTP servers
`encfs`	mount encrypted file systems
`fuseiso`	mounts CD-ROM images with ISO9660 file systems in them
`fusepod`	mount iPods
`fusesmb`	mount browseable Samba clients or Windows shares
`gphotofs`	mount supported digital cameras through gPhoto
`ntfs-3g`	mount NTFS volumes (with read and write support)
`obexfs`	mount Bluetooth devices
`sshfs`	file system client based on SSH file transfer protocol
`wdfs`	mount WebDAV file systems

9.6 For More Information

See the home page http://fuse.sourceforge.net of FUSE for more information.

III Managing and Updating Software

10 Installing or Removing Software

Use YaST's software management module to search for software components you want to add or remove. YaST resolves all dependencies for you. To install packages not shipped with the installation media, add software repositories to your setup and let YaST manage them. Keep your system up-to-date by managing software updates with the update applet.

Change the software collection of your system with the YaST Software Manager. This YaST module is available in two flavors: a graphical variant for X Window and a text-based variant to be used on the command line. The graphical flavor is described here—for details on the text-based YaST, see *Book "Reference", Chapter 1 "YaST in Text Mode"*.

Note: Confirmation and Review of Changes

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

10.1 Definition of Terms

The following terms are important for understanding installing and removing software in openSUSE Leap.

Repository

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

(Repository) Alias/Repository Name

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

Repository Description Files

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

Product

Represents a whole product, for example openSUSE® Leap.

Pattern

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

Package

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

Patch

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

Resolvable

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

Delta RPM

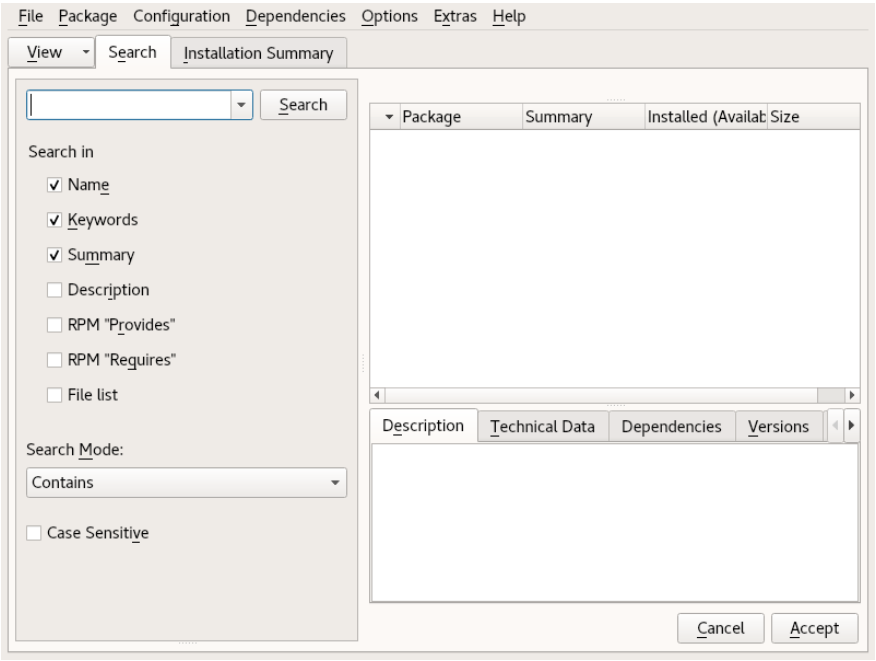
A delta RPM consists only of the binary diff between two defined versions of a package, and therefore has the smallest download size. Before being installed, the full RPM package is rebuilt on the local machine.

Package Dependencies

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

10.2 Using the YaST Software Manager

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



10.2.1 Views for Searching Packages or Patterns

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

Patterns

Lists all patterns available for installation on your system.

Package Groups

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

Languages

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

Repositories

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

Services

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

Search

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

Installation Summary

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

Tip: Finding Packages Not Belonging to an Active Repository

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

10.2.2 Installing and Removing Packages or Patterns

Certain packages are dependent on other packages, such as shared libraries. On the other hand, some packages cannot coexist with others on the system. If possible, YaST automatically resolves these dependencies or conflicts. If your choice results in a dependency conflict that cannot be automatically solved, you need to solve it manually as described in *Section 10.2.4, "Package Dependencies"*.

Note: Removal of Packages

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

1. Search for packages as described in *Section 10.2.1, "Views for Searching Packages or Patterns"*.
2. The packages found are listed in the right pane. To install a package or remove it, right-click it and choose *Install* or *Delete*. If the relevant option is not available, check the package status indicated by the symbol in front of the package name—press Shift–F1 for help.

Tip: Applying an Action to All Packages Listed

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

3. To install a pattern, right-click the pattern name and choose *Install*.
4. It is not possible to remove a pattern. Instead, select the packages of a pattern you want to remove and mark them for removal.
5. To select more packages, repeat the steps mentioned above.
6. Before applying your changes, you can review or modify them by clicking *View* › *Installation Summary*. By default, all packages that will change status, are listed.
7. To revert the status for a package, right-click the package and select one of the following entries: *Keep* if the package was scheduled to be deleted or updated, or *Do Not Install* if it was scheduled for installation. To abandon all changes and quit the Software Manager, click *Cancel* and *Abandon*.
8. When you are finished, click *Accept* to apply your changes.
9. In case YaST found dependencies on other packages, a list of packages that have additionally been chosen for installation, update or removal is presented. Click *Continue* to accept them.
   After all selected packages are installed, updated or removed, the YaST Software Manager automatically terminates.

Note: Installing Source Packages

Installing source packages with YaST Software Manager is not possible at the moment. Use the command line tool **zypper** for this purpose. For more information, see *Book "Reference", Chapter 2 "Managing Software with Command Line Tools", Section 2.1.2.5 "Installing or Downloading Source Packages"*.

10.2.3 Updating Packages

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

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

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

1. To update all installed packages to the latest version, choose *Package* › *All Packages* › *Update if Newer Version Available* from the main menu.
   All repositories are checked for possible update candidates, using the following policy: YaST first tries to restrict the search to packages with the same architecture and vendor like the installed one. If the search is positive, the "best" update candidate from those is selected according to the process below. However, if no comparable package of the same vendor can be found, the search is expanded to all packages with the same architecture. If still no comparable package can be found, all packages are considered and the "best" update candidate is selected according to the following criteria:

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

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

Note: Update Unconditionally

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

2. To make sure that the packages for a mass update derive from a certain repository:

   a. Choose the repository from which to update as described in *Section 10.2.1, "Views for Searching Packages or Patterns"* .

   b. On the right hand side of the window, click *Switch system packages to the versions in this repository*. This explicitly allows YaST to change the package vendor when replacing the packages.
   When you proceed with *Accept*, all installed packages will be replaced by packages deriving from this repository, if available. This may lead to changes in vendor and architecture and even to downgrading some packages.

   c. To refrain from this, click *Cancel switching system packages to the versions in this repository*. Note that you can only cancel this until you click the *Accept* button.

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

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

10.2.4 Package Dependencies

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

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

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

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

Note: Manually Solving Package Conflicts

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

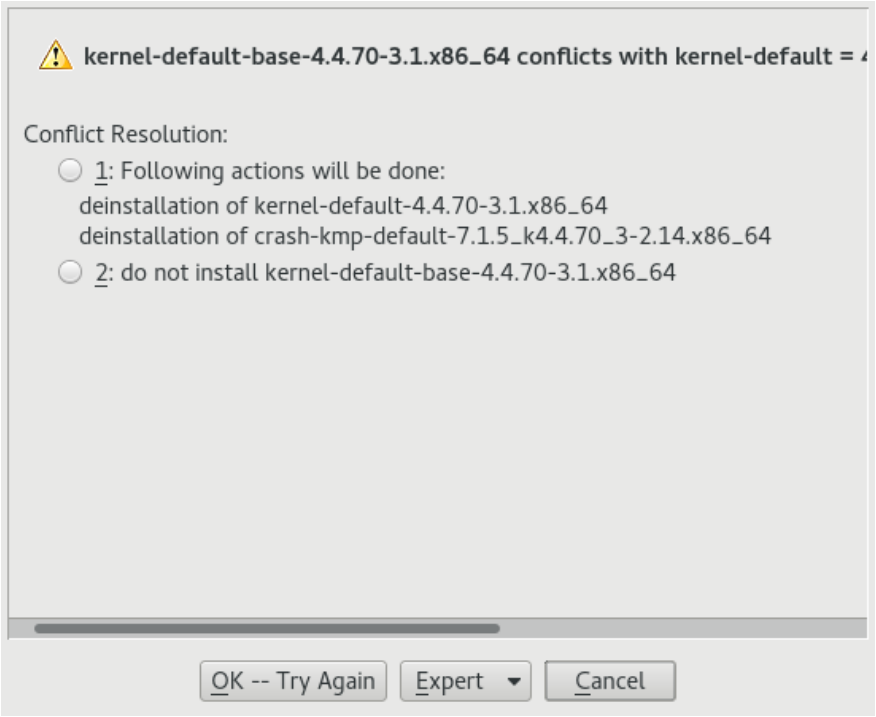


FIGURE 10.1: CONFLICT MANAGEMENT OF THE SOFTWARE MANAGER

10.2.5 Handling of Package Recommendations

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

The way package recommendations are handled has slightly changed starting with openSUSE Leap 42.1. Nothing has changed when installing a new package—recommended packages are still installed by default.

Prior to openSUSE Leap 42.1, missing recommendations for already installed packages were installed automatically. Now these packages will no longer be installed automatically. To switch to the old default, set `PKGMGR_REEVALUATE_RECOMMENDED="yes"` in `/etc/sysconfig/yast2`. To install all missing recommendations for already installed packages, start *YaST* › *Software Manager* and choose *Extras* › *Install All Matching Recommended Packages*.

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

10.3 Managing Software Repositories and Services

To install third-party software, add software repositories to your system. By default, the product repositories such as openSUSE Leap-DVD 15.1 and a matching update repository are automatically configured. Depending on the initially selected product, an additional repository containing translations, dictionaries, etc. might also be configured.

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

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

Warning: Trusting External Software Sources

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

10.3.1 Adding Software Repositories

You can either add repositories from DVD/CD, removable mass storage devices (such as flash disks), a local directory, an ISO image or a network source.

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

1. Click *Add*.

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

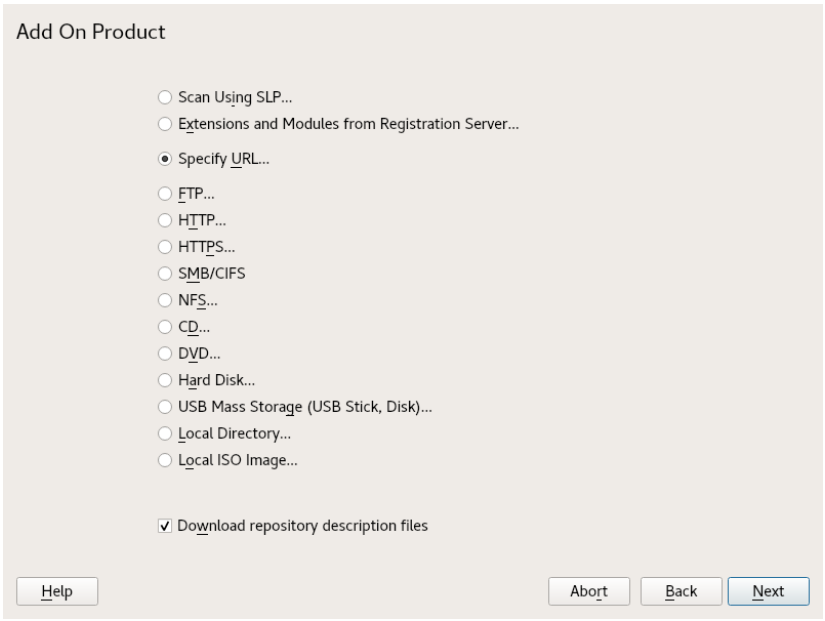


FIGURE 10.2: ADDING A SOFTWARE REPOSITORY

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

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

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

4. If needed, adjust the repository *Properties* as described in *Section 10.3.2, "Managing Repository Properties"*.

5. Confirm your changes with *OK* to close the configuration dialog.

6. After having successfully added the repository, the software manager starts and you can install packages from this repository. For details, refer to *Chapter 10, Installing or Removing Software*.

10.3.2 Managing Repository Properties

The *Configured Software Repositories* overview of the *Software Repositories* lets you change the following repository properties:

Status

The repository status can either be *Enabled* or *Disabled*. You can only install packages from repositories that are enabled. To turn a repository off temporarily, select it and deactivate *Enable*. You can also double-click a repository name to toggle its status. To remove a repository completely, click *Delete*.

Refresh

When refreshing a repository, its content description (package names, versions, etc.) is downloaded to a local cache that is used by YaST. It is sufficient to do this once for static repositories such as CDs or DVDs, whereas repositories whose content changes often should be refreshed frequently. The easiest way to keep a repository's cache up-to-date is to choose *Automatically Refresh*. To do a manual refresh click *Refresh* and select one of the options.

Keep Downloaded Packages

Packages from remote repositories are downloaded before being installed. By default, they are deleted upon a successful installation. Activating *Keep Downloaded Packages* prevents the deletion of downloaded packages. The download location is configured in `/etc/zypp/zypp.conf`, by default it is `/var/cache/zypp/packages`.

Priority

The *Priority* of a repository is a value between `1` and `200`, with `1` being the highest priority and `200` the lowest priority. Any new repositories that are added with YaST get a priority of `99` by default. If you do not care about a priority value for a certain repository, you can also set the value to `0` to apply the default priority to that repository (`99`). If a package is available in more than one repository, then the repository with the highest priority takes precedence. This is useful to avoid downloading packages unnecessarily from the Internet by giving a local repository (for example, a DVD) a higher priority.

Important: Priority Compared to Version

The repository with the highest priority takes precedence in any case. Therefore, make sure that the update repository always has the highest priority, otherwise you might install an outdated version that will not be updated until the next online update.

Name and URL

To change a repository name or its URL, select it from the list with a single-click and then click *Edit*.

10.3.3 Managing Repository Keys

To ensure their integrity, software repositories can be signed with the GPG Key of the repository maintainer. Whenever you add a new repository, YaST offers to import its key. Verify it as you would do with any other GPG key and make sure it does not change. If you detect a key change, something might be wrong with the repository. Disable the repository as an installation source until you know the cause of the key change.

To manage all imported keys, click *GPG Keys* in the *Configured Software Repositories* dialog. Select an entry with the mouse to show the key properties at the bottom of the window. *Add*, *Edit* or *Delete* keys with a click on the respective buttons.

10.4 The GNOME Package Updater

SUSE offers a continuous stream of software security patches and updates for your product. They can be installed using tools available with your desktop or by running the *YaST Online Update* module. This section describes how to update the system from the GNOME desktop using the *Package Updater*.

Contrary to the YaST Online Update module, the GNOME *Package Updater* not only offers to install patches from the update repositories, but also new versions of packages that are already installed. (Patches fix security issues or malfunctions; the functionality and version number is usually not changed. New versions of a package increase the version number and usually add functionality or introduce major changes).

Whenever new patches or package updates are available, GNOME shows a notification in the notification area or on the lock screen.

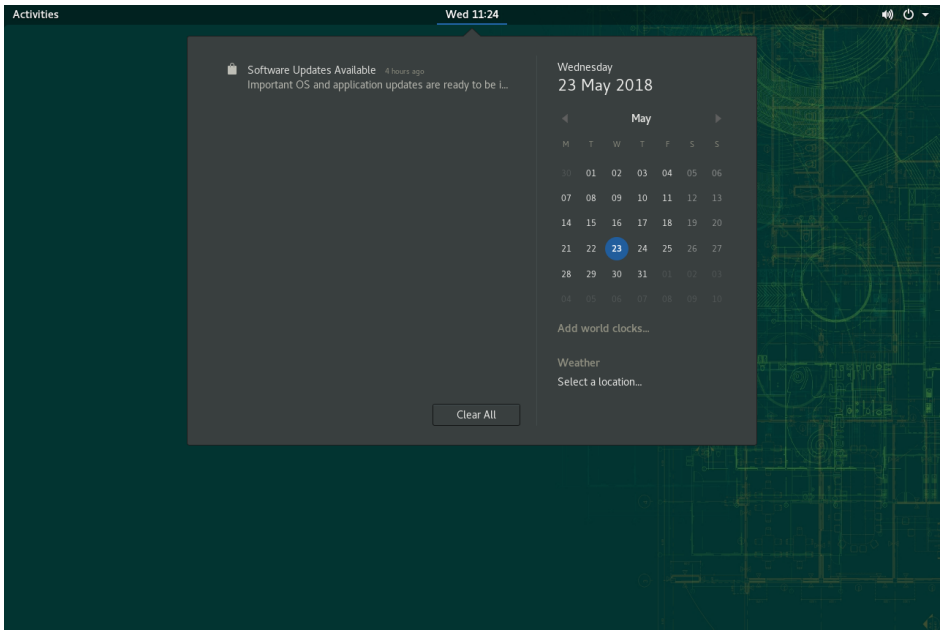
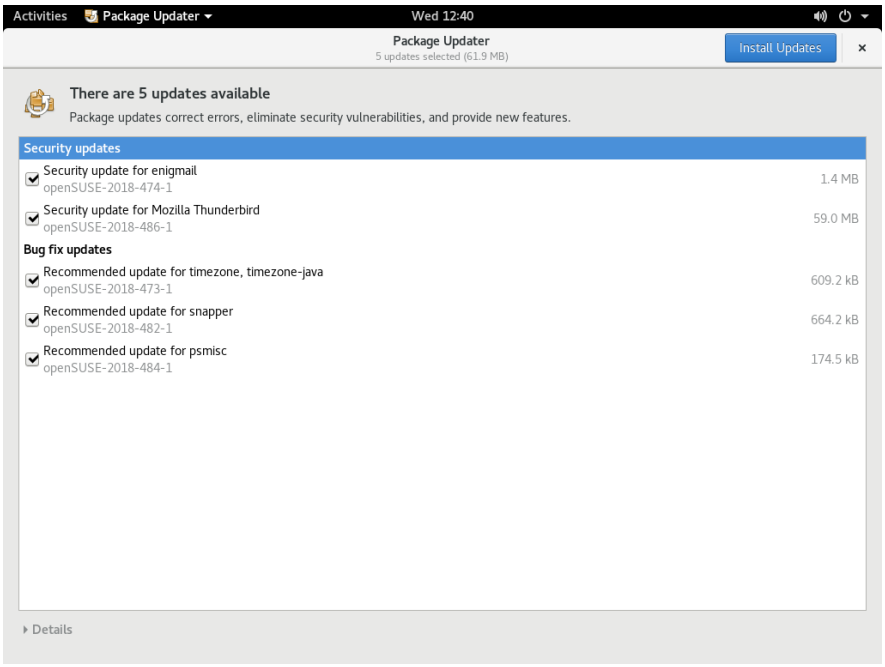


FIGURE 10.3: UPDATE NOTIFICATION ON GNOME DESKTOP

To configure the notification settings for the *Package Updater*, start GNOME *Settings* and choose *Notifications* › *Package Update*.

PROCEDURE 10.1: INSTALLING PATCHES AND UPDATES WITH THE GNOME PACKAGE UPDATER

1. To install the patches and updates, click the notification message. This opens the GNOME *Package Updater*. Alternatively, open the updater from *Activities* by typing `package U` and choosing *Package Updater*.



2. Updates are sorted into four categories:

   Security Updates (Patches)

   Fix severe security hazards and should always be installed.

   Recommended Updates (Patches)

   Fix issues that could compromise your computer. Installing them is strongly recommended.

   Optional Updates (Patches)

   Fix non-security relevant issues or provide enhancements.

   Other Updates

   New versions of packages that are installed.

   All available updates are preselected for installation. If you do not want to install all updates, deselect unwanted updates first. It is strongly recommended to always install all security and recommended updates.

   To get detailed information on an update, click its title and then *Details*. The information will be displayed in a box beneath the package list.

3. Click *Install Updates* to start the installation.

4. Some updates may require to restart the machine or to log out. Check the message that is displayed after the installation for instructions.

10.5 Updating Packages with GNOME Software

In addition to the GNOME *Package Updater*, GNOME provides *GNOME Software* which has the following functionality:

- Install, update, and remove software delivered as an RPM via PackageKit
- Install, update, and remove software delivered as a Flatpak
- Install, update, and remove GNOME shell extensions (https://extensions.gnome.org)
- Update firmware for hardware devices using *Linux Vendor Firmware Service* (LVFS, https://fwupd.org)

In addition to this, *GNOME Software* provides screenshots, ratings and reviews for software.

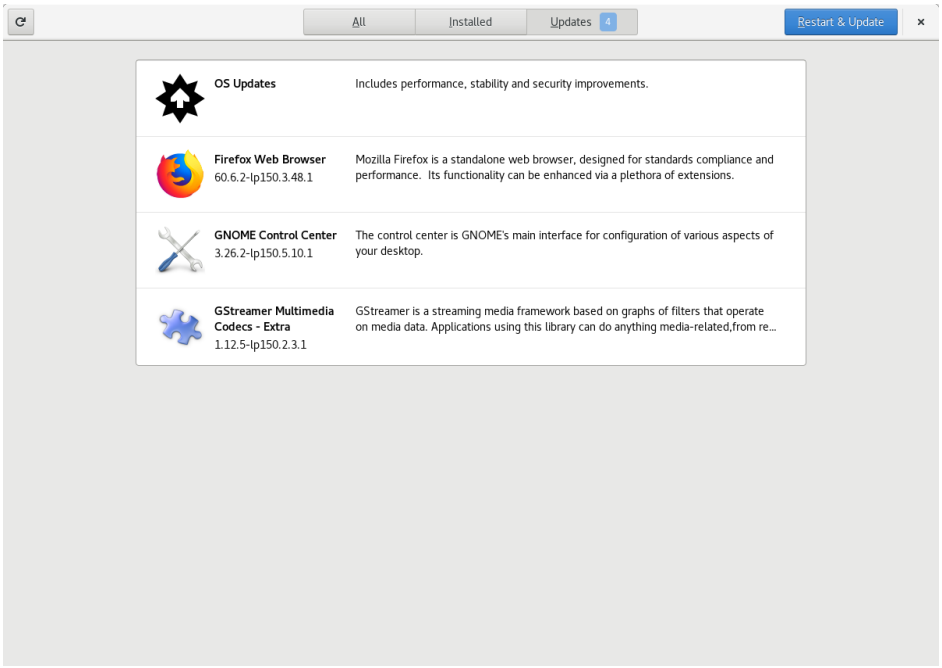


FIGURE 10.4: GNOME SOFTWARE—*UPDATES* VIEW

GNOME Software has the following differences to other tools provided on openSUSE Leap:

- Unlike YaST or Zypper, for installing software packaged as an RPM, *GNOME Software* is restricted to software that provides AppStream metadata. This includes most desktop applications.
- While the GNOME *Package Updater* updates packages within the running system (forcing you to restart the respective applications), *GNOME Software* downloads the updates but only applies them at the next reboot of the system.

11 Installing Add-On Products

Add-on products are system extensions. You can install a third party add-on product or a special system extension of openSUSE® Leap (for example, a CD with support for additional languages or a CD with binary drivers). To install a new add-on, start YaST and select *Software* › *Add-On Products*. You can select various types of product media, like CD, FTP, USB mass storage devices (such as USB flash drives or disks) or a local directory. You can also work directly with ISO files. To add an add-on as ISO file media, select *Local ISO Image* then enter the *Path to ISO Image*. The *Repository Name* is arbitrary.

11.1 Add-Ons

To install a new add-on, proceed as follows:

1. In YaST select *Software* › *Add-On Products* to see an overview of already installed add-on products.
2. To install a new add-on product, click *Add*.
3. From the list of available *Media Types* specify the type matching your repository.
4. To add a repository from a removable medium, choose the relevant option and insert the medium or connect the USB device to the machine, respectively.
5. You can choose to *Download Repository Description Files* now. If the option is deselected, YaST will automatically download the files later, if needed. Click *Next* to proceed.
6. When adding a repository from the network, enter the data you are prompted for. Continue with *Next*.
7. Depending on the repository you have added, you may be asked if you want to import the GPG key with which it is signed or asked to agree to a license.
   After confirming these messages, YaST will download and parse the metadata and add the repository to the list of *Configured Repositories.*.
8. If needed, adjust the repository *Properties* as described in *Section 10.3.2, "Managing Repository Properties"* or confirm your changes with *OK* to close the configuration dialog.

9. After having successfully added the repository for the add-on media, the software manager starts and you can install packages. Refer to *Chapter 10, Installing or Removing Software* for details.

11.2 Binary Drivers

Some hardware needs binary-only drivers to function properly. If you have such hardware, refer to the release notes for more information about availability of binary drivers for your system. To read the release notes, open YaST and select *Miscellaneous* › *Release Notes*.

12 YaST Online Update

SUSE offers a continuous stream of software security updates for your product. By default, the update applet is used to keep your system up-to-date. Refer to *Section 10.4, "The GNOME Package Updater"* for further information on the update applet. This chapter covers the alternative tool for updating software packages: YaST Online Update.

The current patches for openSUSE® Leap are available from an update software repository, which is automatically configured during the installation. Alternatively, you can manually add an update repository from a source you trust. To add or remove repositories, start the Repository Manager with *Software* › *Software Repositories* in YaST. Learn more about the Repository Manager in *Section 10.3, "Managing Software Repositories and Services"*.

SUSE provides updates with different relevance levels:

Security Updates

Fix severe security hazards and should always be installed.

Recommended Updates

Fix issues that could compromise your computer.

Optional Updates

Fix non-security relevant issues or provide enhancements.

12.1 The Online Update Dialog

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

The *Online Update* window consists of four sections.

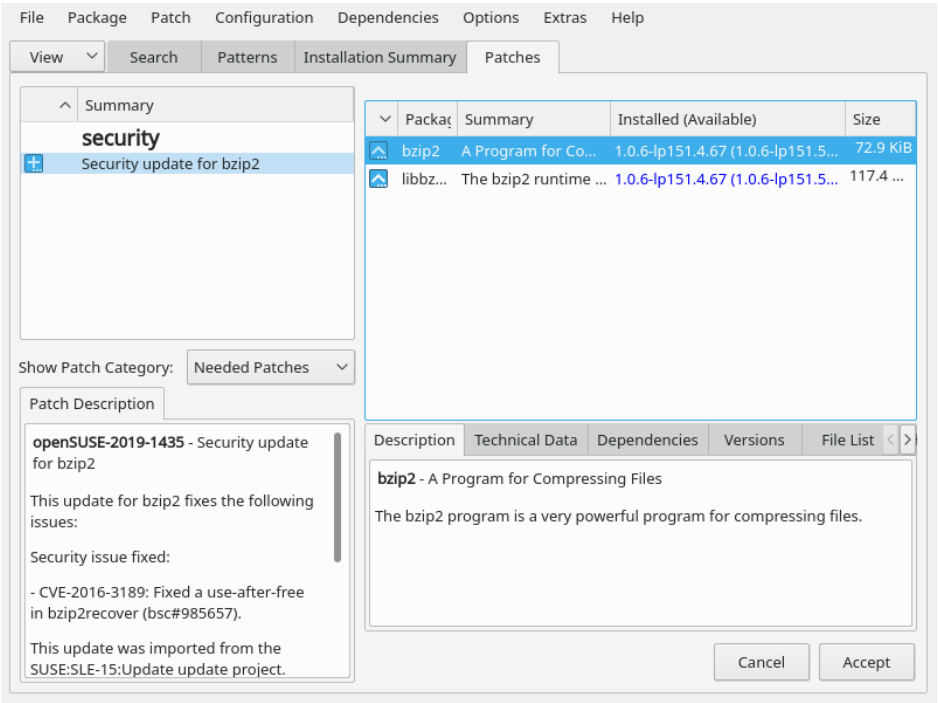


FIGURE 12.1: YAST ONLINE UPDATE

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

***Needed Patches* (default view)**

Non-installed patches that apply to packages installed on your system.

Unneeded Patches

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

All Patches

All patches available for openSUSE Leap.

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

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

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

12.2 Installing Patches

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

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

PROCEDURE 12.1: APPLYING PATCHES WITH YAST ONLINE UPDATE

1. Start YaST and select *Software* › *Online Update*.
2. To automatically apply all new patches (except `optional` ones) that are currently available for your system, click *Apply* or *Accept*.
3. First modify the selection of patches that you want to apply:
    a. Use the respective filters and views that the interface provides. For details, refer to *Section 12.1, “The Online Update Dialog”*.
    b. Select or deselect patches according to your needs and wishes by right-clicking the patch and choosing the respective action from the context menu.

Important: Always Apply Security Updates

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

c. Most patches include updates for several packages. To change actions for single packages, right-click a package in the package view and choose an action.

d. To confirm your selection and apply the selected patches, proceed with *Apply* or *Accept*.

4. After the installation is complete, click *Finish* to leave the YaST *Online Update*. Your system is now up-to-date.

12.3 Automatic Online Update

YaST also offers the possibility to set up an automatic update with daily, weekly or monthly schedule. To use the respective module, you need to install the `yast2-online-update-configuration` package first.

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

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

PROCEDURE 12.2: CONFIGURING THE AUTOMATIC ONLINE UPDATE

1. After installation, start YaST and select *Software* › *Online Update Configuration*. Alternatively, start the module with **`yast2 online_update_configuration`** from the command line.

2. Activate *Automatic Online Update*.

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

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

5. Sometimes patches may require the attention of the administrator, for example when restarting critical services. For example, this might be an update for Docker Open Source Engine that requires all containers to be restarted. Before these patches are installed, the user is informed about the consequences and is asked to confirm the installation of the patch. Such patches are called “Interactive Patches”.
   When installing patches automatically, it is assumed that you have accepted the installation of interactive patches. If you rather prefer to review these patches before they get installed, select *Skip Interactive Patches*. In this case, interactive patches will be skipped during automated patching. Make sure to periodically run a manual online update, to check whether interactive patches are waiting to be installed.
6. To automatically install all packages recommended by updated packages, activate *Include Recommended Packages*.
7. To disable the use of delta RPMs (for performance reasons), deactivate *Use Delta RPMs*.
8. To filter the patches by category (such as security or recommended), activate *Filter by Category* and add the appropriate patch categories from the list. Only patches of the selected categories will be installed. Others will be skipped.
9. Confirm your configuration with *OK*.

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

13 Upgrading the System and System Changes

You can upgrade an existing system without completely reinstalling it. There are two types of renewing the system or parts of it: *updating individual software packages* and *upgrading the entire system*. Updating individual packages is covered in *Chapter 10, Installing or Removing Software* and *Chapter 12, YaST Online Update*. Two ways to upgrade the system are discussed in the following sections— see *Section 13.1.3, "Upgrading with YaST"* and *Section 13.1.4, "Distribution Upgrade with Zypper"*.

13.1 Upgrading the System

Important: openSUSE Leap 15.1 is only available as 64-bit version

openSUSE Leap 15.1 is only available as 64-bit version. Upgrading 32-bit installations to 64-bit is not supported. Please follow the instructions in *Chapter 1, Installation Quick Start* and *Chapter 3, Installation Steps* to install openSUSE Leap on your computer or consider switching to openSUSE Tumbleweed (https://en.opensuse.org/Portal:Tumbleweed).

The release notes are bundled in the installer, and you may also read them online at openSUSE Leap Release Notes (https://doc.opensuse.org/release-notes/).

Software tends to "grow" from version to version. Therefore, take a look at the available partition space with `df` before updating. If you suspect you are running short of disk space, secure your data before you update and repartition your system. There is no general rule regarding how much space each partition should have. Space requirements depend on your particular partitioning profile, the software selected, and the version numbers of the system.

13.1.1 Preparations

Before upgrading, copy the old configuration files to a separate medium (such as removable hard disk or USB flash drive) to secure the data. This primarily applies to files stored in `/etc` as well as some of the directories and files in `/var`. You may also want to write the user data in `/home` (the `HOME` directories) to a backup medium. Back up this data as `root`. Only `root` has read permission for all local files.

Before starting your update, make note of the root partition. The command **`df /`** lists the device name of the root partition. In *Example 13.1, "List with* **`df -h`***"*, the root partition to write down is `/dev/sda3` (mounted as `/`).

EXAMPLE 13.1: LIST WITH **`df -h`**

```
Filesystem     Size  Used Avail Use% Mounted on
/dev/sda3       74G   22G   53G  29% /
udev           252M  124K  252M   1% /dev
/dev/sda5      116G  5.8G  111G   5% /home
/dev/sda1       39G  1.6G   37G   4% /windows/C
/dev/sda2      4.6G  2.6G  2.1G  57% /windows/D
```

13.1.2 Possible Problems

If you upgrade a default system from the previous version to this version, YaST works out the necessary changes and performs them. Depending on your customization, some steps (or the entire upgrade procedure) may fail and you must resort to copying back your backup data. Check the following issues before starting the system update.

13.1.2.1 Checking passwd and group in /etc

Before upgrading the system, make sure that `/etc/passwd` and `/etc/group` do not contain any syntax errors. For this purpose, start the verification utilities **`pwck`** and **`grpck`** as `root` to eliminate any reported errors.

13.1.2.2 Shut Down Virtual Machines

If your machine serves as a VM Host Server for KVM or Xen, make sure to properly shut down all running VM Guests prior to the update. Otherwise you may not be able to access the guests after the update.

13.1.2.3 PostgreSQL

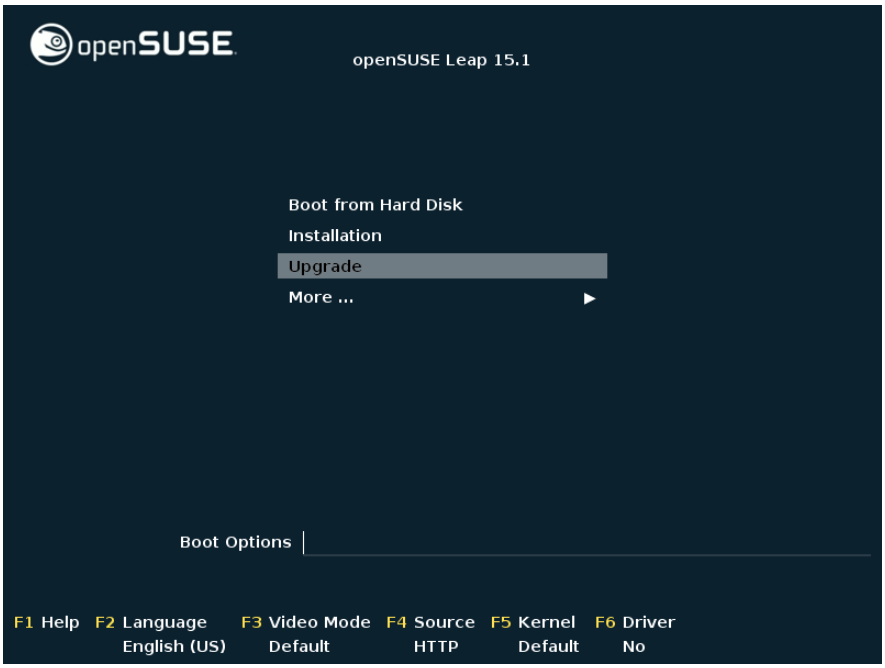
Before updating PostgreSQL (`postgres`), dump the databases. See the manual page of **`pg_dump`**. This is only necessary if you actually used PostgreSQL prior to your update.

13.1.3 Upgrading with YaST

Following the preparation procedure outlined in *Section 13.1.1, "Preparations"*, you can now upgrade your system:

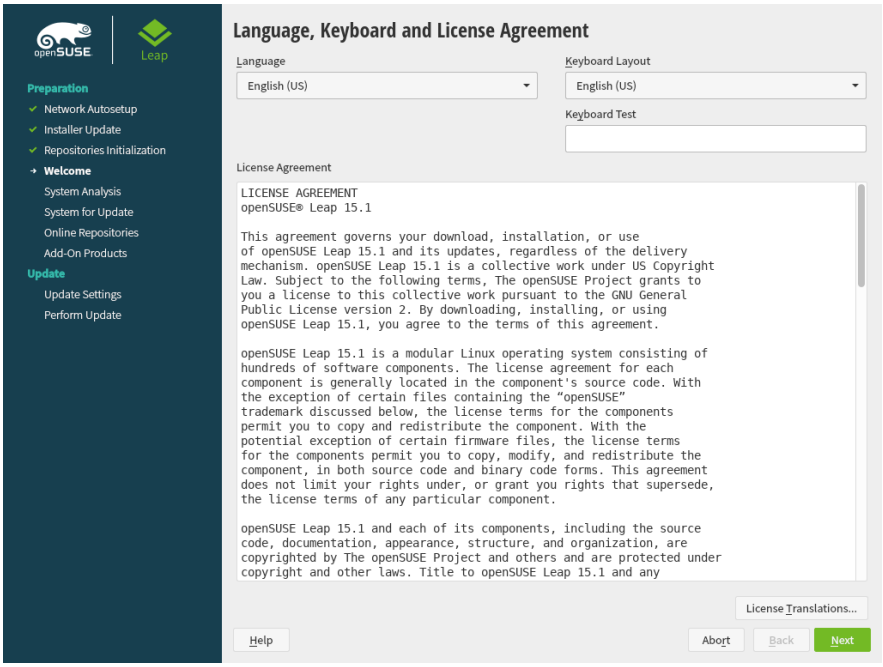
1. Insert the openSUSE Leap DVD into the drive, then reboot the computer to start the installation program. On machines with a traditional BIOS you will see the graphical boot screen shown below. On machines equipped with UEFI, a slightly different boot screen is used. Secure boot on UEFI machines is supported.

   Use F2 to change the language for the installer. A corresponding keyboard layout is chosen automatically. See *Section 2.2.1, "The Boot Screen on Machines Equipped with Traditional BIOS"* or *Section 2.2.2, "The Boot Screen on Machines Equipped with UEFI"* for more information about changing boot parameters.

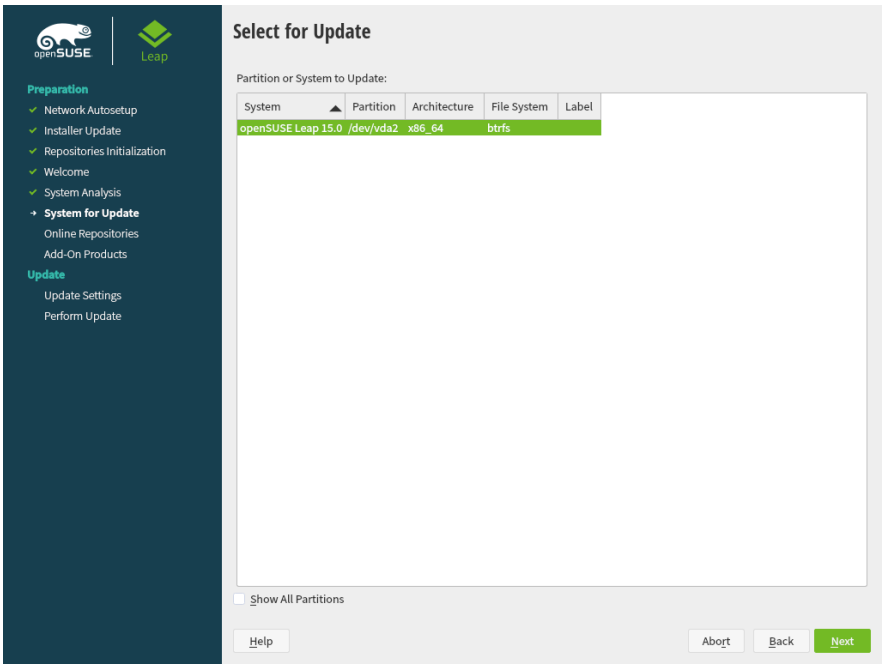


2. Select *Upgrade* on the boot screen, then press Enter. This boots the system and loads the openSUSE Leap installer. Do not select *Installation*.

3. The *Language* and *Keyboard Layout* are initialized with the language settings you have chosen on the boot screen. Change them here, if necessary.
   Read the License Agreement. It is presented in the language you have chosen on the boot screen. *License Translations* are available. Proceed with *Next*.



4. YaST determines if there are multiple root partitions. If there is only one, continue with the next step. If there are several, select the right partition and confirm with *Next* (`/dev/sda3` was selected in the example in *Section 13.1.1, "Preparations"*). YaST reads the old `fstab` on this partition to analyze and mount the file systems listed there.



Tip: Release Notes

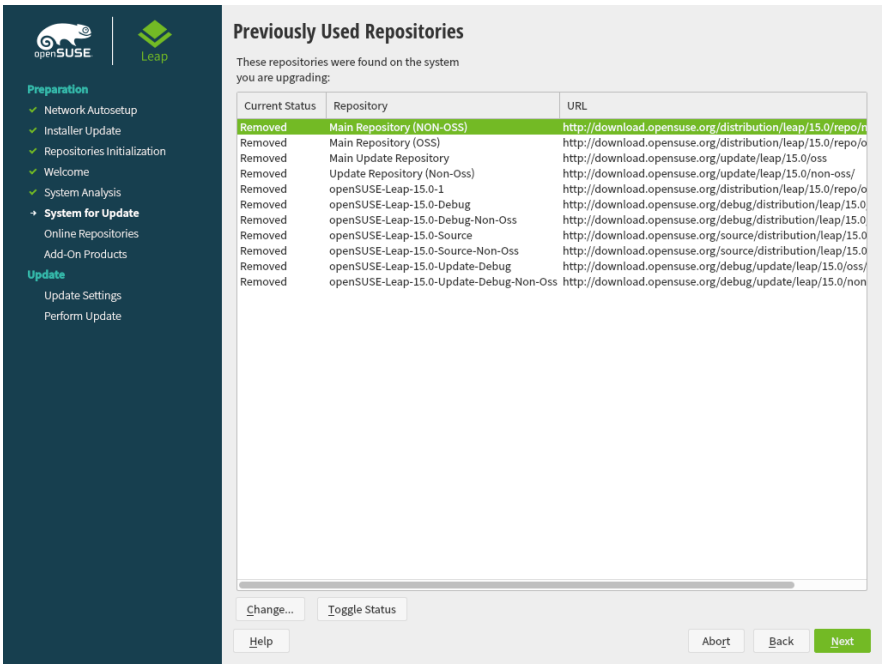
From this point on, the Release Notes can be viewed from any screen during the installation process by selecting *Release Notes*.

5. YaST shows a list of *Previously Used Repositories*. By default all repositories will get removed. If you had not added any custom repositories, do not change the settings. The packages for the upgrade will be installed from DVD and you can optionally enable the default online repositories can be chosen in the next step.
   If you have had added custom repositories, for example from the openSUSE Build Service, you have two choices:

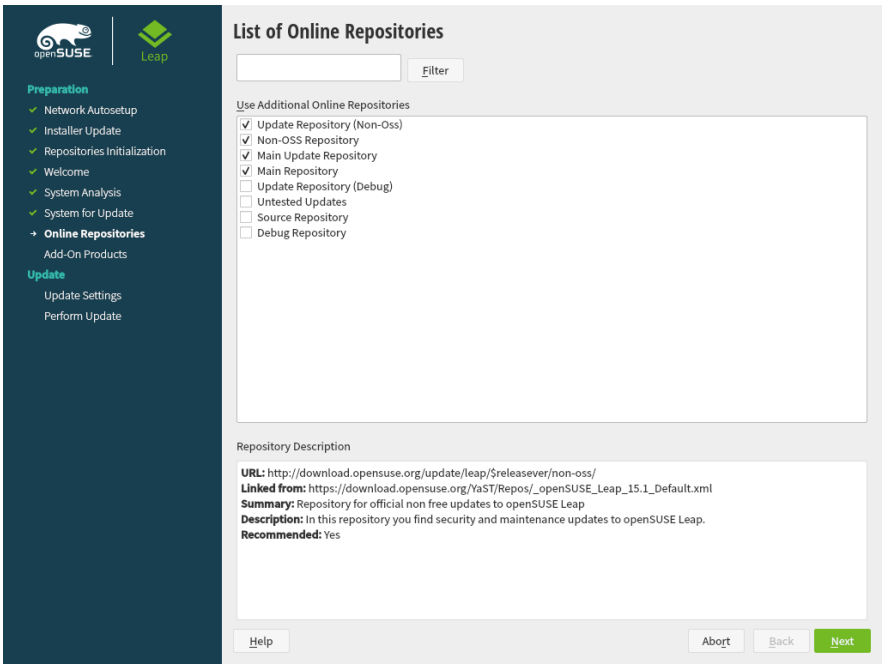
   - Leave the repository in state `Removed`. Software that was installed from this repository will get removed during the upgrade. Use this method if no version of the repository that matches the new openSUSE Leap version, is available.

   - Update and enable the repository. Use this method if a version that matches the new openSUSE Leap version is available for the repository. Change it's URL by clicking the repository in the list and then *Change*. Enable the repository afterwards by clicking *Toggle Status* until it is set to *Enable*.

Do not use repositories matching the previous version unless you are absolutely sure they will also work with the new openSUSE version. If not, the system may be unstable or not work at all.



6. In case an Internet connection is available, you may now activate optional online repositories. Please enable all repositories you had enable before to ensure all packages get upgraded correctly. Enabling the update repositories is strongly recommended—this will ensure that you get the latest package versions available, including ll security updates and fixes.



After having proceeded with *Next*, you need to confirm the license agreement for the online repositories with *Next*.

7. Use the *Installation Settings* screen to review and—if necessary—change several proposed installation settings. The current configuration is listed for each setting. To change it, click the headline.

   System

   View detailed hardware information by clicking *System*. In the resulting screen you can also change *Kernel Settings*—see *Section 3.11.6, "System"* for more information.

   Update Options

   By default, YaST will update perform full *Update with Installation of New Software and Features* based on a selection of patterns. Each pattern contains several software packages needed for specific functions (for example, Web and LAMP server or a print server).

   Here you can change the package selection or change the *Update Mode* to *Only Update Installed Packages*.

   Packages

You can further tweak the package selection on the *Packages* screen. Here you can not only select patterns but also list their contents and search for individual packages. See *Chapter 10, Installing or Removing Software* for more information.

If you intend to enhance your system, it is recommended to finish the upgrade first and then install additional software.

Backup

You also have the possibility to make backups of various system components. Selecting backups slows down the upgrade process. Use this option if you do not have a recent system backup.

Language

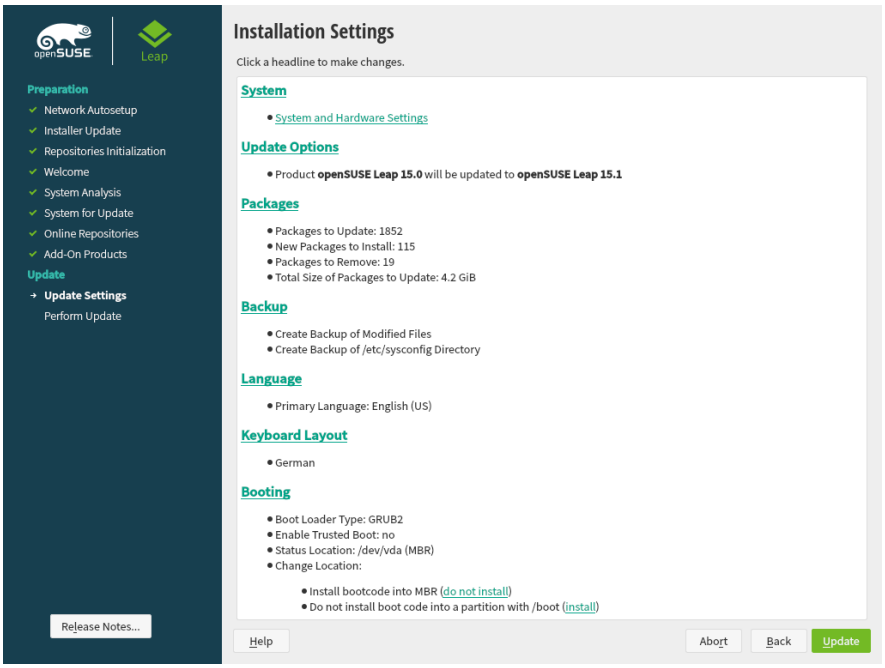
This section allows you to change the *Primary Language*primary language and configure additional *Secontry Languages.*. Optionally, you can adjust the keyboard layout and timezone to the selected primary language.

Keyboard Layout

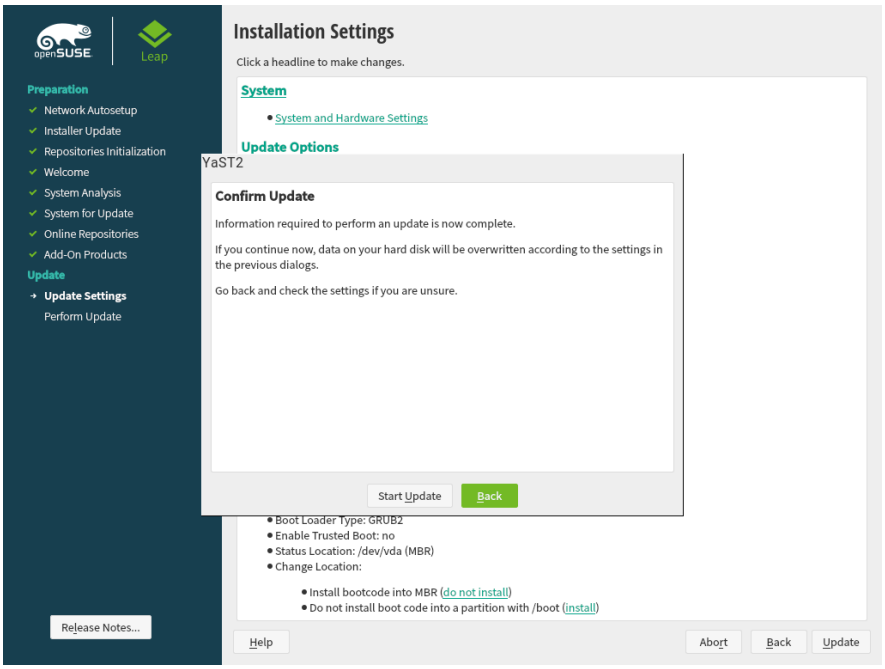
Here you can change the keyboard layout and adjust additional *Expert Keyboard Settings*.

Booting

This section shows the boot loader configuration. Changing the defaults is only recommended if really needed. Refer to *Book "*Reference*", Chapter 12 "The Boot Loader GRUB 2"* for details.



8. After you have finalized the system configuration on the *Installation Settings* screen, click *Update*. Depending on your software selection you may need to agree to license agreements before the installation confirmation screen pops up. Up to this point no changes have been made to your system. After you click *Update* a second time, the upgrade process starts.



Once the basic upgrade installation is finished, YaST reboots the system. Finally, YaST updates the remaining software, if any and displays the release notes, if wanted.

13.1.4 Distribution Upgrade with Zypper

With the `zypper` command line utility you can upgrade to the next version of the distribution. Most importantly, you can initiate the system upgrade process from within the running system.

This feature is attractive for advanced users who want to run remote upgrades or upgrades on many similarly configured systems.

13.1.4.1 Preparing the Upgrade with Zypper

To avoid unexpected errors during the upgrade process using `zypper`, minimize risky constellations.

- Quit as many applications and stop unneeded services as possible and log out all regular users.
- Disable third party repositories before starting the upgrade, or lower the priority of these repositories to make sure packages from the default system repositories will get preference. Enable them again after the upgrade and edit their version string to match the version number of the distribution of the upgraded now running system.

13.1.4.2 The Upgrade Procedure

Warning: Check Your System Backup

Before actually starting the upgrade procedure, check that your system backup is up-to-date and restorable. This is especially important because you need to enter many of the following steps manually.

The program `zypper` supports long and short command names. For example, you can abbreviate `zypper install` as `zypper in`. In the following text, the short variants are used.

1. Run the online update to make sure the software management stack is up-to-date. For more information, see *Chapter 12, YaST Online Update*.

2. Configure the repositories you want to use as update sources. Getting this right is crucial. Either use YaST (see *Section 10.3, "Managing Software Repositories and Services"*) or **zypper** (see *Book "Reference", Chapter 2 "Managing Software with Command Line Tools", Section 2.1 "Using Zypper"*). The name of the repositories used in the following steps may vary depending on your customizations.
   To view your current repositories enter:

```
tux > zypper lr -u
```

   a. Increase the version number of the system repositories from 42.3 to 15.0 `leap/`. Add the new repositories with commands such as:

```
server=http://download.example.org
tux > sudo zypper ar $server/distribution/leap/15.0/repo/oss/ Leap-15-OSS
tux > sudo zypper ar $server/update/leap/15.0/oss/ Leap-15-Update
```

      And remove the old repositories:

```
zypper rr Leap-42.3-OSS
zypper rr Leap-42.3-Update
```

      If necessary, repeat these steps for other repositories to ensure a clean upgrade path for all your packages.

   b. Disable third party repositories or other Open Build Service repositories, because **zypper dup** is guaranteed to work with the default repositories only (replace *REPO-ALIAS* with the name of the repository you want to disable):

```
tux > sudo zypper mr -d REPO-ALIAS
```

      Alternatively, you can lower the priority of these repositories.

Note: Handling of Unresolved Dependencies

zypper dup will remove all packages having unresolved dependencies, but it keeps packages of disabled repositories as long as their dependencies are satisfied.

`zypper dup` ensures that all installed packages come from one of the available repositories. It does not consider the version or architecture, but prevents changing the vendor of the installed packages by default, using the `--no-allow-vendor-change` option. Packages that are no longer available in the repositories are considered orphaned. Such packages get uninstalled if their dependencies cannot be satisfied. If they can be satisfied, such packages stay installed.

c. Once done, check your repository configuration with:

```
tux > zypper lr -d
```

3. Refresh local metadata and repository contents with `zypper ref`.

4. Update Zypper and the package management itself with `zypper patch --updatestack-only`.

5. Run the actual distribution upgrade with `zypper dup`. You are asked to confirm the license of openSUSE Leap and of some packages—depending on the set of installed packages.

6. Reboot the system with `shutdown -r now`.

13.1.5 Updating Individual Packages

Regardless of your overall updated environment, you can always update individual packages. From this point on, however, it is your responsibility to ensure that your system remains consistent.

Use the YaST software management tool to update packages as described in *Chapter 10, Installing or Removing Software*. Select components from the YaST package selection list according to your needs. If a newer version of a package exists, the version numbers of the installed and the available versions are listed in blue color in the *Installed (Available)* column. If you select a package essential for the overall operation of the system, YaST issues a warning. Such packages should be updated only in the update mode. For example, many packages contain *shared libraries*. Updating these programs and applications in the running system may lead to system instability.

13.2 Additional Information

Problems and special issues of the various versions are published online as they are identified. See the links listed below. Important updates of individual packages can be accessed using the YaST Online Update. For more information, see *Chapter 12, YaST Online Update*.

Refer to the `Product highlights` (http://en.opensuse.org/Product_highlights) and the `Bugs` article in the openSUSE wiki at http://en.opensuse.org/openSUSE:Most_annoying_bugs for information about recent changes and issues.

IV The Bash Shell

14 Shell Basics

When working with Linux, you can communicate with the system almost without ever requiring a command line interpreter (the shell). After booting your Linux system, you are usually directed to a graphical user interface that guides you through the login process and the following interactions with the operating system. The graphical user interface in Linux is initially configured during installation and used by desktop environments such as KDE or GNOME.

Nevertheless, it is useful to have some basic knowledge of working with a shell because you might encounter situations where the graphical user interface is not available. For example, if some problem with the X Window System occurs. If you are not familiar with a shell, you might feel a bit uncomfortable at first when entering commands, but the more you get used to it, the more you will realize that the command line is often the quickest and easiest way to perform some daily tasks.

For Unix or Linux, several shells are available which differ slightly in behavior and in the commands they accept. The default shell in openSUSE® Leap is Bash (GNU Bourne-Again Shell).

The following sections will guide you through your first steps with the Bash shell and will show you how to complete some basic tasks via the command line. If you are interested in learning more or rather feel like a shell “power user” already, refer to *Chapter 15, Bash and Bash Scripts*.

14.1 Starting a Shell

Basically, there are two different ways to start a shell from the graphical user interface which usually shows after you have booted your computer:

- you can leave the graphical user interface or
- you can start a terminal window *within* the graphical user interface.

While the first option is always available, you can only make use of the second option when you are already logged in to a desktop such as KDE or GNOME. Whichever way you choose, there is always a way back and you can switch back and forth between the shell and the graphical user interface.

If you want to give it a try, press Ctrl–Alt–F2 to leave the graphical user interface. The graphical user interface disappears and you are taken to a shell which prompts you to log in. Type your username and press Enter. Then type your password and press Enter. The prompt now changes and shows some useful information as in the following example:

```
  ❶    ❷    ❸
tux@linux:~>
```

❶ Your login.

❷ The hostname of your computer.

❸ Path to the current directory. Directly after login, the current directory usually is your home directory, indicated by the ~ symbol (tilde) .

When you are logged in at a remote computer the information provided by the prompt always shows you which system you are currently working on.

When the cursor is located behind this prompt, you can pass commands directly to your computer system. For example, you can now enter **ls** -l to list the contents of the current directory in a detailed format. If this is enough for your first encounter with the shell and you want to go back to the graphical user interface, you should log out from your shell session first. To do so, type **exit** and press Enter. Then press Alt–F7 to switch back to the graphical user interface. You will find your desktop and the applications running on it unchanged.

When you are already logged in to the GNOME or the KDE desktop and want to start a terminal window within the desktop, press Alt–F2 and enter **konsole** (for KDE) or **gnome-terminal** (for GNOME). This opens a terminal window on your desktop. As you are already logged in to your desktop, the prompt shows information about your system as described above. You can now enter commands and execute tasks just like in any shell which runs parallel to your desktop. To switch to another application on the desktop just click on the corresponding application window or select it from the taskbar of your panel. To close the terminal window press Alt–F4.

14.2 Entering Commands

As soon as the prompt appears on the shell it is ready to receive and execute commands. A command can consist of several elements. The first element is the actual command, followed by parameters or options. You can type a command and edit it by using the following keys: ←, →, Home, End, <– (Backspace), Del, and Space. You can correct typing errors or add options. The command is not executed until you press Enter.

Important: No News Is Good News

The shell is not verbose: in contrast to some graphical user interfaces, it usually does not provide confirmation messages when commands have been executed. Messages only appear in case of problems or errors —or if you explicitly ask for them by executing a command with a certain option.

Also keep this in mind for commands to delete objects. Before entering a command like `rm` (without any option) for removing a file, you should know if you really want to get rid of the object: it will be deleted irretrievably, without confirmation.

14.2.1 Using Commands without Options

In *Section 14.6.1, "Permissions for User, Group and Others"* you already got to know one of the most basic commands: `ls`, which used to list the contents of a directory. This command can be used with or without options. Entering the plain `ls` command shows the contents of the current directory:

```
tux > ls
bin Desktop Documents public_html tux.txt
tux >
```

Files in Linux may have a file extension or a suffix, such as `.txt`, but do not need to have one. This makes it difficult to differentiate between files and folders in this output of the `ls`. By default, the colors in the Bash shell give you a hint: directories are usually shown in blue, files in black.

14.2.2 Using Commands with Options

A better way to get more details about the contents of a directory is using the `ls` command with a string of options. Options modify the way a command works so that you can get it to carry out specific tasks. Options are separated from the command with a blank and are usually prefixed with a hyphen. The `ls -l` command shows the contents of the same directory in full detail (long listing format):

```
tux > ls -l
drwxr-xr-x 1 tux users     48 2015-06-23 16:08 bin
drwx---r-- 1 tux users  53279 2015-06-21 13:16 Desktop
```

```
drwx------ 1 tux users    280 2015-06-23 16:08 Documents
drwxr-xr-x 1 tux users  70733 2015-06-21 09:35 public_html
-rw-r--r-- 1 tux users  47896 2015-06-21 09:46 tux.txt
tux >
```

This output shows the following information about each object:

```
drwxr-xr-x❶ 1❷ tux❸ users❹ 48❺ 2006-06-23 16:08❻ bin❼
```

❶ Type of object and access permissions. For further information, refer to *Section 14.6.1, "Permissions for User, Group and Others"*.

❷ Number of hard links to this file.

❸ Owner of the file or directory. For further information, refer to *Section 14.6.1, "Permissions for User, Group and Others"*.

❹ Group assigned to the file or directory. For further information, refer to *Section 14.6.1, "Permissions for User, Group and Others"*.

❺ File size in bytes.

❻ Date and time of the last change.

❼ Name of the object.

Usually, you can combine several options by prefixing only the first option with a hyphen and then write the others consecutively without a blank. For example, if you want to see all files in a directory in long listing format, you can combine the two options `-l` and `-a` (show all files) for the **`ls`** command. Executing **`ls`** `-la` shows also hidden files in the directory, indicated by a dot in front (for example, `.hiddenfile`).

The list of contents you get with **`ls`** is sorted alphabetically by filenames. But like in a graphical file manager, you can also sort the output of **`ls`** `-l` according to various criteria such as date, file extension or file size:

- For date and time, use **`ls`** `-lt` (displays newest first).
- For extensions, use **`ls`** `-lx` (displays files with no extension first).
- For file size, use **`ls`** `-lS` (displays largest first).

To revert the order of sorting, add `-r` as an option to your **`ls`** command. For example, **`ls`** `-lr` gives you the contents list sorted in reverse alphabetical order, **`ls`** `-ltr` shows the oldest files first. There are lots of other useful options for **`ls`**. In the following section you will learn how to investigate them.

14.2.3 Bash Shortcut Keys

After having entered several commands, your shell will begin to fill up with all sorts of commands and the corresponding outputs. In the following table, find some useful shortcut keys for navigating and editing in the shell.

Shortcut Key	Function
Ctrl – L	Clears the screen and moves the current line to the top of the page.
Ctrl – C	Aborts the command which is currently being executed.
Shift – Page ↑	Scrolls upwards.
Shift – Page ↓	Scrolls downwards.
Ctrl – U	Deletes from cursor position to start of line.
Ctrl – K	Deletes from cursor position to the end of line.
Ctrl – D	Closes the shell session.
↑, ↓	Browses in the history of executed commands.

14.3 Getting Help

If you remember the name of command but are not sure about the options or the syntax of the command, choose one of the following possibilities:

`--help` / `-h` option

If you only want to look up the options of a certain command, try entering the command followed by a space and `--help`. This `--help` option exists for many commands. For example, `ls --help` displays all the options for the `ls` command.

Manual Pages

To learn more about the various commands, you can also use the manual pages. Manual pages also give a short description of what the command does. They can be accessed with **man** followed by the name of the command, for example, **man ls**.
Man pages are displayed directly in the shell. To navigate them, use the following keys:

- Move up and down with Page ↑ and Page ↓
- Move between the beginning and the end of a document with Home and End
- Quit the man page viewer by pressing Q

For more information about the **man** command, use **man man**.

Info Pages

Info pages usually provide even more information about commands. To view the info page for a certain command, enter **info** followed by the name of the command (for example, **info ls**).
Info pages are displayed directly in the shell. To navigate them, use the following keys:

- Use Space to move forward a section (*node*). Use <– to move backward a section.
- Move up and down with Page ↑ and Page ↓
- Quit the info page viewer by pressing Q

Note that man pages and info pages do not exist for all commands. Sometimes both are available (usually for key commands), sometimes only a man page or an info page exists, and sometimes neither of them are available.

14.4 Working with Files and Directories

To address a certain file or directory, you must specify the path leading to that directory or file. There are two ways to specify a path:

Absolute Path

The entire path from the root directory (`/`) to the relevant file or directory. For example, the absolute path to a text file named `file.txt` in your `Documents` directory might be:

```
/home/tux/Documents/file.txt
```

Relative Path

The path from the current working directory to the relevant file or directory. If your current working directory is `/home/tux`, the relative path `file.txt` in your `Documents` directory is:

```
Documents/file.txt
```

However, if your working directory is `/home/tux/Music` instead, you need to move up a level to `/home/tux` (with `..`) before you can go further down:

```
../Documents/file.txt
```

Paths contain file names, directories or both, separated by slashes. Absolute paths always start with a slash. Relative paths do not have a slash at the beginning, but can have one or two dots.

When entering commands, you can choose either way to specify a path, depending on your preferences or the amount of typing, both will lead to the same result. To change directories, use the **`cd`** command and specify the path to the directory.

Note: Handling Blanks in Filenames or Directory Names

If a filename or the name of a directory contains a space, either escape the space using a back slash (`\`) in front of the blank or enclose the filename in single quotes. Otherwise Bash interprets a filename like `My Documents` as the names of two files or directories, `My` and `Documents` in this case.

When specifying paths, the following “shortcuts” can save you a lot of typing:

- The tilde symbol (`~`) is a shortcut for home directories. For example, to list the contents of your home directory, use **`ls`** `~`. To list the contents of another user's home directory, enter **`ls`** *`~USERNAME`* (or course, this will only work if you have permission to view the contents, see *Section 14.6, “File Access Permissions”*). For example, entering **`ls ~tux`** would list the contents of the home directory of a user named `tux`. You can use the tilde symbol as shortcut for home directories also if you are working in a network environment where your home directory may not be called `/home` but can be mapped to any directory in the file system.

From anywhere in the file system, you can reach your home directory by entering `cd ~` or by simply entering `cd` without any options.

- When using relative paths, refer to the current directory with a dot (`.`). This is mainly useful for commands such as `cp` or `mv` by which you can copy or move files and directories.
- The next higher level in the tree is represented by two dots (`..`). In order to switch to the parent directory of your current directory, enter `cd ..`, to go up two levels from the current directory enter `cd ../..` etc.

To apply your knowledge, find some examples below. They address basic tasks you may want to execute with files or folders using Bash.

14.4.1 Examples for Working with Files and Directories

Suppose you want to copy a file located somewhere in your home directory to a subdirectory of `/tmp` that you need to create first.

PROCEDURE 14.1: CREATING AND CHANGING DIRECTORIES

From your home directory create a subdirectory in `/tmp`:

1. Enter

   ```
   tux > mkdir /tmp/test
   ```

   `mkdir` stands for "make directory". This command creates a new directory named `test` in the `/tmp` directory. In this case, you are using an absolute path to create the `test` directory.

2. To check what happened, now enter

   ```
   tux > ls -l /tmp
   ```

   The new directory `test` should appear in the list of contents of the `/tmp` directory.

3. Switch to the newly created directory with

   ```
   tux > cd /tmp/test
   ```

PROCEDURE 14.2: CREATING AND COPYING FILES

Now create a new file in a subdirectory of your home directory and copy it to `/tmp/test`. Use a relative path for this task.

Important: Overwriting of Existing Files

Before copying, moving or renaming a file, check if your target directory already contains a file with the same name. If yes, consider changing one of the filenames or use **cp** or **mv** with options like `-i`, which will prompt before overwriting an existing file. Otherwise Bash will overwrite the existing file without confirmation.

1. To list the contents of your home directory, enter

   ```
   tux > ls -l ~
   ```

   It should contain a subdirectory called `Documents` by default. If not, create this subdirectory with the **mkdir** command you already know:

   ```
   tux > mkdir ~/Documents
   ```

2. To create a new, empty file named `myfile.txt` in the `Documents` directory, enter

   ```
   tux > touch ~/Documents/myfile.txt
   ```

   Usually, the **touch** command updates the modification and access date for an existing file. If you use **touch** with a filename which does not exist in your target directory, it creates a new file.

3. Enter

   ```
   tux > ls -l ~/Documents
   ```

   The new file should appear in the list of contents.

4. To copy the newly created file, enter

   ```
   tux > cp ~/Documents/myfile.txt .
   ```

   Do not forget the dot at the end.

   This command tells Bash to go to your home directory and to copy `myfile.txt` from the `Documents` subdirectory to the current directory, `/tmp/test`, without changing the name of the file.

5. Check the result by entering

   ```
   tux > ls -l
   ```

The file `myfile.txt` should appear in the list of contents for `/tmp/test`.

PROCEDURE 14.3: RENAMING AND REMOVING FILES OR DIRECTORIES

Now suppose you want to rename `myfile.txt` into `tuxfile.txt`. Finally you decide to remove the renamed file and the `test` subdirectory.

1. To rename the file, enter

```
tux > mv myfile.txt tuxfile.txt
```

2. To check what happened, enter

```
tux > ls -l
```

Instead of `myfile.txt`, `tuxfile.txt` should appear in the list of contents.
`mv` stands for `move` and is used with two options: the first option specifies the source, the second option specifies the target of the operation. You can use `mv` either

- to rename a file or a directory,
- to move a file or directory to a new location or
- to do both in one step.

3. Coming to the conclusion that you do not need the file any longer, you can delete it by entering

```
tux > rm tuxfile.txt
```

Bash deletes the file without any confirmation.

4. Move up one level with `cd ..` and check with

```
tux > ls -l test
```

if the `test` directory is empty now.

5. If yes, you can remove the `test` directory by entering

```
tux > rmdir test
```

14.5 Becoming Root

`root`, also called the superuser, has privileges which authorize them to access all parts of the system and to execute administrative tasks. They have the unrestricted capacity to make changes to the system and they have unlimited access to all files. Therefore, performing some administrative tasks or running certain programs such as YaST requires `root` permissions.

14.5.1 Using **su**

In order to temporarily become `root` in a shell, proceed as follows:

1. Enter **`su`**. You are prompted for the `root` password.
2. Enter the password. If you mistyped the `root` password, the shell displays a message. In this case, you have to re-enter **`su`** before retyping the password. If your password is correct, a hash symbol `#` appears at the end of the prompt, signaling that you are acting as `root` now.
3. Execute your task. For example, transfer ownership of a file to a new user which only `root` is allowed to do:

   ```
   tux > chown wilber kde_quick.xml
   ```
4. After having completed your tasks as `root`, switch back to your normal user account. To do so, enter

   ```
   tux > exit
   ```

   The hash symbol disappears and you are acting as “normal” user again.

14.5.2 Using **sudo**

Alternatively, you can also use **`sudo`** (superuser “do”) to execute some tasks which normally are for `root`s only. With sudo, administrators can grant certain users `root` privileges for some commands. Depending on the system configuration, users can then run `root` commands by entering their normal password only. Due to a timestamp function, users are only granted a “ticket” for a restricted period of time after having entered their password. The ticket usually expires after a few minutes. In openSUSE, sudo requires the `root` password by default (if not configured otherwise by your system administrator).

For users, sudo is convenient as it prevents you from switching accounts twice (to `root` and back again). To change the ownership of a file using sudo, only one command is necessary instead of three:

```
tux > sudo chown wilber kde_quick.xml
```

After you have entered the password which you are prompted for, the command is executed. If you enter a second `root` command shortly after that, you are not prompted for the password again, because your ticket is still valid. After a certain amount of time, the ticket automatically expires and the password is required again. This also prevents unauthorized persons from gaining `root` privileges in case a user forgets to switch back to their normal user account again and leaves a `root` shell open.

14.6 File Access Permissions

In Linux, objects such as files or folders or processes generally belong to the user who created or initiated them. There are some exceptions to this rule. For more information about the exceptions, refer to *Book "Security Guide", Chapter 10 "Access Control Lists in Linux"*. The group which is associated with a file or a folder depends on the primary group the user belongs to when creating the object.

When you create a new file or directory, initial access permissions for this object are set according to a predefined scheme. As an owner of a file or directory, you can change the access permissions for this object. For example, you can protect files holding sensitive data against read access by other users and you can authorize the members of your group or other users to write, read, or execute several of your files where appropriate. As `root`, you can also change the ownership of files or folders.

14.6.1 Permissions for User, Group and Others

Three permission sets are defined for each file object on a Linux system. These sets include the read, write, and execute permissions for each of three types of users—the owner, the group, and other users.

The following example shows the output of an **`ls`** `-l` command in a shell. This command lists the contents of a directory and shows the details for each file and folder in that directory.

EXAMPLE 14.1: ACCESS PERMISSIONS FOR FILES AND FOLDERS

```
-rw-r----- 1 tux users      0 2015-06-23 16:08 checklist.txt
-rw-r--r-- 1 tux users  53279 2015-06-21 13:16 gnome_quick.xml
-rw-rw---- 1 tux users      0 2015-06-23 16:08 index.htm
-rw-r--r-- 1 tux users  70733 2015-06-21 09:35 kde-start.xml
-rw-r--r-- 1 tux users  47896 2015-06-21 09:46 kde_quick.xml
drwxr-xr-x 2 tux users     48 2015-06-23 16:09 local
-rwxr--r-- 1 tux users 624398 2015-06-23 15:43 tux.sh
```

As shown in the third column, all objects belong to user `tux`. They are assigned to the group `users` which is the primary group the user `tux` belongs to. To retrieve the access permissions the first column of the list must be examined more closely. Let's have a look at the file `kde-start.xml`:

Type	User Permissions	Group Permissions	Permissions for Others
`-`	`rw-`	`r--`	`r--`

The first column of the list consists of one leading character followed by nine characters grouped in three blocks. The leading character indicates the file type of the object: in this case, the hyphen (`-`) shows that `kde-start.xml` is a file. If you find the character `d` instead, this shows that the object is a directory, like `local` in *Example 14.1, "Access Permissions For Files and Folders"*.

The next three blocks show the access permissions for the owner, the group and other users (from left to right). Each block follows the same pattern: the first position shows read permissions (`r`), the next position shows write permissions (`w`), the last one shows execute permission (`x`). A lack of either permission is indicated by `-`. In our example, the owner of `kde-start.xml` has read and write access to the file but cannot execute it. The `users` group can read the file but cannot write or execute it. The same holds true for the other users as shown in the third block of characters.

14.6.2 Files and Folders

Access permissions have a slightly different impact depending on the type of object they apply to: file or directory. The following table shows the details:

TABLE 14.1: ACCESS PERMISSIONS FOR FILES AND DIRECTORIES

Access Permission	File	Folder
Read (r)	Users can open and read the file.	Users can view the contents of the directory. Without this permission, users cannot list the contents of this directory with `ls -l`, for example. However, if they only have execute permission for the directory, they can nevertheless access certain files in this directory if they know of their existence.
Write (w)	Users can change the file: They can add or drop data and can even delete the contents of the file. However, this does not include the permission to remove the file completely from the directory as long as they do not have write permissions for the directory where the file is located.	Users can create, rename or delete files in the directory.
Execute (x)	Users can execute the file. This permission is only relevant for files like programs or shell scripts, not for text files. If the operating	Users can change into the directory and execute files there. If they do not have read access to that directory they cannot list

Access Permission	File	Folder
	system can execute the file directly, users do not need read permission to execute the file. However, if the file must me interpreted like a shell script or a perl program, additional read permission is needed.	the files but can access them nevertheless if they know of their existence.

Note that access to a certain file is always dependent on the correct combination of access permissions for the file itself *and* the directory it is located in.

14.6.3 Modifying File Permissions

In Linux, objects such as files or folder or processes generally belong to the user who created or initiated them. The group which is associated with a file or a folder depends on the primary group the user belongs to when creating the object. When you create a new file or directory, initial access permissions for this object are set according to a predefined scheme. For further details refer to *Section 14.6, "File Access Permissions"*.

As the owner of a file or directory (and, of course, as `root`), you can change the access permissions to this object.

To change object attributes like access permissions of a file or folder, use the **`chmod`** command followed by the following parameters:

- the users for which to change the permissions,
- the type of access permission you want to remove, set or add and
- the files or folders for which you want to change permissions separated by spaces.

The users for which you can change file access permissions fall into the following categories: the owner of the file (user, `u`), the group that own the file (group, `g`) and the other users (others, `o`). You can add, remove or set one or more of the following permissions: read, write or execute.

As `root`, you can also change the ownership of a file: with the command **`chown`** (change owner) you can transfer ownership to a new user.

14.6.3.1 Examples for Changing Access Permissions and Ownership

The following example shows the output of an `ls -l` command in a shell.

EXAMPLE 14.2: ACCESS PERMISSIONS FOR FILES AND FOLDERS

```
-rw-r----- 1 tux users      0 2015-06-23 16:08 checklist.txt
-rw-r--r-- 1 tux users  53279 2015-06-21 13:16 gnome_quick.xml
-rw-rw---- 1 tux users      0 2015-06-23 16:08 index.htm
-rw-r--r-- 1 tux users  70733 2015-06-21 09:35 kde-start.xml
-rw-r--r-- 1 tux users  47896 2015-06-21 09:46 kde_quick.xml
drwxr-xr-x 2 tux users     48 2015-06-23 16:09 local
-r-xr-xr-x 1 tux users 624398 2015-06-23 15:43 tux.jpg
```

In the example above, user `tux` owns the file `kde-start.xml` and has read and write access to the file but cannot execute it. The `users` group can read the file but cannot write or execute it. The same holds true for the other users as shown by the third block of characters.

PROCEDURE 14.4: CHANGING ACCESS PERMISSIONS

Suppose you are `tux` and want to modify the access permissions to your files:

1. If you want to grant the `users` group also write access to `kde-start.xml`, enter

   ```
   tux > chmod g+w kde-start.xml
   ```

2. To grant the `users` group and other users write access to `kde-start.xml`, enter

   ```
   tux > chmod go+w kde-start.xml
   ```

3. To remove write access for all users, enter

   ```
   tux > chmod -w kde-start.xml
   ```

   If you do not specify any kind of users, the changes apply to all users— the owner of the file, the owning group and the others. Now even the owner `tux` does not have write access to the file without first reestablishing write permissions.

4. To prohibit the `users` group and others to change into the directory `local`, enter

   ```
   tux > chmod go-x local
   ```

5. To grant others write permissions for two files, for `kde_quick.xml` and `gnome_quick.xml`, enter

   ```
   tux > chmod o+w  kde_quick.xml gnome_quick.xml
   ```

PROCEDURE 14.5: CHANGING OWNERSHIP

Suppose you are `tux` and want to transfer the ownership of the file `kde_quick.xml` to an other user, named `wilber`. In this case, proceed as follows:

1. Enter the username and password for `root`.
2. Enter

```
root # chown wilber kde_quick.xml
```

3. Check what happened with

```
tux > ls -l kde_quick.xml
```

You should get the following output:

```
-rw-r--r-- 1 wilber users  47896 2006-06-21 09:46 kde_quick.xml
```

4. If the ownership is set according to your wishes, switch back to your normal user account.

14.7 Time-Saving Features of Bash

Entering commands in Bash can involve a lot of typing. This section introduces some features that can save you both time and typing.

History

By default, Bash "remembers" commands you have entered. This feature is called *history*. You can browse through commands that have been entered before, select one you want to repeat and then execute it again. To do so, press ↑ repeatedly until the desired command appears at the prompt. To move forward through the list of previously entered commands, press ↓. For easier repetition of a certain command from Bash history, just type the first letter of the command you want to repeat and press Page ↑.

You can now edit the selected command (for example, change the name of a file or a path), before you execute the command by pressing Enter. To edit the command line, move the cursor to the desired position using the arrow keys and start typing.

You can also search for a certain command in the history. Press Ctrl–R to start an incremental search function. showing the following prompt:

```
tux > (reverse-i-search)`':
```

Just type one or several letters from the command you are searching for. Each character you enter narrows down the search. The corresponding search result is shown on the right side of the colon whereas your input appears on the left of the colon. To accept a search result, press Esc. The prompt now changes to its normal appearance and shows the command you chose. You can now edit the command or directly execute it by pressing Enter.

Completion

Completing a filename or directory name to its full length after typing its first letters is another helpful feature of Bash. To do so, type the first letters then press →| (Tabulator). If the filename or path can be uniquely identified, it is completed at once and the cursor moves to the end of the filename. You can then enter the next option of the command, if necessary. If the filename or path cannot be uniquely identified (because there are several filenames starting with the same letters), the filename or path is only completed up to the point where it becomes ambiguous again. You can then obtain a list of them by pressing →| a second time. After this, you can enter the next letters of the file or path then try completion again by pressing →|. When completing filenames and paths with →|, you can simultaneously check whether the file or path you want to enter really exists (and you can be sure of getting the spelling right).

Wild Cards

You can replace one or more characters in a filename with a wild card for pathname expansion. Wild cards are characters that can stand for other characters. There are three different types of these in Bash:

Wild Card	Function
?	Matches exactly one arbitrary character
*	Matches any number of characters
[*SET*]	Matches one of the characters from the group specified inside the square brackets, which is represented here by the string *SET*.

14.7.1 Examples For Using History, Completion and Wildcards

The following examples illustrate how to make use of these convenient features of Bash.

PROCEDURE 14.6: USING HISTORY AND COMPLETION

If you already did the example *Section 14.4.1, "Examples for Working with Files and Directories"*, your shell buffer should be filled with commands which you can retrieve using the history function.

1. Press ↑ repeatedly until **cd ~** appears.
2. Press Enter to execute the command and to switch to your home directory.
   By default, your home directory contains two subdirectories starting with the same letter, `Documents` and `Desktop`.
3. Type **cd D** and press →|.
   Nothing happens since Bash cannot identify to which one of the subdirectories you want to change.
4. Press →| again to see the list of possible choices:

```
tux > cd D
Desktop/ Documents/ Downloads/
tux > cd D
```

5. The prompt still shows your initial input. Type the next character of the subdirectory you want to go to and press →| again.
   Bash now completes the path.
6. You can now execute the command with Enter.

PROCEDURE 14.7: USING WILDCARDS

Now suppose that your home directory contains several files with various file extensions. It also holds several versions of one file which you saved under different filenames `myfile1.txt`, `myfile2.txt` etc. You want to search for certain files according to their properties.

1. First, create some test files in your home directory:
   a. Use the **touch** command to create several (empty) files with different file extensions, for example `.pdf`, `.xml` and `.jpg`.

You can do this consecutively (do not forget to use the Bash history function) or with only one **touch** command: simply add several filenames separated by a space.

b. Create at least two files that have the same file extension, for example `.html`.

c. To create several "versions" of one file, enter

```
tux > touch myfile{1..5}.txt
```

This command creates five consecutively numbered files: `myfile1.txt`, ..., `myfile5.txt`.

d. List the contents of the directory. It should look similar to this:

```
tux > ls -l
-rw-r--r-- 1 tux users   0 2006-07-14 13:34 foo.xml
-rw-r--r-- 1 tux users   0 2006-07-14 13:47 home.html
-rw-r--r-- 1 tux users   0 2006-07-14 13:47 index.html
-rw-r--r-- 1 tux users   0 2006-07-14 13:47 toc.html
-rw-r--r-- 1 tux users   0 2006-07-14 13:34 manual.pdf
-rw-r--r-- 1 tux users   0 2006-07-14 13:49 myfile1.txt
-rw-r--r-- 1 tux users   0 2006-07-14 13:49 myfile2.txt
-rw-r--r-- 1 tux users   0 2006-07-14 13:49 myfile3.txt
-rw-r--r-- 1 tux users   0 2006-07-14 13:49 myfile4.txt
-rw-r--r-- 1 tux users   0 2006-07-14 13:49 myfile5.txt
-rw-r--r-- 1 tux users   0 2006-07-14 13:32 tux.png
```

2. With wild cards, select certain subsets of the files according to various criteria:

a. To list all files with the `.html` extension, enter

```
tux > ls -l *.html
```

b. To list all "versions" of `myfile.txt`, enter

```
tux > ls -l myfile?.txt
```

Note that you can only use the `?` wild card here because the numbering of the files is single-digit. As soon as you have a file named `myfile10.txt` you must to use the `*` wild card to view all versions of `myfile.txt` (or add another question mark, so your string looks like **`myfile??.txt`**).

c. To remove, for example, version 1-3 and version 5 of `myfile.txt`, enter

```
tux > rm myfile[1-3,5].txt
```

d. Check the result with

```
tux > ls -l
```

Of all `myfile.txt` versions only `myfile4.txt` should be left.

You can also combine several wild cards in one command. In the example above, **`rm myfile[1-3,5].*`** would lead to the same result as **`rm myfile[1-3,5].txt`** because there are only files with the extension `.txt` available.

Note: Using Wildcards in **`rm`** Commands

Wildcards in a **`rm`** command can be very useful but also dangerous: you might delete more files from your directory than intended. To see which files would be affected by the **`rm`**, run your wildcard string with **`ls`** instead of **`rm`** first.

14.8 Editing Texts

In order to edit files from the command line, you will need to know the vi editor. vi is a default editor which can be found on nearly every UNIX/Linux system. It can run several operating modes in which the keys you press have different functions. This does not make it very easy for beginners, but you should know at least the most basic operations with vi. There may be situations where no other editor than vi is available.

Basically, vi makes use of three operating modes:

***command* mode**

In this mode, vi accepts certain key combinations as commands. Simple tasks such as searching words or deleting a line can be executed.

***insert* mode**

In this mode, you can write normal text.

***extended* mode**

In this mode, also known as colon mode (as you have to enter a colon to switch to this mode), vi can execute also more complex tasks such as searching and replacing text.

In the following (very simple) example, you will learn how to open and edit a file with vi, how to save your changes and quit vi.

14.8.1 Example: Editing with vi

Note: Display of Keys

In the following, find several commands that you can enter in vi by just pressing keys. These appear in uppercase as on a keyboard. If you need to enter a key in uppercase, this is stated explicitly by showing a key combination including the `Shift` key.

1. To create and open a new file with vi, enter

   ```
   tux > vi textfile.txt
   ```

   By default, vi opens in *command* mode in which you cannot enter text.

2. Press `I` to switch to insert mode. The bottom line changes and indicates that you now can insert text.

3. Write some sentences. If you want to insert a new line, first press `Esc` to switch back to command mode. Press `O` to insert a new line and to switch to insert mode again.

4. In the insert mode, you can edit the text with the arrow keys and with `Del`.

5. To leave vi, press `Esc` to switch to command mode again. Then press `:` which takes you to the extended mode. The bottom line now shows a colon.

6. To leave vi and save your changes, type **`wq`** (`w` for `write`; `q` for `quit`) and press `Enter`. If you want to save the file under a different name, type **`w`** *`FILENAME`* and press `Enter`. To leave vi without saving, type **`q!`** instead and press `Enter`.

14.9 Searching for Files or Contents

Bash offers you several commands to search for files and to search for the contents of files:

`find`

With **find**, search for a file in a given directory. The first argument specifies the directory in which to start the search. The option `-name` must be followed by a search string, which may also include wild cards. Unlike **locate**, which uses a database, **find** scans the actual directory.

grep

The **grep** command finds a specific search string in the specified text files. If the search string is found, the command displays the line in which `searchstring` was found, along with the filename. If desired, use wild cards to specify filenames.

14.9.1 Examples for Searching

- To search your home directory for all occurrences of filenames that contain the file extension `.txt`, use:

```
tux > find ~ -name '*.txt' -print
```

- To search a directory (in this case, your home directory) for all occurrences of files which contain, for example, the word `music`, use:

```
tux > grep music ~/*
```

grep is case-sensitive by default. Hence, with the command above you will not find any files containing `Music`.To ignore case, use the `-i` option.

- To use a search string which consists of more than one word, enclose the string in double quotation marks, for example:

```
tux > grep "music is great" ~/*
```

14.10 Viewing Text Files

When searching for the contents of a file with **grep**, the output gives you the line in which the `searchstring` was found along with the filename. Often this contextual information is still not enough information to decide whether you want to open and edit this file. Bash offers you several commands to have a quick look at the contents of a text file directly in the shell, without opening an editor.

head

With **head** you can view the first lines of a text file. If you do not specify the command any further, **head** shows the first 10 lines of a text file.

tail

The **tail** command is the counterpart of **head**. If you use **tail** without any further options it displays the last 10 lines of a text file. This can be very useful to view log files of your system, where the most recent messages or log entries are usually found at the end of the file.

less

With **less**, display the whole contents of a text file. To move up and down half a page use Page ↑ and Page ↓. Use Space to scroll down one page. Home takes you to the beginning, and End to the end of the document. To end the viewing mode, press Q.

more

Instead of less, you can also use the older program more. It has basically the same function—however, it is less convenient because it does not allow you to scroll backward. Use Space to move forward. When you reach the end of the document, the viewer closes automatically.

cat

The **cat** command displays the contents of a file, printing the entire contents to the screen without interruption. As **cat** does not allow you to scroll it is not very useful as viewer but it is rather often used in combination with other commands.

14.11 Redirection and Pipes

Sometimes it would be useful if you could write the output of a command to a file for further editing or if you could combine several commands, using the output of one command as the input for the next one. The shell offers this function by means of redirection or pipes.

Normally, the standard output in the shell is your screen (or an open shell window) and the standard input is the keyboard. With certain symbols you can redirect the input or the output to another object, such as a file or another command.

Redirection

With > you can forward the output of a command to a file (output redirection), with < you can use a file as input for a command (input redirection).

Pipe

By means of a pipe symbol `|` you can also redirect the output: with a pipe, you can combine several commands, using the output of one command as input for the next command. In contrast to the other redirection symbols `>` and `<`, the use of the pipe is not constrained to files.

14.11.1 Examples for Redirection and Pipe

1. To write the output of a command like **`ls`** to a file, enter

```
tux > ls -l > filelist.txt
```

   This creates a file named `filelist.txt` that contains the list of contents of your current directory as generated by the **`ls`** command.
   However, if a file named `filelist.txt` already exists, this command overwrites the existing file. To prevent this, use `>>` instead of `>`. Entering

```
tux > ls -l >> filelist.txt
```

   simply appends the output of the **`ls`** command to an already existing file named `filelist.txt`. If the file does not exist, it is created.

2. Redirections also works the other way round. Instead of using the standard input from the keyboard for a command, you can use a file as input:

```
tux > sort < filelist.txt
```

   This will force the **`sort`** command to get its input from the contents of `filelist.txt`. The result is shown on the screen. Of course, you can also write the result into another file, using a combination of redirections:

```
tux > sort < filelist.txt > sorted_filelist.txt
```

3. If a command generates a lengthy output, like **`ls`** `-l` may do, it may be useful to pipe the output to a viewer like `less` to be able to scroll through the pages. To do so, enter

```
tux > ls -l | less
```

   The list of contents of the current directory is shown in `less`.

The pipe is also often used in combination with the **grep** command in order to search for a certain string in the output of another command. For example, if you want to view a list of files in a directory which are owned by the user `tux`, enter

```
tux > ls -l | grep tux
```

14.12 Starting Programs and Handling Processes

As you have seen in *Section 14.8, "Editing Texts"*, programs can be started from the shell. Applications with a graphical user interface need the X Window System and can only be started from a terminal window within a graphical user interface. For example, if you want to open a file named `vacation.pdf` in your home directory from a terminal window in KDE or GNOME, simply run **okular ~/vacation.pdf** (or **evince ~/vacation.pdf**) to start a PDF viewer displaying your file.

When looking at the terminal window again you will realize that the command line is blocked as long as the PDF viewer is open, meaning that your prompt is not available. To change this, press Ctrl–Z to suspend the process and enter **bg** to send the process to the background.

Now you can still have a look at `vacation.pdf` while your prompt is available for further commands. An easier way to achieve this is by sending a process to the background directly when starting it. To do so, add an ampersand at the end of the command:

```
tux > okular ~/vacation.pdf &
```

If you have started several background processes (also named jobs) from the same shell, the **jobs** command gives you an overview of the jobs. It also shows the job number in brackets and their status:

```
tux > jobs
[1]   Running        okular book.opensuse.startup-xep.pdf &
[2]-  Running        okular book.opensuse.reference-xep.pdf &
[3]+  Stopped        man jobs
```

To bring a job to the foreground again, enter **fg** *JOB_NUMBER*.

Whereas **job** only shows the background processes started from a specific shell, the **ps** command (run without options) shows a list of all your processes—those you started. Find an example output below:

```
tux > ps
```

```
PID TTY          TIME CMD
15500 pts/1    00:00:00 bash
28214 pts/1    00:00:00 okular
30187 pts/1    00:00:00 kwrite
30280 pts/1    00:00:00 ps
```

In case a program cannot be terminated in the normal way, use the **kill** command to stop the process (or processes) belonging to that program. To do so, specify the process ID (PID) shown by the output of **ps**. For example, to shut down the KWrite editor in the example above, enter

```
tux > kill 30187
```

This sends a *TERM* signal that instructs the program to shut itself down.

Alternatively, if the program or process you want to terminate is a background job and is shown by the **jobs** command, you can also use the **kill** command in combination with the job number to terminate this process. When identifying the job with the job number, you must prefix the number with a percent character (`%`):

```
tux > kill %JOB_NUMBER
```

If **kill** does not help—as is sometimes the case for "runaway" programs—try

```
tux > kill -9 PID
```

This sends a *KILL* signal instead of a *TERM* signal, usually bringing the specified process to an end.

This section is intended to introduce the most basic set of commands for handling jobs and processes. Find an overview for system administrators in *Book "*System Analysis and Tuning Guide*", Chapter 2 "System Monitoring Utilities", Section 2.3 "Processes"*.

14.13 Archives and Data Compression

On Linux, there are two types of commands that make data easier to transfer:

- Archivers, which create a big file out of several smaller ones. The most commonly used archiver is **tar**, another example is **cpio**.
- Compressors, which losslessly make a file smaller. The most commonly used compressors are **gzip** and **bzip2**.

When combining these two types of commands, their effect is comparable to the compressed archive files that are prevalent on other operating systems, for example, `ZIP` or `RAR`.

To pack the `test` directory with all its files and subdirectories into an archive named `testarchive.tar`, do the following:

PROCEDURE 14.8: ARCHIVING FILES

1. Open a shell.
2. Use **`cd`** to change to your home directory where the `test` directory is located.
3. Commpress the file with:

   ```
   tux > tar -cvf testarchive.tar test
   ```

   The `-c` option creates the archive, making it a file as directed by `-f`. The `-v` option lists the files as they are processed.
   The `test` directory with all its files and directories has remained unchanged on your hard disk.
4. View the contents of the archive file with:

   ```
   tux > tar -tf testarchive.tar
   ```
5. To unpack the archive, use:

   ```
   tux > tar -xvf testarchive.tar
   ```

   If files in your current directory are named the same as the files in the archive, they will be overwritten without warning.

To compress files, use **`gzip`** or, for better compression, **`bzip2`**.

PROCEDURE 14.9: COMPRESSING A FILE

1. For this example, reuse the archive `testarchive.tar` from *Procedure 14.8, "Archiving Files"*. To compress the archive, use:

   ```
   tux > gzip testarchive.tar
   ```

   With **`ls`**, now see that the file `testarchive.tar` is no longer there and that the file `testarchive.tar.gz` has been created instead.

As an alternative, use **bzip2 testarchive.tar** which works analogously but provides somewhat better compression.

2. Now decompress and unarchive the file again:

   - This can be done in two steps by first decompressing and then unarchiving the file:

     ```
     tux > gzip --decompress testarchive.tar.gz
     tux > tar -xvf testarchive.tar
     ```

   - You can also decompress and unarchive in one step:

     ```
     tux > tar -xvf testarchive.tar
     ```

   With **ls**, you can see that a new `test` directory has been created with the same contents as your `test` directory in your home directory.

14.14 Important Linux Commands

This section provides an overview of the most important Linux commands. There are many more commands than listed in this chapter. Along with the individual commands, parameters are listed and, where appropriate, a typical sample application is introduced.

Adjust the parameters to your needs. It makes no sense to write **ls file** if no file named `file` actually exists. You can usually combine several parameters, for example, by writing **ls -la** instead of **ls -l -a**.

14.14.1 File Commands

The following section lists the most important commands for file management. It covers everything from general file administration to the manipulation of file system ACLs.

14.14.1.1 File Administration

ls *OPTIONS FILES*

If you run **ls** without any additional parameters, the program lists the contents of the current directory in short form.

`-l`

Detailed list

`-a`

Displays hidden files

cp *OPTIONS SOURCE TARGET*

Copies `source` to `target`.

-i

Waits for confirmation, if necessary, before an existing `target` is overwritten

-r

Copies recursively (includes subdirectories)

mv *OPTIONS SOURCE TARGET*

Copies `source` to `target` then deletes the original `source`.

-b

Creates a backup copy of the `source` before moving

-i

Waits for confirmation, if necessary, before an existing `targetfile` is overwritten

rm *OPTIONS FILES*

Removes the specified files from the file system. Directories are not removed by **rm** unless the option `-r` is used.

`-r`

Deletes any existing subdirectories

`-i`

Waits for confirmation before deleting each file

ln *OPTIONS SOURCE TARGET*

Creates an internal link from *source* to *target*. Normally, such a link points directly to *source* on the same file system. However, if **ln** is executed with the `-s` option, it creates a symbolic link that only points to the directory in which `source` is located, enabling linking across file systems.

-s

Creates a symbolic link

cd *OPTIONS DIRECTORY*

Changes the current directory. **cd** without any parameters changes to the user's home directory.

mkdir *OPTIONS DIRECTORY*

Creates a new directory.

rmdir *OPTIONS DIRECTORY*

Deletes the specified directory if it is already empty.

chown *OPTIONS USER_NAME*[:*GROUP*] *FILES*

Transfers ownership of a file to the user with the specified user name.

`-R`

Changes files and directories in all subdirectories

chgrp *OPTIONS GROUP_NAME FILES*

Transfers the group ownership of a given `file` to the group with the specified group name. The file owner can change group ownership only if a member of both the current and the new group.

chmod *OPTIONS MODE FILES*

Changes the access permissions.

The `mode` parameter has three parts: `group`, `access`, and `access type`. `group` accepts the following characters:

u

User

g

Group

o

Others

For `access`, grant access with `+` and deny it with `-`.

The `access type` is controlled by the following options:

r

Read

w

Write

x

Execute—executing files or changing to the directory

s

Setuid bit—the application or program is started as if it were started by the owner of the file

As an alternative, a numeric code can be used. The four digits of this code are composed of the sum of the values 4, 2, and 1—the decimal result of a binary mask. The first digit sets the set user ID (SUID) (4), the set group ID (2), and the sticky (1) bits. The second digit defines the permissions of the owner of the file. The third digit defines the permissions of the group members and the last digit sets the permissions for all other users. The read permission is set with 4, the write permission with 2, and the permission for executing a file is set with 1. The owner of a file would usually receive a 6 or a 7 for executable files.

gzip *PARAMETERS FILES*

This program compresses the contents of files using complex mathematical algorithms. Files compressed in this way are given the extension `.gz` and need to be uncompressed before they can be used. To compress several files or even entire directories, use the **tar** command.

-d

Decompresses the packed gzip files so they return to their original size and can be processed normally (like the command **gunzip**)

tar *OPTIONS ARCHIVE FILES*

tar puts one or more files into an archive. Compression is optional. **tar** is a quite complex command with several options available. The most frequently used options are:

`-f`

Writes the output to a file and not to the screen as is usually the case

`-c`

Creates a new TAR archive

`-r`

Adds files to an existing archive

`-t`

Outputs the contents of an archive

`-u`

Adds files, but only if they are newer than the files already contained in the archive

`-x`

Unpacks files from an archive (*extraction*)

`-z`

Packs the resulting archive with **`gzip`**

`-j`

Compresses the resulting archive with **`bzip2`**

`-v`

Lists files processed

The archive files created by **`tar`** end with `.tar`. If the TAR archive was also compressed using **`gzip`**, the ending is `.tgz` or `.tar.gz`. If it was compressed using **`bzip2`**, the ending is `.tar.bz2`.

`find` *OPTIONS*

With **`find`**, search for a file in a given directory. The first argument specifies the directory in which to start the search. The option **`-name`** must be followed by a search string, which may also include wild cards. Unlike **`locate`**, which uses a database, **`find`** scans the actual directory.

14.14.1.2 Commands to Access File Contents

`file` *OPTIONS FILES*

In Linux, files can have a file extensions but do not need to have one. The **`file`** determines the file type of a given file. With the output of **`file`**, you can then choose an appropriate application with which to open the file.

-z

Tries to look inside compressed files

`cat` *OPTIONS FILES*

The **`cat`** command displays the contents of a file, printing the entire contents to the screen without interruption.

-n

Numbers the output on the left margin

less *OPTIONS FILES*

This command can be used to browse the contents of the specified file. Scroll half a screen page up or down with Page ↑ and Page ↓ or a full screen page down with Space. Jump to the beginning or end of a file using Home and End. Press Q to quit the program.

grep *OPTIONS SEARCH_STRING FILES*

The **grep** command finds a specific search string in the specified files. If the search string is found, the command displays the line in which *SEARCH_STRING* was found along with the file name.

`-i`

Ignores case

`-H`

Only displays the names of the relevant files, but not the text lines

`-n`

Additionally displays the numbers of the lines in which it found a hit

`-l`

Only lists the files in which `searchstring` does not occur

diff *OPTIONS FILE_1 FILE_2*

The **diff** command compares the contents of any two files. The output produced by the program lists all lines that do not match. This is frequently used by programmers who need only to send their program alterations and not the entire source code.

`-q`

Only reports whether the two files differ

`-u`

Produces a “unified” diff, which makes the output more readable

14.14.1.3 File Systems

mount *OPTIONS DEVICE MOUNT_POINT*

This command can be used to mount any data media, such as hard disks, CD-ROM drives, and other drives, to a directory of the Linux file system.

`-r`

Mount read-only

`-t` *`FILE_SYSTEM`*

Specify the file system: For Linux hard disks, this is commonly `ext4`, `xfs`, or `btrfs`. For hard disks not defined in the file `/etc/fstab`, the device type must also be specified. In this case, only `root` can mount it. If the file system needs to also be mounted by other users, enter the option `user` in the appropriate line in the `/etc/fstab` file (separated by commas) and save this change. Further information is available in the `mount(1)` man page.

`umount` *`OPTIONS MOUNT_POINT`*

This command unmounts a mounted drive from the file system. To prevent data loss, run this command before taking a removable data medium from its drive. Normally, only `root` is allowed to run the commands **`mount`** and **`umount`**. To enable other users to run these commands, edit the `/etc/fstab` file to specify the option `user` for the relevant drive.

14.14.2 System Commands

The following section lists a few of the most important commands needed for retrieving system information and controlling processes and the network.

14.14.2.1 System Information

`df` *`OPTIONS DIRECTORY`*

The **`df`** (disk free) command, when used without any options, displays information about the total disk space, the disk space currently in use, and the free space on all the mounted drives. If a directory is specified, the information is limited to the drive on which that directory is located.

`-h`

Shows the number of occupied blocks in gigabytes, megabytes, or kilobytes—in human-readable format

`-T`

Type of file system (ext2, nfs, etc.)

`du` *`OPTIONS PATH`*

This command, when executed without any parameters, shows the total disk space occupied by files and subdirectories in the current directory.

`-a`
: Displays the size of each individual file

`-h`
: Output in human-readable form

`-s`
: Displays only the calculated total size

`free` *`OPTIONS`*
: The command **`free`** displays information about RAM and swap space usage, showing the total and the used amount in both categories. See *Book "Reference", Chapter 15 "Special System Features", Section 15.1.7 "The* **`free`** *Command"* for more information.

`-b`
: Output in bytes

`-k`
: Output in kilobytes

`-m`
: Output in megabytes

`date` *`OPTIONS`*
: This simple program displays the current system time. If run as `root`, it can also be used to change the system time. Details about the program are available in the date(1) man page.

14.14.2.2 Processes

`top` *`OPTIONS`*
: **`top`** provides a quick overview of the currently running processes. Press H to access a page that briefly explains the main options for customizing the program.

`ps` *`OPTIONS PROCESS_ID`*
: If run without any options, this command displays a table of all your own programs or processes—those you started. The options for this command are not preceded by hyphen.

`aux`

Displays a detailed list of all processes, independent of the owner

kill *OPTIONS PROCESS_ID*

Unfortunately, sometimes a program cannot be terminated in the normal way. In most cases, you should still be able to stop such a runaway program by executing the **kill** command, specifying the respective process ID (see **top** and **ps**). **kill** sends a *TERM* signal that instructs the program to shut itself down. If this does not help, the following parameter can be used:

-9

Sends a *KILL* signal instead of a *TERM* signal, bringing the specified process to an end in almost all cases

killall *OPTIONS PROCESS_NAME*

This command is similar to **kill**, but uses the process name (instead of the process ID) as an argument, ending all processes with that name.

14.14.2.3 Network

ping *OPTIONS HOSTNAME_OR_IP_ADDRESS*

The **ping** command is the standard tool for testing the basic functionality of TCP/IP networks. It sends a small data packet to the destination host, requesting an immediate reply. If this works, **ping** displays a message to that effect, which indicates that the network link is functioning.

`-c` *NUMBER*

Determines the total number of packages to send and ends after they have been dispatched (by default, there is no limitation set)

`-f`

flood ping: sends as many data packages as possible; a popular means, reserved for `root`, to test networks

`-i` *VALUE*

Specifies the interval between two data packages in seconds (default: one second)

host *OPTIONS HOSTNAME SERVER*

The domain name system resolves domain names to IP addresses. With this tool, send queries to name servers (DNS servers).

ssh *OPTIONS* [*USER@*]*HOSTNAME* *COMMAND*

SSH is actually an Internet protocol that enables you to work on remote hosts across a network. SSH is also the name of a Linux program that uses this protocol to enable operations on remote computers.

14.14.2.4 Miscellaneous

passwd *OPTIONS* *USER_NAME*

Users may change their own passwords at any time using this command. The administrator `root` can use the command to change the password of any user on the system.

su *OPTIONS* *USER_NAME*

The **su** command makes it possible to log in under a different user name from a running session. Specify a user name and the corresponding password. The password is not required from `root`, because `root` is authorized to assume the identity of any user. When using the command without specifying a user name, you are prompted for the `root` password and change to the superuser (`root`). Use **su -** to start a login shell for a different user.

halt *OPTIONS*

To avoid loss of data, you should always use this program to shut down your system.

reboot *OPTIONS*

Does the same as **halt** except the system performs an immediate reboot.

clear

This command cleans up the visible area of the console. It has no options.

14.14.3 For More Information

There are many more commands than listed in this chapter. For information about other commands or more detailed information, also see the publication *Linux in a Nutshell* by O'Reilly.

15 Bash and Bash Scripts

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

15.1 What is “The Shell”?

Traditionally, *the* shell is Bash (Bourne again Shell). When this chapter speaks about “the shell” it means Bash. There are more shells available (ash, csh, ksh, zsh, …), each employing different features and characteristics. If you need further information about other shells, search for *shell* in YaST.

15.1.1 Knowing the Bash Configuration Files

A shell can be invoked as an:

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

Depending on the type of shell you use, different configuration files will be read. The following tables show the login and non-login shell configuration files.

TABLE 15.1: BASH CONFIGURATION FILES FOR LOGIN SHELLS

File	Description
`/etc/profile`	Do not modify this file, otherwise your modifications may be destroyed during your next update!

File	Description
`/etc/profile.local`	Use this file if you extend `/etc/profile`
`/etc/profile.d/`	Contains system-wide configuration files for specific programs
`~/.profile`	Insert user specific configuration for login shells here

Note that the login shell also sources the configuration files listed under *Table 15.2, "Bash Configuration Files for Non-Login Shells"*.

TABLE 15.2: BASH CONFIGURATION FILES FOR NON-LOGIN SHELLS

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

Additionally, Bash uses some more files:

TABLE 15.3: SPECIAL FILES FOR BASH

File	Description
`~/.bash_history`	Contains a list of all commands you have typed
`~/.bash_logout`	Executed when logging out
`~/.alias`	User defined aliases of frequently used commands. See **`man 1 alias`** for more details about defining aliases.

15.1.2 The Directory Structure

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

TABLE 15.4: OVERVIEW OF A STANDARD DIRECTORY TREE

Directory	Contents
`/`	Root directory—the starting point of the directory tree.
`/bin`	Essential binary files, such as commands that are needed by both the system administrator and normal users. Usually also contains the shells, such as Bash.
`/boot`	Static files of the boot loader.
`/dev`	Files needed to access host-specific devices.
`/etc`	Host-specific system configuration files.
`/home`	Holds the home directories of all users who have accounts on the system. However, `root`'s home directory is not located in `/home` but in `/root`.
`/lib`	Essential shared libraries and kernel modules.
`/media`	Mount points for removable media.
`/mnt`	Mount point for temporarily mounting a file system.
`/opt`	Add-on application software packages.
`/root`	Home directory for the superuser `root`.
`/sbin`	Essential system binaries.
`/srv`	Data for services provided by the system.
`/tmp`	Temporary files.
`/usr`	Secondary hierarchy with read-only data.

Directory	Contents
`/var`	Variable data such as log files.
`/windows`	Only available if you have both Microsoft Windows* and Linux installed on your system. Contains the Windows data.

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

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

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

`/dev`
: Holds device files that represent hardware components.

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

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

Note: Home Directory in a Network Environment

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

`/lib`

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

`/media`

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

`/mnt`

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

`/opt`

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

`/root`

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

`/run`

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

`/sbin`

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

`/srv`

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

`/tmp`

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

Important: Cleaning up `/tmp` at Boot Time

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

`/usr`

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

`/usr/bin`

Contains generally accessible programs.

`/usr/sbin`

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

`/usr/local`

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

`/usr/share/doc`

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

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

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

`/var`

Whereas `/usr` holds static, read-only data, `/var` is for data which is written during system operation and thus is variable data, such as log files or spooling data. For an overview of the most important log files you can find under `/var/log/`, refer to *Table 17.1, "Log Files"*.

`/windows`

Only available if you have both Microsoft Windows and Linux installed on your system. Contains the Windows data available on the Windows partition of your system. Whether you can edit the data in this directory depends on the file system your Windows partition

uses. If it is FAT32, you can open and edit the files in this directory. For NTFS, openSUSE Leap also includes write access support. However, the driver for the NTFS-3g file system has limited functionality.

15.2 Writing Shell Scripts

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

EXAMPLE 15.1: A SHELL SCRIPT PRINTING A TEXT

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

❶ The first line begins with the *Shebang* characters (`#!`) which indicate that this file is a script. The interpreter, specified after the *Shebang*, executes the script. In this case, the specified interpreter is **`/bin/sh`**.

❷ The second line is a comment beginning with the hash sign. We recommend that you comment difficult lines. With proper commenting, you can remember the purpose and function of the line. Also, other readers will hopefully understand your script. Commenting is considered good practice in the development community.

❸ The third line uses the built-in command **`echo`** to print the corresponding text.

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

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

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

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

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

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

1. **As Absolute Path.** The script can be executed with an absolute path. In our case, it is `~/bin/hello.sh`.
2. **Everywhere.** If the `PATH` environment variable contains the directory where the script is located, you can execute the script with `hello.sh`.

15.3 Redirecting Command Events

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

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

To redirect these channels, there are the following possibilities:

`Command > File`

Saves the output of the command into a file, an existing file will be deleted. For example, the `ls` command writes its output into the file `listing.txt`:

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

`Command >> File`

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

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

`Command < File`

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

```
tux > read a < foo
```

`Command1 | Command2`

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

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

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

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

15.4 Using Aliases

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

```
alias NAME=DEFINITION
```

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

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

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

15.5 Using Variables in Bash

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

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

```
tux > printenv PATH
```

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

```
tux > echo $PATH
```

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

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

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

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

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

```
tux > unset NAME
```

The following table contains some common environment variables which can be used in you shell scripts:

TABLE 15.5: USEFUL ENVIRONMENT VARIABLES

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

15.5.1 Using Argument Variables

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

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

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

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

```
#!/bin/sh
echo \"$1\" \"$2\" \"$3\" \"$4\"
```

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

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

15.5.2 Using Variable Substitution

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

`${VAR#pattern}`

removes the shortest possible match from the left:

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

`${VAR##pattern}`

removes the longest possible match from the left:

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

`${VAR%pattern}`

removes the shortest possible match from the right:

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

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

`${VAR%%pattern}`

removes the longest possible match from the right:

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

`${VAR/pattern_1/pattern_2}`

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

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

15.6 Grouping and Combining Commands

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

The following list shows, how commands can be grouped:

`Command1 ; Command2`

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

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

`Command1 && Command2`

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

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

`Command1 || Command2`

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

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

`funcname(){ ... }`

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

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

You can call this function like this:

```
tux > hello Tux
```

which prints:

```
Hello Tux
```

15.7 Working with Common Flow Constructs

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

15.7.1 The if Control Command

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

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

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

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

```
fi
```

The test expression can also be abbreviated in square brackets:

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

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

15.7.2 Creating Loops with the `for` Command

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

```
for i in *.png; do
 ls -l $i
done
```

15.8 For More Information

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

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

V Help and Troubleshooting

16 Help and Documentation

openSUSE® Leap comes with various sources of information and documentation, many of which are already integrated into your installed system.

Documentation in `/usr/share/doc`

This traditional help directory holds various documentation files and release notes for your system. It contains also information of installed packages in the subdirectory `packages`. Find more detailed information in *Section 16.1, "Documentation Directory"*.

Man Pages and Info Pages for Shell Commands

When working with the shell, you do not need to know the options of the commands by heart. Traditionally, the shell provides integrated help by means of man pages and info pages. Read more in *Section 16.2, "Man Pages"* and *Section 16.3, "Info Pages"*.

Desktop Help Center

The help center of the GNOME desktop (Help) provides central access to the most important documentation resources on your system in searchable form. These resources include online help for installed applications, man pages, info pages, and the SUSE manuals delivered with your product.

Separate Help Packages for Some Applications

When installing new software with YaST, the software documentation is usually installed automatically and appears in the help center of your desktop. However, some applications, such as GIMP, may have different online help packages that can be installed separately with YaST and do not integrate into the help centers.

16.1 Documentation Directory

The traditional directory to find documentation on your installed Linux system is `/usr/share/doc`. Usually, the directory contains information about the packages installed on your system, plus release notes, manuals, and more.

Note: Contents Depends on Installed Packages

In the Linux world, many manuals and other kinds of documentation are available in the form of packages, like software. How much and which information you find in `/usr/share/docs` also depends on the (documentation) packages installed. If you cannot find the subdirectories mentioned here, check if the respective packages are installed on your system and add them with YaST, if needed.

16.1.1 SUSE Manuals

We provide HTML and PDF versions of our books in different languages. In the `manual` subdirectory, find HTML versions of most of the SUSE manuals available for your product. For an overview of all documentation available for your product refer to the preface of the manuals.

If more than one language is installed, `/usr/share/doc/manual` may contain different language versions of the manuals. The HTML versions of the SUSE manuals are also available in the help center of both desktops. For information on where to find the PDF and HTML versions of the books on your installation media, refer to the openSUSE Leap Release Notes. They are available on your installed system under `/usr/share/doc/release-notes/` or online at your product-specific Web page at https://doc.opensuse.org/release-notes/.

16.1.2 Package Documentation

Under `packages`, find the documentation that is included in the software packages installed on your system. For every package, a subdirectory `/usr/share/doc/packages/`*`PACKAGENAME`* is created. It often contains README files for the package and sometimes examples, configuration files, or additional scripts. The following list introduces typical files to be found under `/usr/share/doc/packages`. None of these entries are mandatory and many packages might only include a few of them.

`AUTHORS`
: List of the main developers.

`BUGS`
: Known bugs or malfunctions. Might also contain a link to a Bugzilla Web page where you can search all bugs.

`CHANGES`, `ChangeLog`

Summary of changes from version to version. Usually interesting for developers, because it is very detailed.

`COPYING`, `LICENSE`

Licensing information.

`FAQ`

Question and answers collected from mailing lists or newsgroups.

`INSTALL`

How to install this package on your system. As the package is already installed by the time you get to read this file, you can safely ignore the contents of this file.

`README`, `README.*`

General information on the software. For example, for what purpose and how to use it.

`TODO`

Things that are not implemented yet, but probably will be in the future.

`MANIFEST`

List of files with a brief summary.

`NEWS`

Description of what is new in this version.

16.2 Man Pages

Man pages are an essential part of any Linux system. They explain the usage of a command and all available options and parameters. Man pages can be accessed with **`man`** followed by the name of the command, for example, **`man ls`**.

Man pages are displayed directly in the shell. To navigate them, move up and down with Page ↑ and Page ↓. Move between the beginning and the end of a document with Home and End. End this viewing mode by pressing Q. Learn more about the **`man`** command itself with **`man man`**. Man pages are sorted in categories as shown in *Table 16.1, "Man Pages—Categories and Descriptions"* (taken from the man page for man itself).

TABLE 16.1: MAN PAGES—CATEGORIES AND DESCRIPTIONS

Number	Description
1	Executable programs or shell commands
2	System calls (functions provided by the kernel)
3	Library calls (functions within program libraries)
4	Special files (usually found in `/dev`)
5	File formats and conventions (`/etc/fstab`)
6	Games
7	Miscellaneous (including macro packages and conventions), for example, man(7), groff(7)
8	System administration commands (usually only for `root`)
9	Kernel routines (nonstandard)

Each man page consists of several parts labeled *NAME* , *SYNOPSIS* , *DESCRIPTION* , *SEE ALSO* , *LICENSING* , and *AUTHOR* . There may be additional sections available depending on the type of command.

16.3 Info Pages

Info pages are another important source of information on your system. Usually, they are more detailed than man pages. They consist of more than command line options and contain sometimes whole tutorials or reference documentation. To view the info page for a certain command, enter **`info`** followed by the name of the command, for example, **`info ls`** . You can browse an info page with a viewer directly in the shell and display the different sections, called "nodes". Use Space to move forward and <- to move backward. Within a node, you can also

browse with Page ↑ and Page ↓ but only Space and <– will take you also to the previous or subsequent node. Press Q to end the viewing mode. Not every command comes with an info page and vice versa.

16.4 Online Resources

In addition to the online versions of the SUSE manuals installed under `/usr/share/doc`, you can also access the product-specific manuals and documentation on the Web. For an overview of all documentation available for openSUSE Leap check out your product-specific documentation Web page at http:/doc.opensuse.org/.

If you are searching for additional product-related information, you can also refer to the following Web sites:

SUSE Forums

There are several forums where you can dive in on discussions about SUSE products. See http://forums.opensuse.org/ for a list.

GNOME Documentation

Documentation for GNOME users, administrators and developers is available at http://library.gnome.org/.

The Linux Documentation Project

The Linux Documentation Project (TLDP) is run by a team of volunteers who write Linux-related documentation (see http://www.tldp.org). It is probably the most comprehensive documentation resource for Linux. The set of documents contains tutorials for beginners, but is mainly focused on experienced users and professional system administrators. TLDP publishes HOWTOs, FAQs, and guides (handbooks) under a free license. Parts of the documentation from TLDP are also available on openSUSE Leap.

You can also try general-purpose search engines. For example, use the search terms `Linux CD-RW help` or `OpenOffice file conversion problem` if you have trouble with burning CDs or LibreOffice file conversion.

17 Common Problems and Their Solutions

This chapter describes a range of potential problems and their solutions. Even if your situation is not precisely listed here, there may be one similar enough to offer hints to the solution of your problem.

17.1 Finding and Gathering Information

Linux reports things in a very detailed way. There are several places to look when you encounter problems with your system, most of which are standard to Linux systems in general, and some are relevant to openSUSE Leap systems. Most log files can be viewed with YaST (*Miscellaneous* › *Start-Up Log*).

YaST offers the possibility to collect all system information needed by the support team. Use *Other* › *Support* and select the problem category. When all information is gathered, attach it to your support request.

A list of the most frequently checked log files follows with the description of their typical purpose. Paths containing `~` refer to the current user's home directory.

TABLE 17.1: LOG FILES

Log File	Description
`~/.xsession-errors`	Messages from the desktop applications currently running.
`/var/log/apparmor/`	Log files from AppArmor, see *Book "*Security Guide" for detailed information.
`/var/log/audit/audit.log`	Log file from Audit to track any access to files, directories, or resources of your system, and trace system calls. See *Book "*Security Guide" for detailed information.
`/var/log/mail.*`	Messages from the mail system.
`/var/log/NetworkManager`	Log file from NetworkManager to collect problems with network connectivity

Log File	Description
`/var/log/samba/`	Directory containing Samba server and client log messages.
`/var/log/warn`	All messages from the kernel and system log daemon with the “warning” level or higher.
`/var/log/wtmp`	Binary file containing user login records for the current machine session. View it with **`last`**.
`/var/log/Xorg.*.log`	Various start-up and runtime log files from the X Window System. It is useful for debugging failed X start-ups.
`/var/log/YaST2/`	Directory containing YaST's actions and their results.
`/var/log/zypper.log`	Log file of Zypper.

Apart from log files, your machine also supplies you with information about the running system. See *Table 17.2: System Information With the `/proc` File System*

TABLE 17.2: SYSTEM INFORMATION WITH THE `/proc` FILE SYSTEM

File	Description
`/proc/cpuinfo`	Contains processor information, including its type, make, model, and performance.
`/proc/dma`	Shows which DMA channels are currently being used.
`/proc/interrupts`	Shows which interrupts are in use, and how many of each have been in use.
`/proc/iomem`	Displays the status of I/O (input/output) memory.

File	Description
`/proc/ioports`	Shows which I/O ports are in use at the moment.
`/proc/meminfo`	Displays memory status.
`/proc/modules`	Displays the individual modules.
`/proc/mounts`	Displays devices currently mounted.
`/proc/partitions`	Shows the partitioning of all hard disks.
`/proc/version`	Displays the current version of Linux.

Apart from the `/proc` file system, the Linux kernel exports information with the `sysfs` module, an in-memory file system. This module represents kernel objects, their attributes and relationships. For more information about `sysfs`, see the context of udev in *Book "*Reference*", Chapter 16 "Dynamic Kernel Device Management with* `udev`*"*. *Table 17.3* contains an overview of the most common directories under `/sys`.

TABLE 17.3: SYSTEM INFORMATION WITH THE `/sys` FILE SYSTEM

File	Description
`/sys/block`	Contains subdirectories for each block device discovered in the system. Generally, these are mostly disk type devices.
`/sys/bus`	Contains subdirectories for each physical bus type.
`/sys/class`	Contains subdirectories grouped together as a functional types of devices (like graphics, net, printer, etc.)
`/sys/device`	Contains the global device hierarchy.

Linux comes with several tools for system analysis and monitoring. See *Book "*System Analysis and Tuning Guide*", Chapter 2 "System Monitoring Utilities"* for a selection of the most important ones used in system diagnostics.

Each of the following scenarios begins with a header describing the problem followed by a paragraph or two offering suggested solutions, available references for more detailed solutions, and cross-references to other scenarios that are related.

17.2 Boot Problems

Boot problems are situations when your system does not boot properly (does not boot to the expected target and login screen).

17.2.1 The GRUB 2 Boot Loader Fails to Load

If the hardware is functioning properly, it is possible that the boot loader is corrupted and Linux cannot start on the machine. In this case, it is necessary to repair the boot loader. To do so, you need to start the Rescue System as described in *Section 17.5.2, "Using the Rescue System"* and follow the instructions in *Section 17.5.2.4, "Modifying and Re-installing the Boot Loader"*.

Alternatively, you can use the Rescue System to fix the boot loader as follows. Boot your machine from the installation media. In the boot screen, choose *More* › *Boot Linux System*. Select the disk containing the installed system and kernel with the default kernel options.

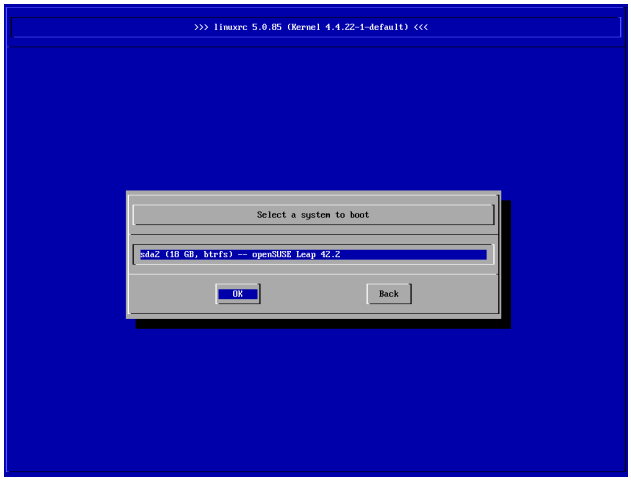


FIGURE 17.1: SELECT DISK

When the system is booted, start YaST and switch to *System* › *Boot Loader*. Make sure that the *Write generic Boot Code to MRB* option is enabled, and click *OK*. This fixes the corrupted boot loader by overwriting it, or installs the boot loader if it is missing.

Other reasons for the machine not booting may be BIOS-related:

BIOS Settings

Check your BIOS for references to your hard disk. GRUB 2 may simply not be started if the hard disk itself cannot be found with the current BIOS settings.

BIOS Boot Order

Check whether your system's boot order includes the hard disk. If the hard disk option was not enabled, your system may install properly, but fails to boot when access to the hard disk is required.

17.2.2 No Login or Prompt Appears

This behavior typically occurs after a failed kernel upgrade and it is known as a *kernel panic* because of the type of error on the system console that sometimes can be seen at the final stage of the process. If, in fact, the machine has just been rebooted following a software update, the immediate goal is to reboot it using the old, proven version of the Linux kernel and associated files. This can be done in the GRUB 2 boot loader screen during the boot process as follows:

1. Reboot the computer using the reset button, or switch it off and on again.
2. When the GRUB 2 boot screen becomes visible, select the *Advanced Options* entry and choose the previous kernel from the menu. The machine will boot using the prior version of the kernel and its associated files.
3. After the boot process has completed, remove the newly installed kernel and, if necessary, set the default boot entry to the old kernel using the YaST *Boot Loader* module. For more information refer to *Book "*Reference*", Chapter 12 "The Boot Loader GRUB 2", Section 12.3 "Configuring the Boot Loader with YaST"*. However, doing this is probably not necessary because automated update tools normally modify it for you during the rollback process.
4. Reboot.

If this does not fix the problem, boot the computer using the installation media. After the machine has booted, continue with *Step 3*.

17.2.3 No Graphical Login

If the machine starts, but does not boot into the graphical login manager, anticipate problems either with the choice of the default systemd target or the configuration of the X Window System. To check the current systemd default target run the command **`sudo systemctl get-default`**. If the value returned is *not* `graphical.target`, run the command **`sudo systemctl isolate graphical.target`**. If the graphical login screen starts, log in and start *YaST* › *System* › *Services Manager* and set the *Default System Target* to *Graphical Interface*. From now on the system should boot into the graphical login screen.

If the graphical login screen does not start even if having booted or switched to the graphical target, your desktop or X Window software is probably misconfigured or corrupted. Examine the log files at `/var/log/Xorg.*.log` for detailed messages from the X server as it attempted to start. If the desktop fails during start, it may log error messages to the system journal that can be queried with the command **`journalctl`** (see *Book "*Reference*", Chapter 11 "***`journalctl`***: Query the* `systemd` *Journal"* for more information). If these error messages hint at a configuration problem in the X server, try to fix these issues. If the graphical system still does not come up, consider reinstalling the graphical desktop.

17.2.4 Root Btrfs Partition Cannot Be Mounted

If a `btrfs` root partition becomes corrupted, try the following options:

- Mount the partition with the `-o recovery` option.
- If that fails, run **`btrfs-zero-log`** on your root partition.

17.2.5 Force Checking Root Partitions

If the root partition becomes corrupted, use the parameter `forcefsck` on the boot prompt. This passes the option `-f` (force) to the **`fsck`** command.

17.3 Login Problems

Login problems occur when your machine does boot to the expected welcome screen or login prompt, but refuses to accept the user name and password, or accepts them but then does not behave properly (fails to start the graphic desktop, produces errors, drops to a command line, etc.).

17.3.1 Valid User Name and Password Combinations Fail

This usually occurs when the system is configured to use network authentication or directory services and, for some reason, cannot retrieve results from its configured servers. The `root` user, as the only local user, is the only user that can still log in to these machines. The following are some common reasons a machine appears functional but cannot process logins correctly:

- The network is not working. For further directions on this, turn to *Section 17.4, "Network Problems"*.
- DNS is not working at the moment (which prevents GNOME from working and the system from making validated requests to secure servers). One indication that this is the case is that the machine takes an extremely long time to respond to any action. Find more information about this topic in *Section 17.4, "Network Problems"*.
- If the system is configured to use Kerberos, the system's local time may have drifted past the accepted variance with the Kerberos server time (this is typically 300 seconds). If NTP (network time protocol) is not working properly or local NTP servers are not working, Kerberos authentication ceases to function because it depends on common clock synchronization across the network.
- The system's authentication configuration is misconfigured. Check the PAM configuration files involved for any typographical errors or misordering of directives. For additional background information about PAM and the syntax of the configuration files involved, refer to *Book "Security Guide", Chapter 2 "Authentication with PAM"*.
- The home partition is encrypted. Find more information about this topic in *Section 17.3.3, "Login to Encrypted Home Partition Fails"*.

In all cases that do not involve external network problems, the solution is to reboot the system into single-user mode and repair the configuration before booting again into operating mode and attempting to log in again. To boot into single-user mode:

1. Reboot the system. The boot screen appears, offering a prompt.
2. Press `Esc` to exit the splash screen and get to the GRUB 2 text-based menu.
3. Press `B` to enter the GRUB 2 editor.
4. Add the following parameter to the line containing the kernel parameters:

   ```
   systemd.unit=rescue.target
   ```

5. Press `F10`.
6. Enter the user name and password for `root`.
7. Make all the necessary changes.
8. Boot into the full multiuser and network mode by entering **`systemctl isolate graphical.target`** at the command line.

17.3.2 Valid User Name and Password Not Accepted

This is by far the most common problem users encounter, because there are many reasons this can occur. Depending on whether you use local user management and authentication or network authentication, login failures occur for different reasons.

Local user management can fail for the following reasons:

- The user may have entered the wrong password.
- The user's home directory containing the desktop configuration files is corrupted or write protected.
- There may be problems with the X Window System authenticating this particular user, especially if the user's home directory has been used with another Linux distribution prior to installing the current one.

To locate the reason for a local login failure, proceed as follows:

1. Check whether the user remembered their password correctly before you start debugging the whole authentication mechanism. If the user may have not have remembered their password correctly, use the YaST User Management module to change the user's password. Pay attention to the Caps Lock key and unlock it, if necessary.

2. Log in as `root` and check the system journal with **`journalctl -e`** for error messages of the login process and of PAM.

3. Try to log in from a console (using Ctrl–Alt–F1). If this is successful, the blame cannot be put on PAM, because it is possible to authenticate this user on this machine. Try to locate any problems with the X Window System or the GNOME desktop. For more information, refer to *Section 17.3.4, "Login Successful but GNOME Desktop Fails"*.

4. If the user's home directory has been used with another Linux distribution, remove the `Xauthority` file in the user's home. Use a console login via Ctrl–Alt–F1 and run **`rm .Xauthority`** as this user. This should eliminate X authentication problems for this user. Try graphical login again.

5. If the desktop could not start because of corrupt configuration files, proceed with *Section 17.3.4, "Login Successful but GNOME Desktop Fails"*.

In the following, common reasons a network authentication for a particular user may fail on a specific machine are listed:

- The user may have entered the wrong password.
- The user name exists in the machine's local authentication files and is also provided by a network authentication system, causing conflicts.
- The home directory exists but is corrupt or unavailable. Perhaps it is write protected or is on a server that is inaccessible at the moment.
- The user does not have permission to log in to that particular host in the authentication system.
- The machine has changed host names, for whatever reason, and the user does not have permission to log in to that host.

- The machine cannot reach the authentication server or directory server that contains that user's information.
- There may be problems with the X Window System authenticating this particular user, especially if the user's home has been used with another Linux distribution prior to installing the current one.

To locate the cause of the login failures with network authentication, proceed as follows:

1. Check whether the user remembered their password correctly before you start debugging the whole authentication mechanism.
2. Determine the directory server which the machine relies on for authentication and make sure that it is up and running and properly communicating with the other machines.
3. Determine that the user's user name and password work on other machines to make sure that their authentication data exists and is properly distributed.
4. See if another user can log in to the misbehaving machine. If another user can log in without difficulty or if `root` can log in, log in and examine the system journal with **`journalctl -e`** > file. Locate the time stamps that correspond to the login attempts and determine if PAM has produced any error messages.
5. Try to log in from a console (using Ctrl–Alt–F1). If this is successful, the problem is not with PAM or the directory server on which the user's home is hosted, because it is possible to authenticate this user on this machine. Try to locate any problems with the X Window System or the GNOME desktop. For more information, refer to *Section 17.3.4, "Login Successful but GNOME Desktop Fails"*.
6. If the user's home directory has been used with another Linux distribution, remove the `Xauthority` file in the user's home. Use a console login via Ctrl–Alt–F1 and run **`rm .Xauthority`** as this user. This should eliminate X authentication problems for this user. Try graphical login again.
7. If the desktop could not start because of corrupt configuration files, proceed with *Section 17.3.4, "Login Successful but GNOME Desktop Fails"*.

17.3.3 Login to Encrypted Home Partition Fails

It is recommended to use an encrypted home partition for laptops. If you cannot log in to your laptop, the reason is usually simple: your partition could not be unlocked.

During the boot time, you need to enter the passphrase to unlock your encrypted partition. If you do not enter it, the boot process continues, leaving the partition locked.

To unlock your encrypted partition, proceed as follows:

1. Switch to the text console with Ctrl–Alt–F1.
2. Become `root`.
3. Restart the unlocking process again with:

```
root # systemctl restart home.mount
```

4. Enter your passphrase to unlock your encrypted partition.
5. Exit the text console and switch back to the login screen with Alt–F7.
6. Log in as usual.

17.3.4 Login Successful but GNOME Desktop Fails

If this is the case, it is likely that your GNOME configuration files have become corrupted. Some symptoms may include the keyboard failing to work, the screen geometry becoming distorted, or even the screen coming up as a bare gray field. The important distinction is that if another user logs in, the machine works normally. It is then likely that the problem can be fixed relatively quickly by simply moving the user's GNOME configuration directory to a new location, which causes GNOME to initialize a new one. Although the user is forced to reconfigure GNOME, no data is lost.

1. Switch to a text console by pressing Ctrl–Alt–F1.
2. Log in with your user name.
3. Move the user's GNOME configuration directories to a temporary location:

```
tux > mv .gconf  .gconf-ORIG-RECOVER
tux > mv .gnome2 .gnome2-ORIG-RECOVER
```

4. Log out.
5. Log in again, but do not run any applications.

6. Recover your individual application configuration data (including the Evolution e-mail client data) by copying the `~/.gconf-ORIG-RECOVER/apps/` directory back into the new `~/.gconf` directory as follows:

```
tux > cp -a .gconf-ORIG-RECOVER/apps .gconf/
```

If this causes the login problems, attempt to recover only the critical application data and reconfigure the remainder of the applications.

17.4 Network Problems

Many problems of your system may be network-related, even though they do not seem to be at first. For example, the reason for a system not allowing users to log in may be a network problem of some kind. This section introduces a simple checklist you can apply to identify the cause of any network problem encountered.

PROCEDURE 17.1: HOW TO IDENTIFY NETWORK PROBLEMS

When checking the network connection of your machine, proceed as follows:

1. If you use an Ethernet connection, check the hardware first. Make sure that your network cable is properly plugged into your computer and router (or hub, etc.). The control lights next to your Ethernet connector are normally both be active.
   If the connection fails, check whether your network cable works with another machine. If it does, your network card causes the failure. If hubs or switches are included in your network setup, they may be faulty, as well.

2. If using a wireless connection, check whether the wireless link can be established by other machines. If not, contact the wireless network's administrator.

3. When you have checked your basic network connectivity, try to find out which service is not responding. Gather the address information of all network servers needed in your setup. Either look them up in the appropriate YaST module or ask your system administrator. The following list gives some typical network servers involved in a setup together with the symptoms of an outage.

   DNS (Name Service)

A broken or malfunctioning name service affects the network's functionality in many ways. If the local machine relies on any network servers for authentication and these servers cannot be found because of name resolution issues, users would not even be able to log in. Machines in the network managed by a broken name server would not be able to “see” each other and communicate.

NTP (Time Service)

A malfunctioning or completely broken NTP service could affect Kerberos authentication and X server functionality.

NFS (File Service)

If any application needs data stored in an NFS mounted directory, it cannot start or function properly if this service was down or misconfigured. In the worst case scenario, a user's personal desktop configuration would not come up if their home directory containing the `.gconf` subdirectory could not be found because of a faulty NFS server.

Samba (File Service)

If any application needs data stored in a directory on a faulty Samba server, it cannot start or function properly.

NIS (User Management)

If your openSUSE Leap system relies on a faulty NIS server to provide the user data, users cannot log in to this machine.

LDAP (User Management)

If your openSUSE Leap system relies on a faulty LDAP server to provide the user data, users cannot log in to this machine.

Kerberos (Authentication)

Authentication will not work and login to any machine fails.

CUPS (Network Printing)

Users cannot print.

4. Check whether the network servers are running and whether your network setup allows you to establish a connection:

Important: Limitations

The debugging procedure described below only applies to a simple network server/client setup that does not involve any internal routing. It assumes both server and client are members of the same subnet without the need for additional routing.

a. Use **`ping`** *`IP_ADDRESS/HOSTNAME`* (replace with the host name or IP address of the server) to check whether each one of them is up and responding to the network. If this command is successful, it tells you that the host you were looking for is up and running and that the name service for your network is configured correctly.
If ping fails with `destination host unreachable`, either your system or the desired server is not properly configured or down. Check whether your system is reachable by running **`ping`** *`IP address`* or *`YOUR_HOSTNAME`* from another machine. If you can reach your machine from another machine, it is the server that is not running or not configured correctly.
If ping fails with `unknown host`, the name service is not configured correctly or the host name used was incorrect. For further checks on this matter, refer to *Step 4.b*. If ping still fails, either your network card is not configured correctly or your network hardware is faulty.

b. Use **`host`** *`HOSTNAME`* to check whether the host name of the server you are trying to connect to is properly translated into an IP address and vice versa. If this command returns the IP address of this host, the name service is up and running. If the **`host`** command fails, check all network configuration files relating to name and address resolution on your host:

`/var/run/netconfig/resolv.conf`

This file is used to keep track of the name server and domain you are currently using. It is a symbolic link to `/run/netconfig/resolv.conf` and is usually automatically adjusted by YaST or DHCP. Make sure that this file has the following structure and all network addresses and domain names are correct:

```
search FULLY_QUALIFIED_DOMAIN_NAME
nameserver IPADDRESS_OF_NAMESERVER
```

This file can contain more than one name server address, but at least one of them must be correct to provide name resolution to your host. If needed, adjust this file using the YaST Network Settings module (Hostname/DNS tab).
If your network connection is handled via DHCP, enable DHCP to change host name and name service information by selecting *Set Hostname via DHCP* (can be set globally for any interface or per interface) and *Update Name Servers and Search List via DHCP* in the YaST Network Settings module (Hostname/DNS tab).

`/etc/nsswitch.conf`

This file tells Linux where to look for name service information. It should look like this:

```
 ...
hosts: files dns
networks: files dns
...
```

The `dns` entry is vital. It tells Linux to use an external name server. Normally, these entries are automatically managed by YaST, but it would be prudent to check.
If all the relevant entries on the host are correct, let your system administrator check the DNS server configuration for the correct zone information. For detailed information about DNS, refer to *Book "Reference", Chapter 19 "The Domain Name System"*. If you have made sure that the DNS configuration of your host and the DNS server are correct, proceed with checking the configuration of your network and network device.

c. If your system cannot establish a connection to a network server and you have excluded name service problems from the list of possible culprits, check the configuration of your network card.
Use the command **`ip addr show`** *`NETWORK_DEVICE`* to check whether this device was properly configured. Make sure that the `inet address` with the netmask (*`/MASK`*) is configured correctly. An error in the IP address or a missing bit in your network mask would render your network configuration unusable. If necessary, perform this check on the server as well.

d. If the name service and network hardware are properly configured and running, but some external network connections still get long time-outs or fail entirely, use **`traceroute`** *`FULLY_QUALIFIED_DOMAIN_NAME`* (executed as `root`) to track the network route these requests are taking. This command lists any gateway (hop) that a request from your machine passes on its way to its destination. It lists the response time of each hop and whether this hop is reachable. Use a combination of traceroute and ping to track down the culprit and let the administrators know.

When you have identified the cause of your network trouble, you can resolve it yourself (if the problem is located on your machine) or let the system administrators of your network know about your findings so they can reconfigure the services or repair the necessary systems.

17.4.1 NetworkManager Problems

If you have a problem with network connectivity, narrow it down as described in *Procedure 17.1, "How to Identify Network Problems"*. If NetworkManager seems to be the culprit, proceed as follows to get logs providing hints on why NetworkManager fails:

1. Open a shell and log in as `root`.
2. Restart the NetworkManager:

```
tux > sudo systemctl restart NetworkManager
```

3. Open a Web page, for example, http://www.opensuse.org as normal user to see, if you can connect.
4. Collect any information about the state of NetworkManager in `/var/log/NetworkManager`.

For more information about NetworkManager, refer to *Book "Reference", Chapter 28 "Using NetworkManager"*.

17.5 Data Problems

Data problems are when the machine may or may not boot properly but, in either case, it is clear that there is data corruption on the system and that the system needs to be recovered. These situations call for a backup of your critical data, enabling you to recover the system state from before your system failed.

17.5.1 Managing Partition Images

Sometimes you need to perform a backup from an entire partition or even hard disk. Linux comes with the **dd** tool which can create an exact copy of your disk. Combined with **gzip** you save some space.

PROCEDURE 17.2: BACKING UP AND RESTORING HARD DISKS

1. Start a Shell as user `root`.
2. Select your source device. Typically this is something like `/dev/sda` (labeled as *SOURCE*).
3. Decide where you want to store your image (labeled as *BACKUP_PATH*). It must be different from your source device. In other words: if you make a backup from `/dev/sda`, your image file must not to be stored under `/dev/sda`.
4. Run the commands to create a compressed image file:

   ```
   root # dd if=/dev/SOURCE | gzip > /BACKUP_PATH/image.gz
   ```

5. Restore the hard disk with the following commands:

   ```
   root # gzip -dc /BACKUP_PATH/image.gz | dd of=/dev/SOURCE
   ```

If you only need to back up a partition, replace the *SOURCE* placeholder with your respective partition. In this case, your image file can lie on the same hard disk, but on a different partition.

17.5.2 Using the Rescue System

There are several reasons a system could fail to come up and run properly. A corrupted file system following a system crash, corrupted configuration files, or a corrupted boot loader configuration are the most common ones.

To help you to resolve these situations, openSUSE Leap contains a rescue system that you can boot. The rescue system is a small Linux system that can be loaded into a RAM disk and mounted as root file system, allowing you to access your Linux partitions from the outside. Using the rescue system, you can recover or modify any important aspect of your system.

- Manipulate any type of configuration file.
- Check the file system for defects and start automatic repair processes.
- Access the installed system in a "change root" environment.
- Check, modify, and re-install the boot loader configuration.
- Recover from a badly installed device driver or unusable kernel.
- Resize partitions using the parted command. Find more information about this tool at the GNU Parted Web site http://www.gnu.org/software/parted/parted.html.

The rescue system can be loaded from various sources and locations. The simplest option is to boot the rescue system from the original installation medium.

1. Insert the installation medium into your DVD drive.
2. Reboot the system.
3. At the boot screen, press F4 and choose *DVD-ROM*. Then choose *Rescue System* from the main menu.
4. Enter `root` at the `Rescue:` prompt. A password is not required.

If your hardware setup does not include a DVD drive, you can boot the rescue system from a network source. The following example applies to a remote boot scenario—if using another boot medium, such as a DVD, modify the `info` file accordingly and boot as you would for a normal installation.

1. Enter the configuration of your PXE boot setup and add the lines `install=`*`PROTOCOL`*`://`*`INSTSOURCE`* and `rescue=1`. If you need to start the repair system, use `repair=1` instead. As with a normal installation, *`PROTOCOL`* stands for any of the supported network protocols (NFS, HTTP, FTP, etc.) and *`INSTSOURCE`* for the path to your network installation source.
2. Boot the system using "Wake on LAN".

3. Enter `root` at the `Rescue:` prompt. A password is not required.

When you have entered the rescue system, you can use the virtual consoles that can be reached with Alt–F1 to Alt–F6.

A shell and other useful utilities, such as the mount program, are available in the `/bin` directory. The `/sbin` directory contains important file and network utilities for reviewing and repairing the file system. This directory also contains the most important binaries for system maintenance, such as **`fdisk`**, **`mkfs`**, **`mkswap`**, **`mount`**, and **`shutdown`**, **`ip`** and **`ss`** for maintaining the network. The directory `/usr/bin` contains the vi editor, find, less, and SSH.

To see the system messages, either use the command **`dmesg`** or view the system log with **`journalctl`**.

17.5.2.1 Checking and Manipulating Configuration Files

As an example for a configuration that might be fixed using the rescue system, imagine you have a broken configuration file that prevents the system from booting properly. You can fix this using the rescue system.

To manipulate a configuration file, proceed as follows:

1. Start the rescue system using one of the methods described above.
2. To mount a root file system located under `/dev/sda6` to the rescue system, use the following command:

   ```
   tux > sudo mount /dev/sda6 /mnt
   ```

   All directories of the system are now located under `/mnt`
3. Change the directory to the mounted root file system:

   ```
   tux > sudo cd /mnt
   ```
4. Open the problematic configuration file in the vi editor. Adjust and save the configuration.
5. Unmount the root file system from the rescue system:

   ```
   tux > sudo umount /mnt
   ```
6. Reboot the machine.

17.5.2.2 Repairing and Checking File Systems

Generally, file systems cannot be repaired on a running system. If you encounter serious problems, you may not even be able to mount your root file system and the system boot may end with a "kernel panic". In this case, the only way is to repair the system from the outside. The system contains the utilities to check and repair the `btrfs`, `ext2`, `ext3`, `ext4`, `xfs`, `dosfs`, and `vfat` file systems. Look for the command **`fsck.`***`FILESYSTEM`*. For example, if you need a file system check for `btrfs`, use **`fsck.btrfs`**.

17.5.2.3 Accessing the Installed System

If you need to access the installed system from the rescue system, you need to do this in a *change root* environment. For example, to modify the boot loader configuration, or to execute a hardware configuration utility.

To set up a change root environment based on the installed system, proceed as follows:

1. 

   ### Tip: Import LVM Volume Groups

   If you are using an LVM setup (refer to *Book "*Reference*", Chapter 5 "Expert Partitioner", Section 5.2 "LVM Configuration"* for more general details), import all existing volume groups to be able to find and mount the device(s):

   ```
   rootvgimport -a
   ```

   Run **`lsblk`** to check which node corresponds to the root partition. It is `/dev/sda2` in our example:

   ```
   tux > lsblk
   NAME        MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
   sda           8:0    0 149,1G  0 disk
   ├─sda1        8:1    0     2G  0 part  [SWAP]
   ├─sda2        8:2    0    20G  0 part  /
   └─sda3        8:3    0   127G  0 part
     └─cr_home 254:0    0   127G  0 crypt /home
   ```

2. Mount the root partition from the installed system:

   ```
   tux > sudo mount /dev/sda2 /mnt
   ```

3. Mount `/proc`, `/dev`, and `/sys` partitions:

```
tux > sudo mount -t proc none /mnt/proc
tux > sudo mount --rbind /dev /mnt/dev
tux > sudo mount --rbind /sys /mnt/sys
```

4. Now you can "change root" into the new environment, keeping the `bash` shell:

```
tux > chroot /mnt /bin/bash
```

5. Finally, mount the remaining partitions from the installed system:

```
tux > mount -a
```

6. Now you have access to the installed system. Before rebooting the system, unmount the partitions with **`umount -a`** and leave the "change root" environment with **`exit`**.

Warning: Limitations

Although you have full access to the files and applications of the installed system, there are some limitations. The kernel that is running is the one that was booted with the rescue system, not with the change root environment. It only supports essential hardware and it is not possible to add kernel modules from the installed system unless the kernel versions are identical. Always check the version of the currently running (rescue) kernel with **`uname -r`** and then find out if a matching subdirectory exists in the `/lib/modules` directory in the change root environment. If yes, you can use the installed modules, otherwise you need to supply their correct versions on other media, such as a flash disk. Most often the rescue kernel version differs from the installed one — then you cannot simply access a sound card, for example. It is also not possible to start a graphical user interface.

Also note that you leave the "change root" environment when you switch the console with Alt–F1 to Alt–F6.

17.5.2.4 Modifying and Re-installing the Boot Loader

Sometimes a system cannot boot because the boot loader configuration is corrupted. The start-up routines cannot, for example, translate physical drives to the actual locations in the Linux file system without a working boot loader.

To check the boot loader configuration and re-install the boot loader, proceed as follows:

1. Perform the necessary steps to access the installed system as described in *Section 17.5.2.3, "Accessing the Installed System"*.
2. Check that the GRUB 2 boot loader is installed on the system. If not, install the package `grub2` and run

   ```
   tux > sudo grub2-install /dev/sda
   ```
3. Check whether the following files are correctly configured according to the GRUB 2 configuration principles outlined in *Book "Reference", Chapter 12 "The Boot Loader GRUB 2"* and apply fixes if necessary.

   - `/etc/default/grub`
   - `/boot/grub2/device.map` (optional file, only present if created manually)
   - `/boot/grub2/grub.cfg` (this file is generated, do not edit)
   - `/etc/sysconfig/bootloader`
4. Re-install the boot loader using the following command sequence:

   ```
   tux > sudo grub2-mkconfig -o /boot/grub2/grub.cfg
   ```
5. Unmount the partitions, log out from the "change root" environment, and reboot the system:

   ```
   tux > umount -a
   exit
   reboot
   ```

17.5.2.5 Fixing Kernel Installation

A kernel update may introduce a new bug which can impact the operation of your system. For example a driver for a piece of hardware in your system may be faulty, which prevents you from accessing and using it. In this case, revert to the last working kernel (if available on the system) or install the original kernel from the installation media.

Tip: How to Keep Last Kernels after Update

To prevent failures to boot after a faulty kernel update, use the kernel multiversion feature and tell `libzypp` which kernels you want to keep after the update.

For example to always keep the last two kernels and the currently running one, add

```
multiversion.kernels = latest,latest-1,running
```

to the `/etc/zypp/zypp.conf` file. See *Book "Reference", Chapter 6 "Installing Multiple Kernel Versions"* for more information.

A similar case is when you need to re-install or update a broken driver for a device not supported by openSUSE Leap. For example when a hardware vendor uses a specific device, such as a hardware RAID controller, which needs a binary driver to be recognized by the operating system. The vendor typically releases a Driver Update Disk (DUD) with the fixed or updated version of the required driver.

In both cases you need to access the installed system in the rescue mode and fix the kernel related problem, otherwise the system may fail to boot correctly:

1. Boot from the openSUSE Leap installation media.
2. If you are recovering after a faulty kernel update, skip this step. If you need to use a driver update disk (DUD), press `F6` to load the driver update after the boot menu appears, and choose the path or URL to the driver update and confirm with *Yes*.
3. Choose *Rescue System* from the boot menu and press `Enter`. If you chose to use DUD, you will be asked to specify where the driver update is stored.
4. Enter `root` at the `Rescue:` prompt. A password is not required.
5. Manually mount the target system and "change root" into the new environment. For more information, see *Section 17.5.2.3, "Accessing the Installed System"*.
6. If using DUD, install/re-install/update the faulty device driver package. Always make sure the installed kernel version exactly matches the version of the driver you are installing.
   If fixing faulty kernel update installation, you can install the original kernel from the installation media with the following procedure.
   a. Identify your DVD device with **`hwinfo --cdrom`** and mount it with **`mount /dev/sr0 /mnt`**.

b. Navigate to the directory where your kernel files are stored on the DVD, for example `cd /mnt/suse/x86_64/`.

c. Install required `kernel-*`, `kernel-*-base`, and `kernel-*-extra` packages of your flavor with the `rpm -i` command.

7. Update configuration files and reinitialize the boot loader if needed. For more information, see *Section 17.5.2.4, "Modifying and Re-installing the Boot Loader"*.

8. Remove any bootable media from the system drive and reboot.

A GNU Licenses

This appendix contains the GNU Free Documentation License version 1.2.

GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or 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.

Security Guide

openSUSE Leap 15.1

Security Guide

openSUSE Leap 15.1

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.

Publication Date: May 25, 2019

SUSE LLC
10 Canal Park Drive
Suite 200
Cambridge MA 02141
USA
https://www.suse.com/documentation

Contents

About This Guide

This manual introduces the basic concepts of system security on openSUSE Leap. It covers extensive documentation about the authentication mechanisms available on Linux, such as NIS or LDAP. It deals with aspects of local security like access control lists, encryption and intrusion detection. In the network security part you learn how to secure computers with firewalls and masquerading, and how to set up virtual private networks (VPN). This manual shows how to use security software like AppArmor (which lets you specify per program which files the program may read, write, and execute) or the auditing system that collects information about security-relevant events.

1 Available Documentation

Note: Online Documentation and Latest Updates

Documentation for our products is available at http://doc.opensuse.org/, where you can also find the latest updates, and browse or download the documentation in various formats.

In addition, the product documentation is usually available in your installed system under `/usr/share/doc/manual`.

The following documentation is available for this product:

***Book* "Start-Up"**

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

***Book* "Reference"**

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

***Book* "Virtualization Guide"**

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

***Book* "AutoYaST Guide"**

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

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.

2 Feedback

Several feedback channels are available:

Bug Reports

To report bugs for openSUSE Leap, go to https://bugzilla.opensuse.org/, log in, and click *New*.

Mail

For feedback on the documentation of this product, you can also send a mail to `doc-team@suse.com`. 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 notices and typographical conventions are used in this documentation:

- `/etc/passwd`: directory names and file names
- *`PLACEHOLDER`*: replace *`PLACEHOLDER`* with the actual value
- `PATH`: the environment variable PATH
- **`ls`**, `--help`: commands, options, and parameters
- `user`: users or groups
- `package name` : name of a package
- Alt, Alt–F1: a key to press or a key combination; keys are shown in uppercase as on a keyboard
- *File*, *File* › *Save As*: menu items, buttons
- *Dancing Penguins* (Chapter *Penguins*, ↑Another Manual): This is a reference to a chapter in another manual.
- Commands that must be run with `root` privileges. Often you can also prefix these commands with the **`sudo`** command to run them as non-privileged user.

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

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

```
tux > command
```

- Notices

Warning: Warning Notice

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

Important: Important Notice

Important information you should be aware of before proceeding.

Note: Note Notice

Additional information, for example about differences in software versions.

Tip: Tip Notice

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

1 Security and Confidentiality

This chapter introduces basic concepts of computer security. Threats and basic mitigation techniques are described. The chapter also provides references to other chapters, guides and websites with further information.

1.1 Overview

One main characteristic of Linux is its ability to handle multiple users at the same time (multiuser) and to allow these users to simultaneously perform tasks (multitasking) on the same computer. To users, there is no difference between working with data stored locally and data stored in the network.

Due to the multiuser capability, data from different users has to be stored separately to guarantee security and privacy. Also important is the ability to keep data available in spite of a lost or damaged data medium, for example a hard disk.

This chapter is primarily focused on confidentiality and privacy. But a comprehensive security concept includes a regularly updated, workable, and tested backup. Without a backup, restoring data after it has been tampered with or after a hardware failure is very hard.

Use a *defense-in-depth* approach to security: Assume that no single threat mitigation can fully protect your systems and data, but multiple layers of defense will make an attack much harder. Components of a defense-in-depth strategy can be the following:

- Hashing passwords (for example with PBKDF2, bcrypt, or scrypt) and salting them
- Encrypting data (for example with AES)
- Logging, monitoring, and intrusion detection
- Firewall
- Antivirus scanner
- Defined and documented emergency procedures
- Backups
- Physical security
- Audits, security scans, and intrusion tests

openSUSE Leap includes software that addresses the requirements of the list above. The following sections provide starting points for securing your system.

1.2 Passwords

On a Linux system, only hashes of passwords are stored. Hashes are one-way algorithms that make it easy to encrypt data. At the same time, hash algorithms make it very hard to compute the original secret from the hash.

The hashes are stored in the file `/etc/shadow`, which cannot be read by normal users. Because restoring passwords is possible with powerful computers, hashed passwords should not be visible to regular users.

The *National Institute of Standards and Technology* (*NIST*) publishes a guideline for passwords, which is available at https://pages.nist.gov/800-63-3/sp800-63b.html#sec5

For details about how to set a password policy, see *Section 8.3, "Password Settings"*. For general information about authentication on Linux, see *Part I, "Authentication"*.

1.3 System Integrity

If it is possible to physically access a computer, the firmware and boot process can be manipulated to gain access as soon as an authorized person boots the machine. While not all computers can be locked into inaccessible rooms, your first step should be physically locking the server room.

Consider taking the following additional measures:

- Configure your system so it cannot be booted from a removable device, either by removing the drives entirely or by setting a UEFI password and configuring the UEFI to allow booting from a hard disk only.
- To make the boot procedure more tamper-resistant, enable the UEFI *secure boot* feature. For more information about Secure Boot, see *Book "Reference", Chapter 14 "UEFI (Unified Extensible Firmware Interface)"*.

- Linux systems are started by a boot loader that usually allows passing additional options to the booted kernel. You can prevent others from using such parameters during boot by setting an additional password for the boot loader. This is crucial to system security. Not only does the kernel itself run with `root` permissions, but it is also the first authority to grant `root` permissions at system start-up.
  For more information about setting a password in the boot loader, see *Book "Reference", Chapter 12 "The Boot Loader GRUB 2", Section 12.2.6 "Setting a Boot Password"*.

- Enable hard disk encryption. For more information, see *Chapter 11, Encrypting Partitions and Files*.

- Use AIDE to detect any changes in your system configuration. For more information, see *Chapter 13, Intrusion Detection with AIDE*.

1.4 File Access

Because of the *everything is a file* approach in Linux, file permissions are important for controlling access to most resources. This means that by using file permissions, you can define access to regular files and directories as well as hardware devices. By default, most hardware devices are only accessible for `root`. However, some devices, for example serial ports, can be accessible for normal users.

As a general rule, always work with the most restrictive privileges possible for a given task. For example, it is definitely not necessary to be `root` to read or write e-mail. If the mail program has a bug, this bug could be exploited for an attack that acts with exactly the permissions of the program at the time of the attack. By following the above rule, minimize the possible damage.

For details, see *Section 10.1, "Traditional File Permissions"* and *Section 10.2, "Advantages of ACLs"*.

AppArmor and SELinux allow you to set constraints for applications and users. For details, see *Part IV, "Confining Privileges with AppArmor"* and *Part V, "SELinux"*.

If there is a chance that hard disks could be accessed outside of the installed operating system, for example by booting a live system or removing the hardware, encrypt the data. openSUSE Leap allows you to encrypt partitions containing data and the operating system. For details, see *Chapter 11, Encrypting Partitions and Files*.

1.5 Networking

Securing network services is a crucial task. Aim to secure as many layers of the *OSI model* as possible.

All communication should be authenticated and encrypted with up-to-date cryptographic algorithms on the transport or application layer. Use a Virtual Private Network (VPN) as an additional secure layer on physical networks.

openSUSE Leap provides many options for securing your network:

- Use `openssl` to create X509 certificates. These certificates can be used for encryption and authentication of many services. You can set up your own *certificate authority* (*CA*) and use it as a source of trust in your network. For details, see `man openssl`.
- Usually, at least parts of networks are exposed to the public Internet. Reduce attack surfaces by closing ports with firewall rules and by uninstalling or at least disabling unrequired services. For details, see *Chapter 16, Masquerading and Firewalls*.
- Use OpenVPN to secure communication channels over insecure physical networks. For details, see *Chapter 17, Configuring a VPN Server*.
- Use strong authentication for network services. For details, see *Part I, "Authentication"*.

1.6 Software Vulnerabilities

Software vulnerabilities are issues in software that can be exploited to obtain unauthorized access or misuse systems. Vulnerabilities are especially critical if they affect remote services, such as HTTP servers. Computer systems are very complex, therefore they always include certain vulnerabilities.

When such issues become known, they must usually be fixed in the software by software developers. The resulting update must then be installed by system administrators in a timely and safe manner on affected systems.

Vulnerabilities are usually announced on centralized databases, for example the *National Vulnerability Database*, which is maintained by the US government. You can subscribe to feeds to stay informed about newly discovered vulnerabilities. In some cases the problems induced by the bugs can be mitigated until a software update is provided. Vulnerabilities are assigned a *Common Vulnerabilities and Exposures* (*CVE*) number and a *Common Vulnerability Scoring System* (*CVSS*) score. The score helps identify the severity of vulnerabilities.

SUSE provides a feed of security advisories. It is available at https://www.suse.com/en-us/support/update/. There is also a list of security updates by CVE number available at https://www.suse.com/en-us/security/cve/.

In general, administrators should be prepared for severe vulnerabilities in their systems. This includes hardening all computers as far as possible. Also, we recommend to have predefined procedures in place for quickly installing updates for severe vulnerabilities.

To reduce the damage of possible attacks, use restrictive file permissions. See *Section 10.1, "Traditional File Permissions"*. SUSE provides a guide to hardening openSUSE Leap.

Other useful links:

- http://lists.opensuse.org/opensuse-security-announce/, mailing list with openSUSE security announcements
- https://nvd.nist.gov/home, the National Vulnerability Database
- https://cve.mitre.org/, MITRE's CVE database
- https://www.bsi.bund.de/DE/Service/Aktuell/Cert_Bund_Meldungen/cert_bund_meldungen_node.html, *German Federal Office for Information Security* vulnerability feed
- https://www.first.org/cvss/, information about the Common Vulnerability Scoring System

1.7 Malware

Malware is software that is intended to interrupt the normal functioning of a computer or steal data. This includes viruses, worms, ransomware, or rootkits. Sometimes malware uses software vulnerabilities to attack a computer. However, in many cases it is accidentally executed by a user, especially when installing third-party software from unknown sources. openSUSE Leap provides an extensive list of programs (packages) in its download repositories. This reduces the need to download third-party software. All packages provided by SUSE are signed. The package manager of openSUSE Leap checks the signatures of packages after the download to verify their integrity.

The command `rpm --checksig RPM_FILE` shows whether the checksum and the signature of a package are correct. You can find the signing key on the first DVD of openSUSE Leap and on most key servers worldwide.

You can use the ClamAV antivirus software to detect malware on your system. ClamAV can be integrated into several services, for example mail servers and HTTP proxies. This can be used to filter malware before it reaches the user.

Restrictive user privileges can reduce the risk of accidental code execution.

1.8 Important Security Tips

The following tips are a quick summary of the sections above:

- Stay informed about the latest security issues. Get and install the updated packages recommended by security announcements as quickly as possible.
- Avoid using `root` privileges whenever possible. Set restrictive file permissions.
- Only use encrypted protocols for network communication.
- Disable any network services you do not absolutely require.
- Conduct regular security audits. For example, scan your network for open ports.
- Monitor the integrity of files on your systems with `AIDE` (Advanced Intrusion Detection Environment).
- Take proper care when installing any third-party software.
- Check all your backups regularly.
- Check your log files, for example with logwatch.
- Configure the firewall to block all ports that are not explicitly whitelisted.
- Design your security measures to be redundant.
- Use encryption where possible, for example for hard disks of mobile computers.

1.9 Reporting Security Issues

If you discover a security-related problem, first check the available update packages. If no update is available, write an e-mail to `security@suse.de`. Include a detailed description of the problem and the version number of the package concerned. We encourage you to encrypt e-mails with GPG.

You can find a current version of the SUSE GPG key at https://www.suse.com/support/security/contact/.

I Authentication

2 Authentication with PAM

Linux uses PAM (pluggable authentication modules) in the authentication process as a layer that mediates between user and application. PAM modules are available on a system wide basis, so they can be requested by any application. This chapter describes how the modular authentication mechanism works and how it is configured.

2.1 What is PAM?

System administrators and programmers often want to restrict access to certain parts of the system or to limit the use of certain functions of an application. Without PAM, applications must be adapted every time a new authentication mechanism, such as LDAP, Samba, or Kerberos, is introduced. However, this process is time-consuming and error-prone. One way to avoid these drawbacks is to separate applications from the authentication mechanism and delegate authentication to centrally managed modules. Whenever a newly required authentication scheme is needed, it is sufficient to adapt or write a suitable *PAM module* for use by the program in question.

The PAM concept consists of:

- *PAM modules*, which are a set of shared libraries for a specific authentication mechanism.
- A *module stack* with of one or more PAM modules.
- A PAM-aware *service* which needs authentication by using a module stack or PAM modules. Usually a service is a familiar name of the corresponding application, like **`login`** or **`su`**. The service name `other` is a reserved word for default rules.
- *Module arguments*, with which the execution of a single PAM module can be influenced.
- A mechanism evaluating each *result* of a single PAM module execution. A positive value executes the next PAM module. The way a negative value is dealt with depends on the configuration: “no influence, proceed” up to “terminate immediately” and anything in between are valid options.

2.2 Structure of a PAM Configuration File

PAM can be configured in two ways:

File based configuration (`/etc/pam.conf`)

The configuration of each service is stored in `/etc/pam.conf`. However, for maintenance and usability reasons, this configuration scheme is not used in openSUSE Leap.

Directory based configuration (`/etc/pam.d/`)

Every service (or program) that relies on the PAM mechanism has its own configuration file in the `/etc/pam.d/` directory. For example, the service for `sshd` can be found in the `/etc/pam.d/sshd` file.

The files under `/etc/pam.d/` define the PAM modules used for authentication. Each file consists of lines, which define a service, and each line consists of a maximum of four components:

```
TYPE  CONTROL
 MODULE_PATH  MODULE_ARGS
```

The components have the following meaning:

TYPE

Declares the type of the service. PAM modules are processed as stacks. Different types of modules have different purposes. For example, one module checks the password, another verifies the location from which the system is accessed, and yet another reads user-specific settings. PAM knows about four different types of modules:

`auth`

Check the user's authenticity, traditionally by querying a password. However, this can also be achieved with a chip card or through biometrics (for example, fingerprints or iris scan).

`account`

Modules of this type check if the user has general permission to use the requested service. As an example, such a check should be performed to ensure that no one can log in with the user name of an expired account.

`password`

The purpose of this type of module is to enable the change of an authentication token. Usually this is a password.

`session`

Modules of this type are responsible for managing and configuring user sessions. They are started before and after authentication to log login attempts and configure the user's specific environment (mail accounts, home directory, system limits, etc.).

`CONTROL`

Indicates the behavior of a PAM module. Each module can have the following control flags:

`required`

A module with this flag must be successfully processed before the authentication may proceed. After the failure of a module with the `required` flag, all other modules with the same flag are processed before the user receives a message about the failure of the authentication attempt.

`requisite`

Modules having this flag must also be processed successfully, in much the same way as a module with the `required` flag. However, in case of failure a module with this flag gives immediate feedback to the user and no further modules are processed. In case of success, other modules are subsequently processed, like any modules with the `required` flag. The `requisite` flag can be used as a basic filter checking for the existence of certain conditions that are essential for a correct authentication.

`sufficient`

After a module with this flag has been successfully processed, the requesting application receives an immediate message about the success and no further modules are processed, provided there was no preceding failure of a module with the `required` flag. The failure of a module with the `sufficient` flag has no direct consequences, in the sense that any subsequent modules are processed in their respective order.

`optional`

The failure or success of a module with this flag does not have any direct consequences. This can be useful for modules that are only intended to display a message (for example, to tell the user that mail has arrived) without taking any further action.

`include`

If this flag is given, the file specified as argument is inserted at this place.

`MODULE_PATH`

Contains a full file name of a PAM module. It does not need to be specified explicitly, as long as the module is located in the default directory `/lib/security` (for all 64-bit platforms supported by openSUSE® Leap, the directory is `/lib64/security`).

`MODULE_ARGS`

Contains a space-separated list of options to influence the behavior of a PAM module, such as `debug` (enables debugging) or `nullok` (allows the use of empty passwords).

In addition, there are global configuration files for PAM modules under `/etc/security`, which define the exact behavior of these modules (examples include `pam_env.conf` and `time.conf`). Every application that uses a PAM module actually calls a set of PAM functions, which then process the information in the various configuration files and return the result to the requesting application.

To simplify the creation and maintenance of PAM modules, common default configuration files for the types `auth`, `account`, `password`, and `session` modules have been introduced. These are retrieved from every application's PAM configuration. Updates to the global PAM configuration modules in `common-*` are thus propagated across all PAM configuration files without requiring the administrator to update every single PAM configuration file.

The global PAM configuration files are maintained using the **`pam-config`** tool. This tool automatically adds new modules to the configuration, changes the configuration of existing ones or deletes modules (or options) from the configurations. Manual intervention in maintaining PAM configurations is minimized or no longer required.

Note: 64-Bit and 32-Bit Mixed Installations

When using a 64-bit operating system, it is possible to also include a runtime environment for 32-bit applications. In this case, make sure that you also install the 32-bit version of the PAM modules.

2.3 The PAM Configuration of sshd

Consider the PAM configuration of sshd as an example:

EXAMPLE 2.1: PAM CONFIGURATION FOR SSHD (`/etc/pam.d/sshd`)

```
#%PAM-1.0 ❶
```

```
auth     requisite      pam_nologin.so                          ❷
auth     include        common-auth                             ❸
account  requisite      pam_nologin.so                          ❷
account  include        common-account                          ❸
password include        common-password                         ❸
session  required       pam_loginuid.so                         ❹
session  include        common-session                          ❸
session  optional       pam_lastlog.so   silent noupdate showfailed ❺
```

❶ Declares the version of this configuration file for PAM 1.0. This is merely a convention, but could be used in the future to check the version.

❷ Checks, if `/etc/nologin` exists. If it does, no user other than `root` may log in.

❸ Refers to the configuration files of four module types: `common-auth`, `common-account`, `common-password`, and `common-session`. These four files hold the default configuration for each module type.

❹ Sets the login UID process attribute for the process that was authenticated.

❺ Displays information about the last login of a user.

By including the configuration files instead of adding each module separately to the respective PAM configuration, you automatically get an updated PAM configuration when an administrator changes the defaults. Formerly, you needed to adjust all configuration files manually for all applications when changes to PAM occurred or a new application was installed. Now the PAM configuration is made with central configuration files and all changes are automatically inherited by the PAM configuration of each service.

The first include file (`common-auth`) calls three modules of the `auth` type: `pam_env.so`, `pam_gnome_keyring.so` and `pam_unix.so`. See *Example 2.2, "Default Configuration for the `auth` Section (`common-auth`)"*.

EXAMPLE 2.2: DEFAULT CONFIGURATION FOR THE `auth` SECTION (`common-auth`)

```
auth  required  pam_env.so                  ❶
auth  optional  pam_gnome_keyring.so        ❷
auth  required  pam_unix.so  try_first_pass ❸
```

❶ `pam_env.so` loads `/etc/security/pam_env.conf` to set the environment variables as specified in this file. It can be used to set the `DISPLAY` variable to the correct value, because the `pam_env` module knows about the location from which the login is taking place.

❷ `pam_gnome_keyring.so` checks the user's login and password against the GNOME key ring

❸ `pam_unix` checks the user's login and password against `/etc/passwd` and `/etc/shadow`.

The whole stack of `auth` modules is processed before `sshd` gets any feedback about whether the login has succeeded. All modules of the stack having the `required` control flag must be processed successfully before `sshd` receives a message about the positive result. If one of the modules is not successful, the entire module stack is still processed and only then is `sshd` notified about the negative result.

When all modules of the `auth` type have been successfully processed, another include statement is processed, in this case, that in *Example 2.3, "Default Configuration for the `account` Section (`common-account`)"*. `common-account` contains only one module, `pam_unix`. If `pam_unix` returns the result that the user exists, sshd receives a message announcing this success and the next stack of modules (`password`) is processed, shown in *Example 2.4, "Default Configuration for the `password` Section (`common-password`)"*.

EXAMPLE 2.3: DEFAULT CONFIGURATION FOR THE `account` SECTION (`common-account`)

```
account  required  pam_unix.so  try_first_pass
```

EXAMPLE 2.4: DEFAULT CONFIGURATION FOR THE `password` SECTION (`common-password`)

```
password  requisite  pam_cracklib.so
password  optional   pam_gnome_keyring.so  use_authtok
password  required   pam_unix.so  use_authtok nullok shadow try_first_pass
```

Again, the PAM configuration of `sshd` involves only an include statement referring to the default configuration for `password` modules located in `common-password`. These modules must successfully be completed (control flags `requisite` and `required`) whenever the application requests the change of an authentication token.

Changing a password or another authentication token requires a security check. This is achieved with the `pam_cracklib` module. The `pam_unix` module used afterward carries over any old and new passwords from `pam_cracklib`, so the user does not need to authenticate again after changing the password. This procedure makes it impossible to circumvent the checks carried out by `pam_cracklib`. Whenever the `account` or the `auth` type are configured to complain about expired passwords, the `password` modules should also be used.

EXAMPLE 2.5: DEFAULT CONFIGURATION FOR THE `session` SECTION (`common-session`)

```
session  required  pam_limits.so
session  required  pam_unix.so  try_first_pass
```

```
session  optional  pam_umask.so
session  optional  pam_systemd.so
session  optional  pam_gnome_keyring.so auto_start only_if=gdm,gdm-password,lxdm,lightdm
session  optional  pam_env.so
```

As the final step, the modules of the `session` type (bundled in the `common-session` file) are called to configure the session according to the settings for the user in question. The `pam_limits` module loads the file `/etc/security/limits.conf`, which may define limits on the use of certain system resources. The `pam_unix` module is processed again. The `pam_umask` module can be used to set the file mode creation mask. Since this module carries the `optional` flag, a failure of this module would not affect the successful completion of the entire session module stack. The `session` modules are called a second time when the user logs out.

2.4 Configuration of PAM Modules

Some PAM modules are configurable. The configuration files are located in `/etc/security`. This section briefly describes the configuration files relevant to the sshd example — `pam_env.conf` and `limits.conf`.

2.4.1 pam_env.conf

`pam_env.conf` can be used to define a standardized environment for users that is set whenever the `pam_env` module is called. With it, preset environment variables using the following syntax:

```
VARIABLE  [DEFAULT=VALUE]  [OVERRIDE=VALUE]
```

`VARIABLE`
: Name of the environment variable to set.

`[DEFAULT=<value>]`
: Default *`VALUE`* the administrator wants to set.

`[OVERRIDE=<value>]`
: Values that may be queried and set by `pam_env`, overriding the default value.

A typical example of how `pam_env` can be used is the adaptation of the `DISPLAY` variable, which is changed whenever a remote login takes place. This is shown in *Example 2.6, "pam_env.conf"*.

EXAMPLE 2.6: PAM_ENV.CONF

```
REMOTEHOST  DEFAULT=localhost          OVERRIDE=@{PAM_RHOST}
DISPLAY     DEFAULT=${REMOTEHOST}:0.0  OVERRIDE=${DISPLAY}
```

The first line sets the value of the `REMOTEHOST` variable to `localhost`, which is used whenever `pam_env` cannot determine any other value. The `DISPLAY` variable in turn contains the value of `REMOTEHOST`. Find more information in the comments in `/etc/security/pam_env.conf`.

2.4.2 pam_mount.conf.xml

The purpose of `pam_mount` is to mount user home directories during the login process, and to unmount them during logout in an environment where a central file server keeps all the home directories of users. With this method, it is not necessary to mount a complete `/home` directory where all the user home directories would be accessible. Instead, only the home directory of the user who is about to log in, is mounted.

After installing `pam_mount`, a template for `pam_mount.conf.xml` is available in `/etc/security`. The description of the various elements can be found in the manual page **`man 5 pam_mount.conf`**.

A basic configuration of this feature can be done with YaST. Select *Network Settings* › *Windows Domain Membership* › *Expert Settings* to add the file server; see *Book "Reference", Chapter 21 "Samba", Section 21.5 "Configuring Clients"*.

2.4.3 limits.conf

System limits can be set on a user or group basis in `limits.conf`, which is read by the `pam_limits` module. The file allows you to set hard limits, which may not be exceeded, and soft limits, which may be exceeded temporarily. For more information about the syntax and the options, see the comments in `/etc/security/limits.conf`.

2.5 Configuring PAM Using pam-config

The **`pam-config`** tool helps you configure the global PAM configuration files (`/etc/pam.d/common-*`) and several selected application configurations. For a list of supported modules, use the **`pam-config --list-modules`** command. Use the **`pam-config`** command to maintain your

PAM configuration files. Add new modules to your PAM configurations, delete other modules or modify options to these modules. When changing global PAM configuration files, no manual tweaking of the PAM setup for individual applications is required.

A simple use case for **pam-config** involves the following:

1. **Auto-generate a fresh Unix-style PAM configuration.** Let pam-config create the simplest possible setup which you can extend later on. The **pam-config --create** command creates a simple Unix authentication configuration. Pre-existing configuration files not maintained by pam-config are overwritten, but backup copies are kept as `*.pam-config-backup`.

2. **Add a new authentication method.** Adding a new authentication method (for example, LDAP) to your stack of PAM modules comes down to a simple **pam-config --add --ldap** command. LDAP is added wherever appropriate across all `common-*-pc` PAM configuration files.

3. **Add debugging for test purposes.** To make sure the new authentication procedure works as planned, turn on debugging for all PAM-related operations. The **pam-config --add --ldap-debug** turns on debugging for LDAP-related PAM operations. Find the debugging output in the `systemd` journal (see *Book "Reference", Chapter 11 "**journalctl**: Query the `systemd` Journal"*).

4. **Query your setup.** Before you finally apply your new PAM setup, check if it contains all the options you wanted to add. The **pam-config --query --** *MODULE* lists both the type and the options for the queried PAM module.

5. **Remove the debug options.** Finally, remove the debug option from your setup when you are entirely satisfied with the performance of it. The **pam-config --delete --ldap-debug** command turns off debugging for LDAP authentication. In case you had debugging options added for other modules, use similar commands to turn these off.

For more information on the **pam-config** command and the options available, refer to the manual page of **pam-config(8)**.

2.6 Manually Configuring PAM

If you prefer to manually create or maintain your PAM configuration files, make sure to disable **pam-config** for these files.

When you create your PAM configuration files from scratch using the **pam-config --create** command, it creates symbolic links from the `common-*` to the `common-*-pc` files. **pam-config** only modifies the `common-*-pc` configuration files. Removing these symbolic links effectively disables pam-config, because pam-config only operates on the `common-*-pc` files and these files are not put into effect without the symbolic links.

Warning: Include `pam_systemd.so` in Configuration

If you are creating your own PAM configuration, make sure to include `pam_systemd.so` configured as `session optional`. Not including the `pam_systemd.so` can cause problems with `systemd` task limits. For details, refer to the man page of `pam_systemd.so`.

2.7 For More Information

In the `/usr/share/doc/packages/pam` directory after installing the `pam-doc` package, find the following additional documentation:

READMEs

In the top level of this directory, there is the `modules` subdirectory holding README files about the available PAM modules.

The Linux-PAM System Administrators' Guide

This document comprises everything that the system administrator should know about PAM. It discusses a range of topics, from the syntax of configuration files to the security aspects of PAM.

The Linux-PAM Module Writers' Manual

This document summarizes the topic from the developer's point of view, with information about how to write standard-compliant PAM modules.

The Linux-PAM Application Developers' Guide

This document comprises everything needed by an application developer who wants to use the PAM libraries.

The PAM Manual Pages

PAM in general and the individual modules come with manual pages that provide a good overview of the functionality of all the components.

3 Using NIS

When multiple Unix systems in a network access common resources, it becomes imperative that all user and group identities are the same for all machines in that network. The network should be transparent to users: their environments should not vary, regardless of which machine they are actually using. This can be done by means of NIS and NFS services. NFS distributes file systems over a network and is discussed in *Book "Reference", Chapter 22 "Sharing File Systems with NFS"*.

NIS (Network Information Service) can be described as a database-like service that provides access to the contents of `/etc/passwd`, `/etc/shadow`, and `/etc/group` across networks. NIS can also be used for other purposes (making the contents of files like `/etc/hosts` or `/etc/services` available, for example), but this is beyond the scope of this introduction. People often refer to NIS as *YP*, because it works like the network's "yellow pages."

3.1 Configuring NIS Servers

To distribute NIS information across networks, either install one single server (a *master*) that serves all clients, or NIS slave servers requesting this information from the master and relaying it to their respective clients.

- To configure just one NIS server for your network, proceed with *Section 3.1.1, "Configuring a NIS Master Server"*.
- If your NIS master server needs to export its data to slave servers, set up the master server as described in *Section 3.1.1, "Configuring a NIS Master Server"* and set up slave servers in the subnets as described in *Section 3.1.2, "Configuring a NIS Slave Server"*.

3.1.1 Configuring a NIS Master Server

To manage the NIS Server functionality with YaST, install the `yast2-nis-server` package by running the **`zypper in yast2-nis-server`** command as root. To configure a NIS master server for your network, proceed as follows:

1. Start *YaST* › *Network Services* › *NIS Server*.

2. If you need just one NIS server in your network or if this server is to act as the master for further NIS slave servers, select *Install and Set Up NIS Master Server*. YaST installs the required packages.

Tip: Already Installed NIS Server Software

If NIS server software is already installed on your machine, initiate the creation of a NIS master server by clicking *Create NIS Master Server*.

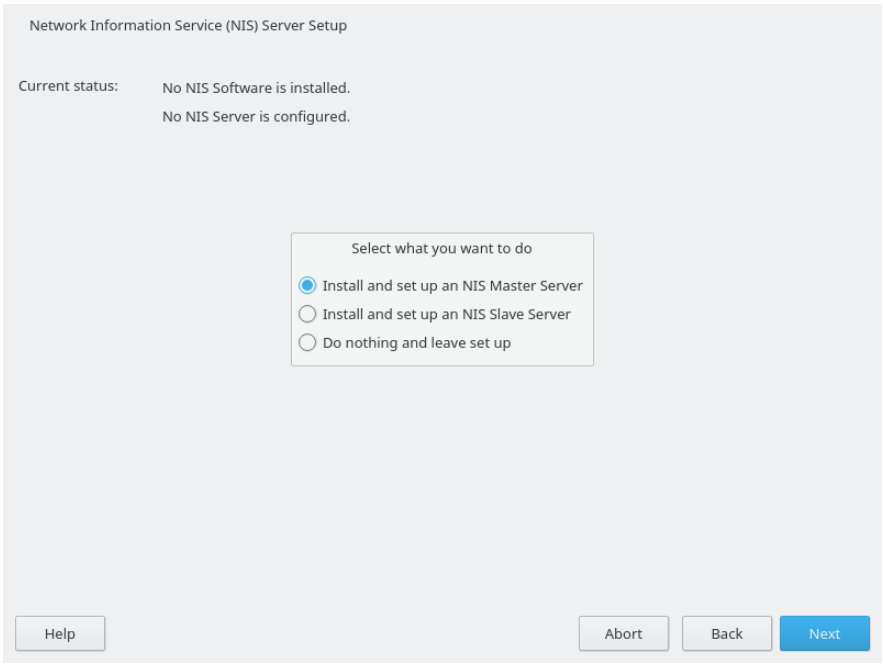


FIGURE 3.1: NIS SERVER SETUP

3. Determine basic NIS setup options:

   a. Enter the NIS domain name.

   b. Define whether the host should also be a NIS client (enabling users to log in and access data from the NIS server) by selecting *This Host is also a NIS Client*.

   c. If your NIS server needs to act as a master server to NIS slave servers in other subnets, select *Active Slave NIS Server Exists*.
   The option *Fast Map Distribution* is only useful with *Active Slave NIS Servers Exist*. It speeds up the transfer of maps to the slaves.

d. Select *Allow Changes to Passwords* to allow users in your network (both local users and those managed through the NIS server) to change their passwords on the NIS server (with the command **`yppasswd`**). This makes the options *Allow Changes to GECOS Field* and *Allow Changes to Login Shell* available. "GECOS" means that the users can also change their names and address settings with the command **`ypchfn`**. "Shell" allows users to change their default shell with the command **`ypchsh`** (for example, to switch from Bash to sh). The new shell must be one of the predefined entries in `/etc/shells`.

e. Select *Open Port in Firewall* to have YaST adapt the firewall settings for the NIS server.

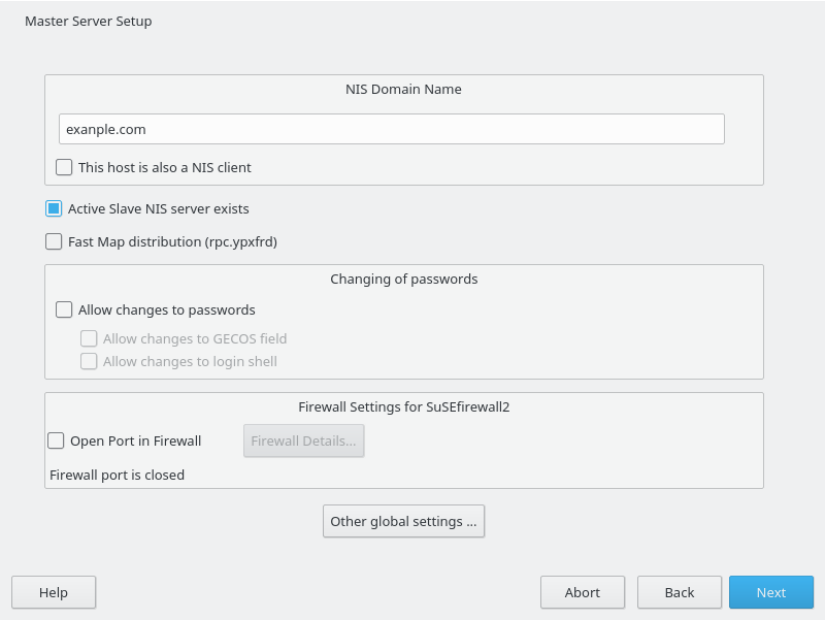


FIGURE 3.2: MASTER SERVER SETUP

f. Leave this dialog with *Next* or click *Other Global Settings* to make additional settings. *Other Global Settings* include changing the source directory of the NIS server (`/etc` by default). In addition, passwords can be merged here. The setting should be *Yes* to create the user database from the system authentication files `/etc/passwd`, `/etc/shadow`, and `/etc/group`. Also, determine the smallest user and group ID that should be offered by NIS. Click *OK* to confirm your settings and return to the previous screen.

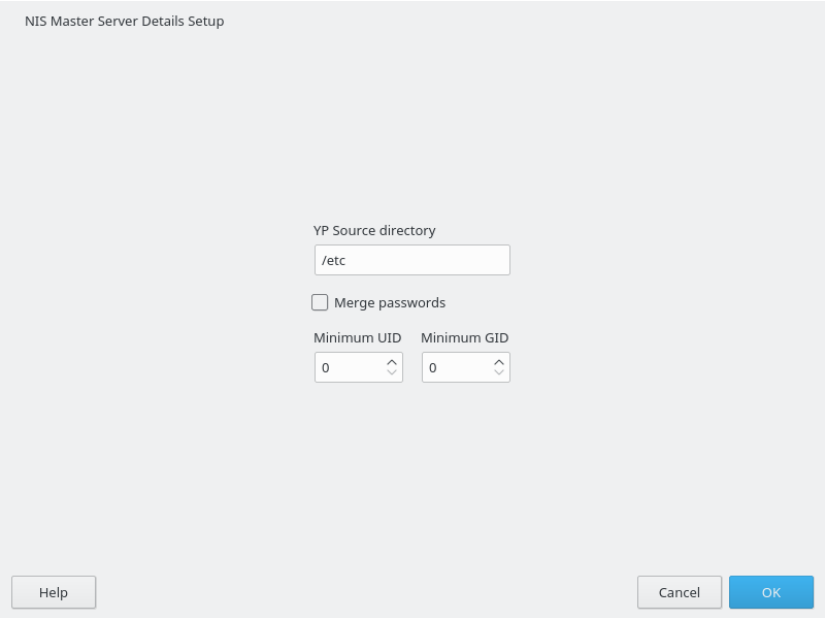


FIGURE 3.3: CHANGING THE DIRECTORY AND SYNCHRONIZING FILES FOR A NIS SERVER

4. If you previously enabled *Active Slave NIS Server Exists*, enter the host names used as slaves and click *Next*. If no slave servers exist, this configuration step is skipped.

5. Continue to the dialog for the database configuration. Specify the *NIS Server Maps*, the partial databases to transfer from the NIS server to the client. The default settings are usually adequate. Leave this dialog with *Next*.

6. Check which maps should be available and click *Next* to continue.

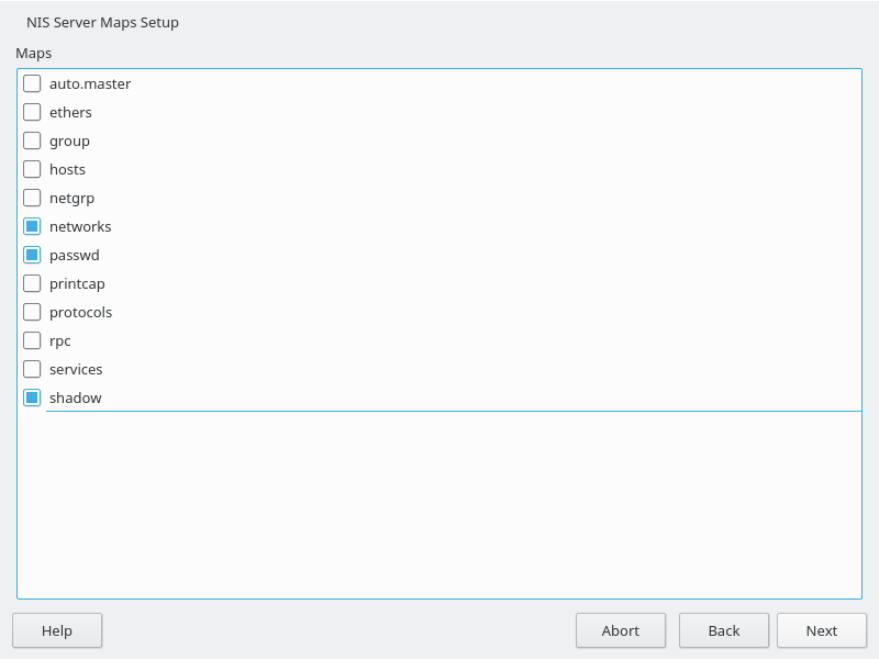


FIGURE 3.4: NIS SERVER MAPS SETUP

7. Determine which hosts are allowed to query the NIS server. You can add, edit, or delete hosts by clicking the appropriate button. Specify from which networks requests can be sent to the NIS server. Normally, this is your internal network. In this case, there should be the following two entries:

```
255.0.0.0     127.0.0.0
0.0.0.0       0.0.0.0
```

The first entry enables connections from your own host, which is the NIS server. The second one allows all hosts to send requests to the server.

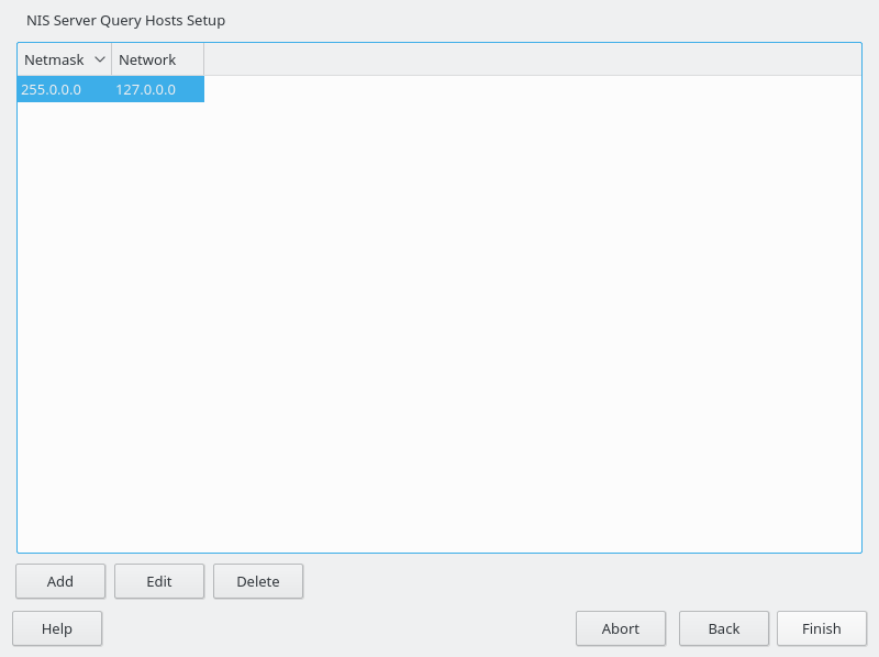


FIGURE 3.5: SETTING REQUEST PERMISSIONS FOR A NIS SERVER

8. Click *Finish* to save your changes and exit the setup.

3.1.2 Configuring a NIS Slave Server

To configure additional NIS *slave servers* in your network, proceed as follows:

1. Start *YaST* › *Network Services* › *NIS Server*.
2. Select *Install and Set Up NIS Slave Server* and click *Next*.

Tip

If NIS server software is already installed on your machine, initiate the creation of a NIS slave server by clicking *Create NIS Slave Server*.

3. Complete the basic setup of your NIS slave server:
   a. Enter the NIS domain.
   b. Enter host name or IP address of the master server.
   c. Set *This Host is also a NIS Client* if you want to enable user logins on this server.

d. Adapt the firewall settings with *Open Ports in Firewall*.

e. Click *Next*.

4. Enter the hosts that are allowed to query the NIS server. You can add, edit, or delete hosts by clicking the appropriate button. Specify all networks from which requests can be sent to the NIS server. If it applies to all networks, use the following configuration:

```
255.0.0.0     127.0.0.0
0.0.0.0       0.0.0.0
```

   The first entry enables connections from your own host, which is the NIS server. The second one allows all hosts with access to the same network to send requests to the server.

5. Click *Finish* to save changes and exit the setup.

3.2 Configuring NIS Clients

To use NIS on a workstation, do the following:

1. Start *YaST* › *Network Services* › *NIS Client*.

2. Activate the *Use NIS* button.

3. Enter the NIS domain. This is usually a domain name given by your administrator or a static IP address received by DHCP. For information about DHCP, see *Book "Reference", Chapter 20 "DHCP"*.

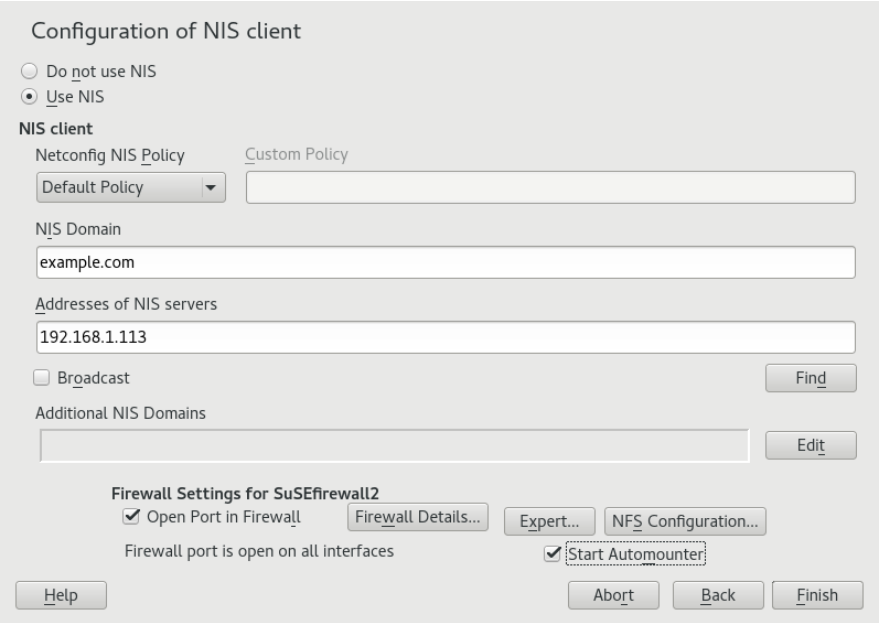


FIGURE 3.6: SETTING DOMAIN AND ADDRESS OF A NIS SERVER

4. Enter your NIS servers and separate their addresses by spaces. If you do not know your NIS server, click *Find* to let YaST search for any NIS servers in your domain. Depending on the size of your local network, this may be a time-consuming process. *Broadcast* asks for a NIS server in the local network after the specified servers fail to respond.

5. Depending on your local installation, you may also want to activate the automounter. This option also installs additional software if required.

6. If you do not want other hosts to be able to query which server your client is using, go to the *Expert* settings and disable *Answer Remote Hosts*. By checking *Broken Server*, the client is enabled to receive replies from a server communicating through an unprivileged port. For further information, see `man ypbind`.

7. Click *Finish* to save them and return to the YaST control center. Your client is now configured with NIS.

4 Setting Up Authentication Servers and Clients Using YaST

The Authentication Server is based on LDAP and optionally Kerberos. On openSUSE Leap you can configure it with a YaST wizard.

For more information about LDAP, see *Chapter 5, LDAP—A Directory Service*, and about Kerberos, see *Chapter 6, Network Authentication with Kerberos*.

4.1 Configuring an Authentication Server with YaST

4.1.1 Initial Configuration of an Authentication Server

To set up an authentication server for user account data, make sure the `yast2-auth-server`, `openldap2`, `krb5-server`, and `krb5-client` packages are installed; YaST will remind you and install them if one of these packages is missing. For Kerberos support, the `krb5-plugin-kdb-ldap` package is required.

The first part of the Authentication Server configuration with YaST is setting up an LDAP server, then you can enable Kerberos.

PROCEDURE 4.1: AUTHENTICATION SERVER CONFIGURATION WITH YAST

1. Start YaST as `root` and select *Network Services* › *Authentication Server* to invoke the configuration wizard.

2. Configure the *Global Settings* of your LDAP server (you can change these settings later)—see *Figure 4.1, "YaST Authentication Server Configuration"*:

FIGURE 4.1: YAST AUTHENTICATION SERVER CONFIGURATION

   a. Set LDAP to be started.

   b. If the LDAP server should announce its services via SLP, check *Register at an SLP Daemon*.

   c. Configure *Firewall Settings*.

   d. Click *Next*.

3. Select the server type: *Stand-alone server*, *Master server in a replication setup*, or *Replica (slave) server*.

4. Select security options (*TLS Settings*).
   It is strongly recommended to *Enable TLS*. For more information, see *Procedure 4.2, "Editing Authentication Server Configuration", Step 4*.

Warning: Authentication Without Encryption

When using authentication without enabling transport encryption using TLS, the password will be transmitted in the clear.

Also consider using LDAP over SSL with certificates.

5. Confirm *Basic Database Settings* with entering an *LDAP Administrator Password* and then clicking *Next*—see *Figure 4.2, "YaST LDAP Server—New Database"*.

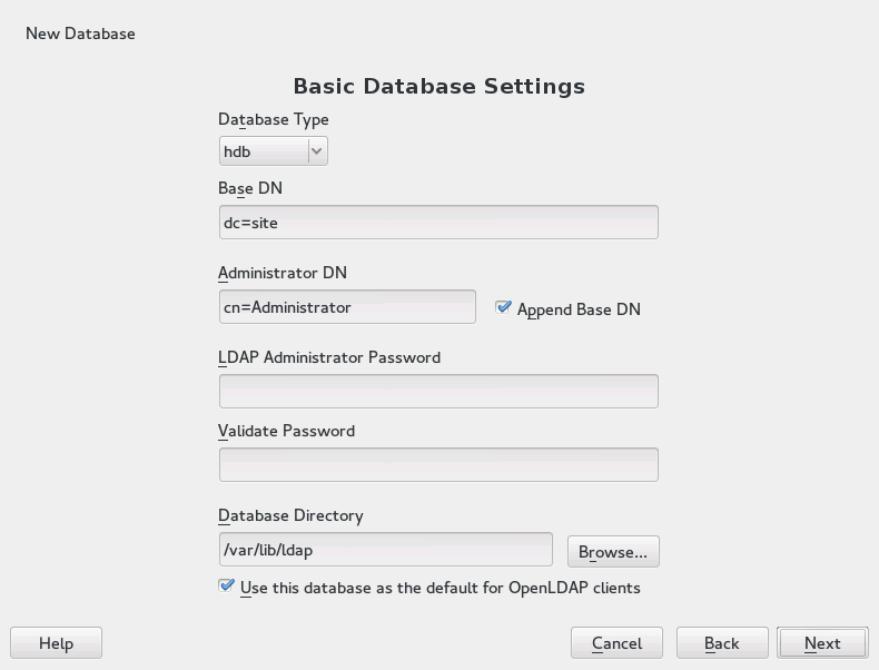


FIGURE 4.2: YAST LDAP SERVER—NEW DATABASE

6. In the *Kerberos Authentication* dialog, decide whether to enable Kerberos authentication or not (you can change these settings later)—see *Figure 4.3, "YaST Kerberos Authentication"*.

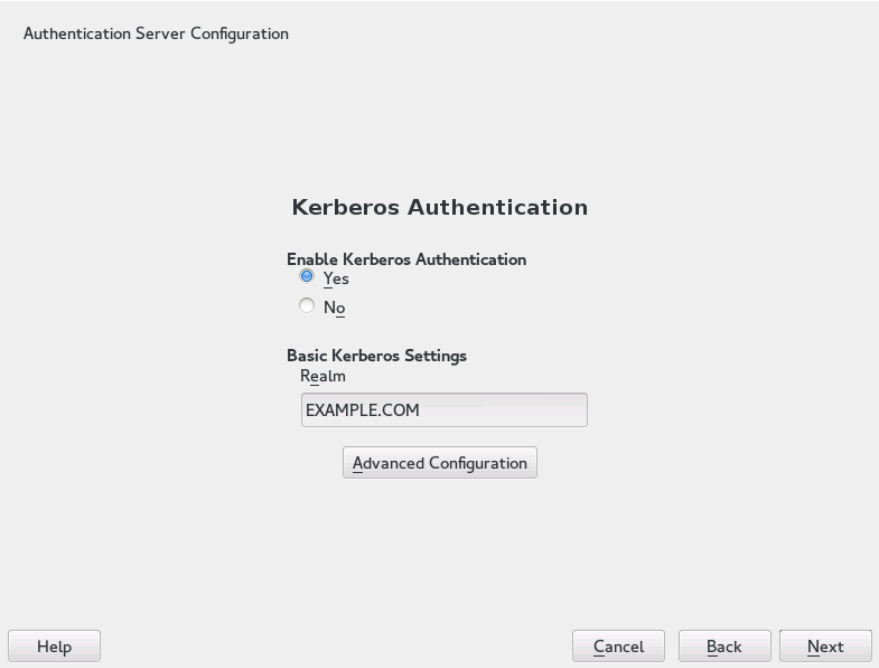


FIGURE 4.3: YAST KERBEROS AUTHENTICATION

7. Choose whether Kerberos support is needed or not. If you enable it, also specify your *Realm*. Then confirm with *Next*.

    - The *Advanced Configuration* allows you to specify various aspects such as *Maximum ticket life time* or ports to use.

8. Finally, check the *Authentication Server Configuration Summary* and click *Finish* to exit the configuration wizard.

4.1.2 Editing an Authentication Server Configuration with YaST

For changes or additional configuration start the Authentication Server module again and in the left pane expand *Global Settings* to make subentries visible—see *Figure 4.4, "YaST Editing Authentication Server Configuration"*:

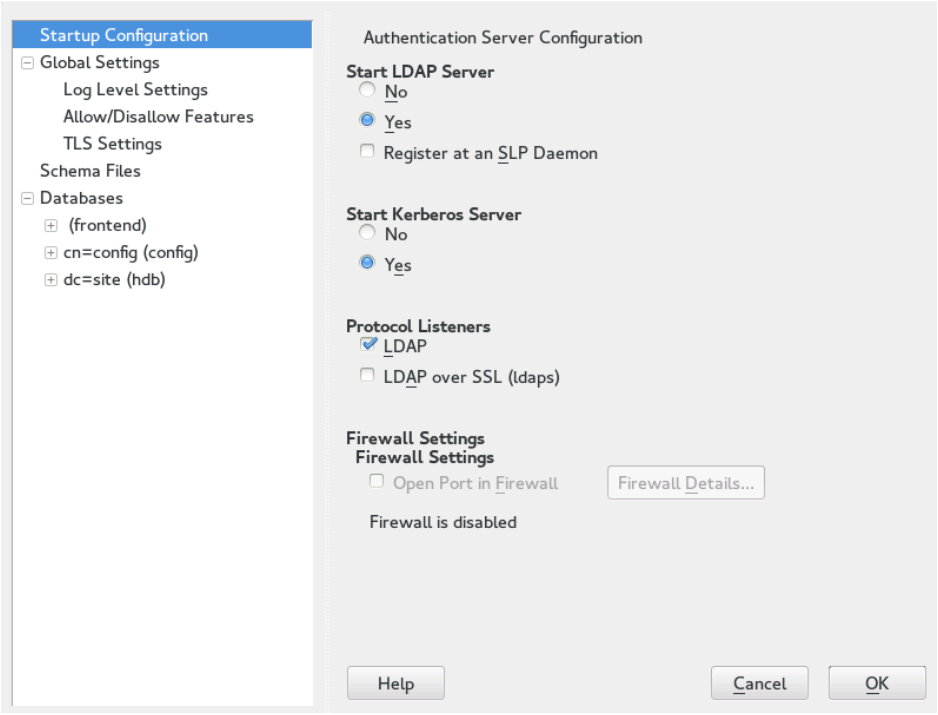


FIGURE 4.4: YAST EDITING AUTHENTICATION SERVER CONFIGURATION

PROCEDURE 4.2: EDITING AUTHENTICATION SERVER CONFIGURATION

1. With *Log Level Settings*, configure the degree of logging activity (verbosity) of the LDAP server. From the predefined list, select or deselect logging options according to your needs. The more options are enabled, the larger your log files grow.

2. Configure which connection types the server should offer under *Allow/Disallow Features*. Choose from:

   LDAPv2 Bind Requests

   This option enables connection requests (bind requests) from clients using the previous version of the protocol (LDAPv2).

   Anonymous Bind When Credentials Not Empty

Normally, the LDAP server denies any authentication attempts with empty credentials, that is, a distinguished name (DN) or a password. However, enabling this option makes it possible to connect with a password and no DN to establish an anonymous connection.

Unauthenticated Bind When DN Not Empty

Enabling this option makes it possible to connect without authentication (anonymously) using a distinguished name (DN) but no password.

Unauthenticated Update Options to Process

Enabling this option allows non-authenticated (anonymous) update operations. Access is restricted according to ACLs and other rules.

3. *Allow/Disallow Features* also lets you configure the server flags. Choose from:

Disable Acceptance of Anonymous Bind Requests

The server will no longer accept anonymous bind requests. Note, that this does not generally prohibit anonymous directory access.

Disable Simple Bind Authentication

Completely disable Simple Bind authentication.

Disable Forcing Session to Anonymous Status upon StartTLS Operation Receipt

The server will no longer force an authenticated connection back to the anonymous state when receiving the StartTLS operation.

Disallow the StartTLS Operation if Authenticated

The server will disallow the StartTLS operation on already authenticated connections.

4. To configure secure communication between client and server, proceed with *TLS Settings*:

   a. Activate *Enable TLS* to enable TLS and SSL encryption of the client/server communication.

   b. Either *Import Certificate* by specifying the exact path to its location or enable the *Use Common Server Certificate*. If the *Use Common Server Certificate* is not available, because it has not been created during installation, go for *Launch CA Management Module* first—for more information, see *Section 18.2, "YaST Modules for CA Management"*.

Add Schema files to be included in the server's configuration by selecting *Schema Files* in the left part of the dialog. The default selection of schema files applies to the server providing a source of YaST user account data.

YaST allows to add traditional Schema files (usually with a name ending in `.schema`) or LDIF files containing Schema definitions in OpenLDAP's LDIF Schema format.

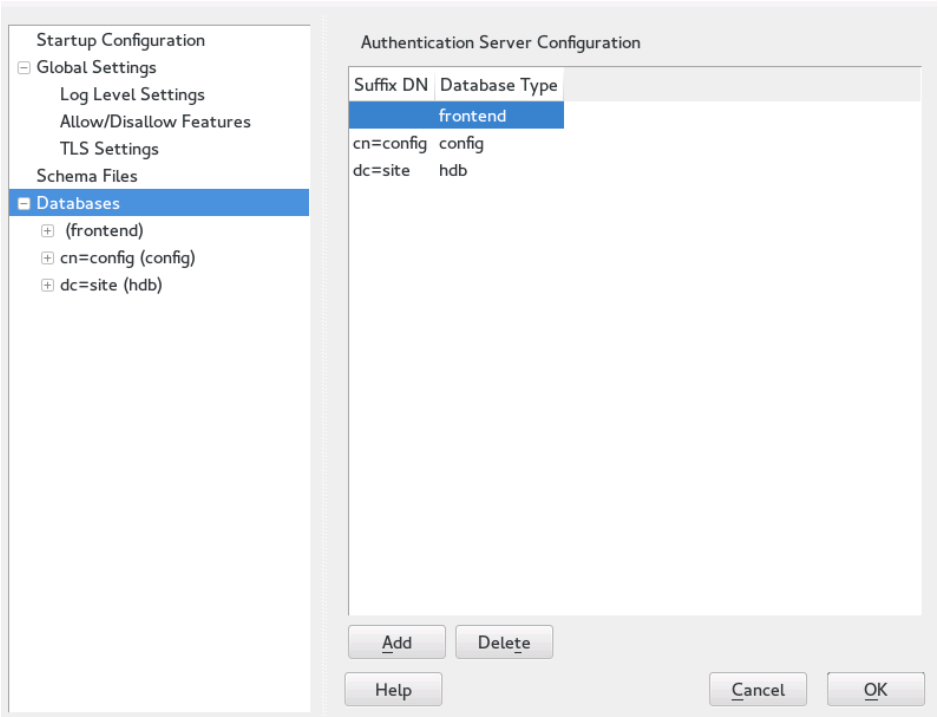


FIGURE 4.5: YAST AUTHENTICATION SERVER DATABASE CONFIGURATION

To configure the databases managed by your LDAP server, proceed as follows:

1. Select the *Databases* item in the left part of the dialog.
2. Click *Add Database* to add a new database.
3. Specify the requested data:

   Base DN

   Enter the base DN (distinguished name) of your LDAP server.

   Administrator DN

   Enter the DN of the administrator in charge of the server. If you check *Append Base DN*, only provide the `cn` of the administrator and the system fills in the rest automatically.

LDAP Administrator Password

Enter the password for the database administrator.

Use This Database as the Default for OpenLDAP Clients

For convenience, check this option if wanted.

4. In the next dialog, configure replication settings.

5. In the next dialog, enable enforcement of password policies to provide extra security to your LDAP server:

   a. Check *Enable Password Policies* to be able to specify a password policy.

   b. Activate *Hash Clear Text Passwords* to have clear text passwords be hashed before they are written to the database whenever they are added or modified.

   c. *Disclose "Account Locked" Status* provides a relevant error message for bind requests to locked accounts.

Warning: Locked Accounts in Security Sensitive Environments

Do not use the *Disclose "Account Locked" Status* option if your environment is sensitive to security issues, because the “Locked Account” error message provides security-sensitive information that can be exploited by a potential attacker.

   d. Enter the DN of the default policy object. To use a DN other than the one suggested by YaST, enter your choice. Otherwise, accept the default settings.

6. Complete the database configuration by clicking *Finish*.

If you have not opted for password policies, your server is ready to run at this point. If you have chosen to enable password policies, proceed with the configuration of the password policy in detail. If you have chosen a password policy object that does not yet exist, YaST creates one:

1. Enter the LDAP server password. In the navigation tree below *Databases* expand your database object and activate the *Password Policy Configuration* item.

2. Make sure *Enable Password Policies* is activated. Then click *Edit Policy*.

3. Configure the password change policies:

    a. Determine the number of passwords stored in the password history. Saved passwords may not be reused by the user.

    b. Determine if users can change their passwords and if they will need to change their passwords after a reset by the administrator. Require the old password for password changes (optional).

    c. Determine whether and to what extent passwords should be subject to quality checking. Set the minimum password length that must be met before a password is valid. If you select *Accept Uncheckable Passwords*, users are allowed to use encrypted passwords, even though the quality checks cannot be performed. If you opt for *Only Accept Checked Passwords* only those passwords that pass the quality tests are accepted as valid.

4. Configure the password time-limit policies:

    a. Determine the minimum password time-limit (the time that needs to pass between two valid password changes) and the maximum password time limit.

    b. Determine the time between a password expiration warning and the actual password expiration.

    c. Set the number of postponement uses of an expired password before the password expires permanently.

5. Configure the lockout policies:

    a. Enable password locking.

    b. Determine the number of bind failures that trigger a password lock.

    c. Determine the duration of the password lock.

    d. Determine the length of time that password failures are kept in the cache before they are purged.

6. Apply your password policy settings with *OK*.

To edit a previously created database, select its base DN in the tree to the left. In the right part of the window, YaST displays a dialog similar to the one used for the creation of a new database (with the main difference that the base DN entry is grayed out and cannot be changed).

After leaving the Authentication Server configuration by selecting *Finish*, you are ready to go with a basic working configuration for your Authentication Server. To fine-tune this setup, use OpenLDAP's dynamic configuration back-end.

The OpenLDAP's dynamic configuration back-end stores the configuration in an LDAP database. That database consists of a set of `.ldif` files in `/etc/openldap/slapd.d`. There is no need to access these files directly. To access the settings you can either use the YaST Authentication Server module (the `yast2-auth-server` package) or an LDAP client such as **`ldapmodify`** or **`ldapsearch`**. For more information on the dynamic configuration of OpenLDAP, see the "OpenLDAP Administration Guide".

4.1.3 Editing LDAP Users and Groups

For editing LDAP users and groups with YaST, see *Section 5.4, "Configuring LDAP Users and Groups in YaST"*.

4.2 Configuring an Authentication Client with YaST

YaST allows setting up authentication to clients using different modules:

- ***User Logon Management***. Use both an identity service (usually LDAP) and a user authentication service (usually Kerberos). This option is based on SSSD and in the majority of cases is best suited for joining Active Directory domains.
  This module is described in *Section 7.3.2, "Joining Active Directory Using User Logon Management"*.

- ***Windows Domain Membership***. Join an Active Directory (which entails use of Kerberos and LDAP). This option is based on **`winbind`** and is best suited for joining an Active Directory domain if support for NTLM or cross-forest trusts is necessary.
  This module is described in *Section 7.3.3, "Joining Active Directory Using Windows Domain Membership"*.

- ***LDAP and Kerberos Authentication***. Allows setting up LDAP identities and Kerberos authentication independently from each other and provides fewer options. While this module also uses SSSD, it is not as well suited for connecting to Active Directory as the previous two options.

This module is described in:

- LDAP: *Section 5.3, "Configuring an LDAP Client with YaST"*
- Kerberos: *Section 6.6, "Setting up Kerberos using LDAP and Kerberos Client"*

4.3 SSSD

Two of the YaST modules are based on SSSD: *User Logon Management* and *LDAP and Kerberos Authentication*.

SSSD stands for System Security Services Daemon. SSSD talks to remote directory services that provide user data and provides various authentication methods, such as LDAP, Kerberos, or Active Directory (AD). It also provides an NSS (Name Service Switch) and PAM (Pluggable Authentication Module) interface.

SSSD can locally cache user data and then allow users to use the data, even if the real directory service is (temporarily) unreachable.

4.3.1 Checking the Status

After running one of the YaST authentication modules, you can check whether SSSD is running with:

```
root # systemctl status sssd
sssd.service - System Security Services Daemon
   Loaded: loaded (/usr/lib/systemd/system/sssd.service; enabled)
   Active: active (running) since Thu 2015-10-23 11:03:43 CEST; 5s ago
   [...]
```

4.3.2 Caching

To allow logging in when the authentication back-end is unavailable, SSSD will use its cache even if it was invalidated. This happens until the back-end is available again.

To invalidate the cache, run `sss_cache -E` (the command `sss_cache` is part of the package `sssd-tools`).

To completely remove the SSSD cache, run:

```
tux > sudo systemctl stop sssd
```

```
tux > sudo rm -f /var/lib/sss/db/*
tux > sudo systemctl start sssd
```

4.3.3 For More Information

For more information, see the SSSD man pages `sssd.conf` (**`man sssd.conf`**) and `sssd` (**`man sssd`**). There are also man pages for most SSSD modules.

5 LDAP—A Directory Service

The Lightweight Directory Access Protocol (LDAP) is a set of protocols designed to access and maintain information directories. LDAP can be used for user and group management, system configuration management, address management, and more. This chapter provides a basic understanding of how OpenLDAP works.

In a network environment, it is crucial to keep important information structured and to serve it quickly. A directory service keeps information available in a well-structured and searchable form.

Ideally, a central server stores the data in a directory and distributes it to all clients using a well-defined protocol. The structured data allow a wide range of applications to access them. A central repository reduces the necessary administrative effort. The use of an open and standardized protocol like LDAP ensures that as many client applications as possible can access such information.

A directory in this context is a type of database optimized for quick and effective reading and searching:

- To make multiple concurrent reading accesses possible, the number of updates is usually very low. The number of read and write accesses is often limited to a few users with administrative privileges. In contrast, conventional databases are optimized for accepting the largest possible data volume in a short time.

- When static data is administered, updates of the existing data sets are very rare. When working with dynamic data, especially when data sets like bank accounts or accounting are concerned, the consistency of the data is of primary importance. If an amount should be subtracted from one place to be added to another, both operations must happen concurrently, within one *transaction*, to ensure balance over the data stock. Traditional relational databases usually have a very strong focus on data consistency, such as the referential integrity support of transactions. Conversely, short-term inconsistencies are usually acceptable in LDAP directories. LDAP directories often do not have the same strong consistency requirements as relational databases.

The design of a directory service like LDAP is not laid out to support complex update or query mechanisms. All applications are guaranteed to access this service quickly and easily.

5.1 LDAP versus NIS

Unix system administrators traditionally use NIS (Network Information Service) for name resolution and data distribution in a network. The configuration data contained in the files `group`, `hosts`, `mail`, `netgroup`, `networks`, `passwd`, `printcap`, `protocols`, `rpc`, and `services` in the `/etc` directory is distributed to clients all over the network. These files can be maintained without major effort because they are simple text files. The handling of larger amounts of data, however, becomes increasingly difficult because of nonexistent structuring. NIS is only designed for Unix platforms, and is not suitable as a centralized data administration tool in heterogeneous networks.

Unlike NIS, the LDAP service is not restricted to pure Unix networks. Windows™ servers (starting with Windows 2000) support LDAP as a directory service. The application tasks mentioned above are additionally supported in non-Unix systems.

The LDAP principle can be applied to any data structure that needs to be centrally administered. A few application examples are:

- Replacement for the NIS service
- Mail routing (postfix)
- Address books for mail clients, like Mozilla Thunderbird, Evolution, and Outlook
- Administration of zone descriptions for a BIND 9 name server
- User authentication with Samba in heterogeneous networks

This list can be extended because LDAP is extensible, unlike NIS. The clearly-defined hierarchical structure of the data simplifies the administration of large amounts of data, as it can be searched more easily.

5.2 Structure of an LDAP Directory Tree

To get background knowledge on how an LDAP server works and how the data is stored, it is vital to understand the way the data is organized on the server and how this structure enables LDAP to provide fast access to the data. To successfully operate an LDAP setup, you also need to be familiar with some basic LDAP terminology. This section introduces the basic layout of an LDAP directory tree and provides the basic terminology used with regard to LDAP. Skip this

introductory section if you already have some LDAP background knowledge and only want to learn how to set up an LDAP environment in openSUSE Leap. Read on at *Section 5.5, "Manually Configuring an LDAP Server"*.

An LDAP directory has a tree structure. All entries (called objects) of the directory have a defined position within this hierarchy. This hierarchy is called the *directory information tree* (DIT). The complete path to the desired entry, which unambiguously identifies it, is called the *distinguished name* or DN. A single node along the path to this entry is called *relative distinguished name* or RDN.

The relations within an LDAP directory tree become more evident in the following example, shown in *Figure 5.1, "Structure of an LDAP Directory"*.

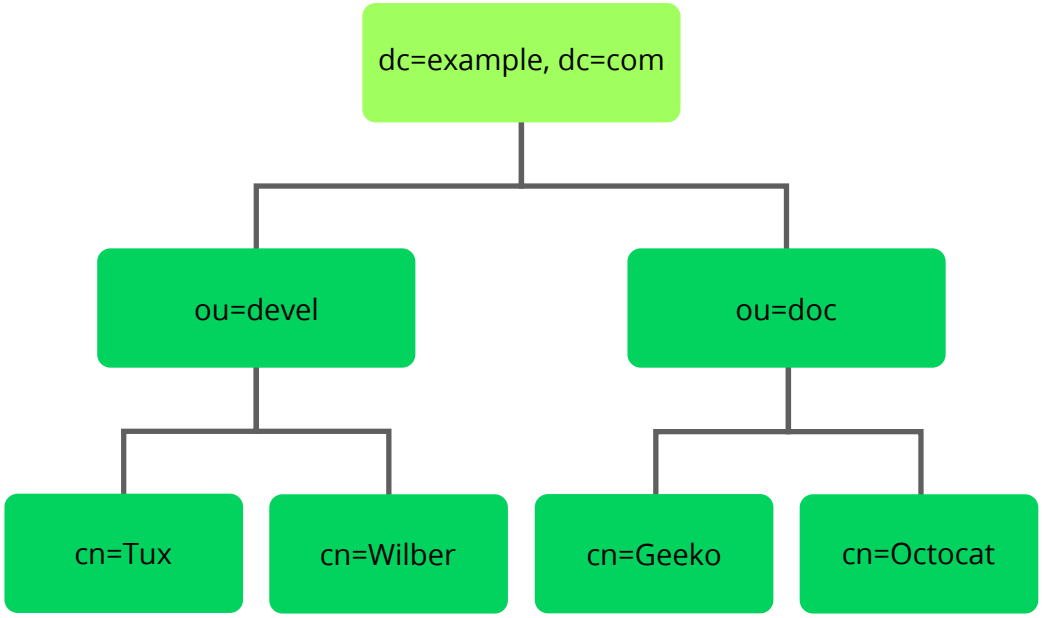


FIGURE 5.1: STRUCTURE OF AN LDAP DIRECTORY

The complete diagram is a fictional directory information tree. The entries on three levels are depicted. Each entry corresponds to one box in the image. The complete, valid *distinguished name* for the fictional employee `Geeko Linux`, in this case, is `cn=Geeko Linux,ou=doc,dc=example,dc=com`. It is composed by adding the RDN `cn=Geeko Linux` to the DN of the preceding entry `ou=doc,dc=example,dc=com`.

The types of objects that can be stored in the DIT are globally determined following a *Schema*. The type of an object is determined by the *object class*. The object class determines what attributes the relevant object must or can be assigned. The Schema, therefore, must contain definitions of all object classes and attributes used in the desired application scenario. There are a few common Schemas (see RFC 2252 and 2256). The LDAP RFC defines a few commonly used Schemas (see for

example, RFC4519). Additionally, Schemas are available for many other use cases (for example, Samba or NIS replacement). It is, however, possible to create custom Schemas or to use multiple Schemas complementing each other (if this is required by the environment in which the LDAP server should operate).

Table 5.1, "Commonly Used Object Classes and Attributes" offers a small overview of the object classes from `core.schema` and `inetorgperson.schema` used in the example, including required attributes (Req. Attr.) and valid attribute values.

TABLE 5.1: COMMONLY USED OBJECT CLASSES AND ATTRIBUTES

Object Class	Meaning	Example Entry	Req. Attr.
`dcObject`	*domainComponent* (name components of the domain)	example	dc
`organizationalUnit`	*organizationalUnit* (organizational unit)	doc	ou
`inetOrgPerson`	*inetOrgPerson* (person-related data for the intranet or Internet)	Geeko Linux	sn and cn

Example 5.1, "Excerpt from schema.core" shows an excerpt from a Schema directive with explanations.

EXAMPLE 5.1: EXCERPT FROM SCHEMA.CORE

```
attributetype (2.5.4.11 NAME ( 'ou' 'organizationalUnitName') ❶
       DESC 'RFC2256: organizational unit this object belongs to' ❷
       SUP name ) ❸

objectclass ( 2.5.6.5 NAME 'organizationalUnit' ❹
       DESC 'RFC2256: an organizational unit' ❺
       SUP top STRUCTURAL ❻
       MUST ou ❼
MAY (userPassword $ searchGuide $ seeAlso $ businessCategory ❽
  $ x121Address $ registeredAddress $ destinationIndicator
  $ preferredDeliveryMethod $ telexNumber
  $ teletexTerminalIdentifier $ telephoneNumber
  $ internationaliSDNNumber $ facsimileTelephoneNumber
  $ street $ postOfficeBox $ postalCode $ postalAddress
```

```
  $ physicalDeliveryOfficeName
  $ st $ l $ description) )
  ...
```

The attribute type `organizationalUnitName` and the corresponding object class `organizationalUnit` serve as an example here.

1. The name of the attribute, its unique OID (*object identifier*) (numerical), and the abbreviation of the attribute.
2. A brief description of the attribute with `DESC`. The corresponding RFC, on which the definition is based, is also mentioned here.
3. `SUP` indicates a superordinate attribute type to which this attribute belongs.
4. The definition of the object class `organizationalUnit` begins—the same as in the definition of the attribute—with an OID and the name of the object class.
5. A brief description of the object class.
6. The `SUP top` entry indicates that this object class is not subordinate to another object class.
7. With `MUST` list all attribute types that must be used with an object of the type `organizationalUnit`.
8. With `MAY` list all attribute types that are permitted with this object class.

A very good introduction to the use of Schemas can be found in the OpenLDAP documentation (`openldap2-doc`). When installed, find it in `/usr/share/doc/packages/openldap2/adminguide/guide.html`.

5.3 Configuring an LDAP Client with YaST

YaST includes the module *LDAP and Kerberos Client* that helps define authentication scenarios involving either LDAP or Kerberos.

It can also be used to join Kerberos and LDAP separately. However, in many such cases, using this module may not be the first choice, such as for joining Active Directory (which uses a combination of LDAP and Kerberos). For more information, see *Section 4.2, "Configuring an Authentication Client with YaST"*.

Start the module by selecting *Network Services* › *LDAP and Kerberos Client*.

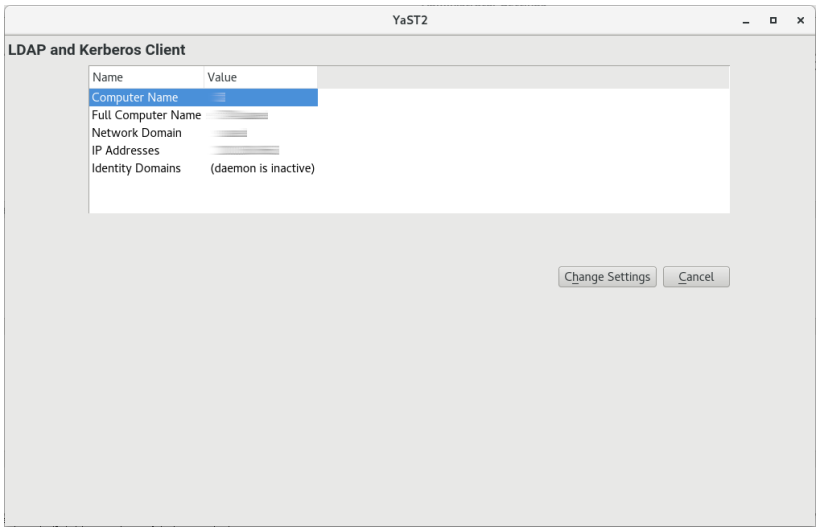
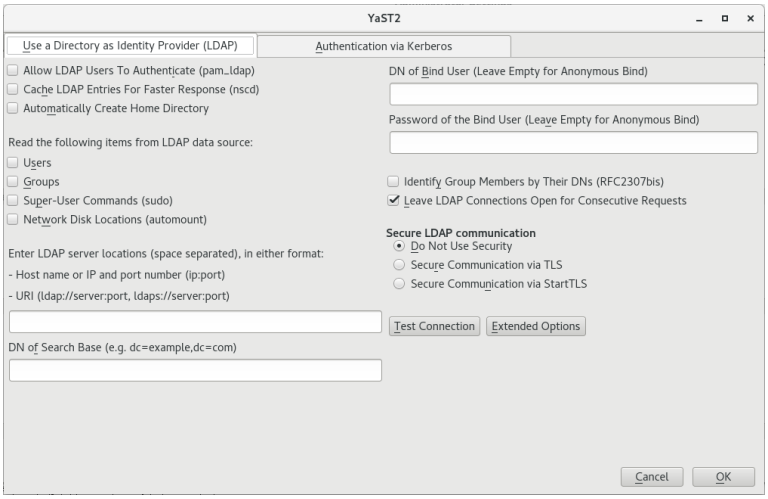


FIGURE 5.2: *LDAP AND KERBEROS CLIENT* WINDOW

To configure an LDAP client, follow the procedure below:

1. In the window *LDAP and Kerberos Client*, click *Change Settings*.
   Make sure that the tab *Use a Directory as Identity Provider (LDAP)* is chosen.



2. Specify one or more LDAP server URLs, host names, or IP addresses under *Enter LDAP server locations*. When specifying multiple addresses, separate them with spaces.

3. Specify the appropriate LDAP distinguished name (DN) under *DN of Search Base*. For example, a valid entry could be `dc=example,dc=com`.

4. If your LDAP server supports TLS encryption, choose the appropriate security option under *Secure LDAP Connection*.

To first ask the server whether it supports TLS encryption and be able to downgrade to an unencrypted connection if it does not, use *Secure Communication via StartTLS*.

5. Activate other options as necessary:

   - You can *Allow users to authenticate via LDAP* and *Automatically Create Home Directories* on the local computer for them.
   - Use *Cache LDAP Entries For Faster Response* to cache LDAP entries locally. However, this bears the danger that entries can be slightly out of date.
   - Specify the types of data that should be used from the LDAP source, such as *Users* and *Groups*, *Super-User Commands*, and *Network Disk Locations* (network-shared drives that can be automatically mounted on request).
   - Specify the distinguished name (DN) and password of the user under whose name you want to bind to the LDAP directory in *DN of Bind User* and *Password of the Bind User*.
     Otherwise, if the server supports it, you can also leave both text boxes empty to bind anonymously to the server.

     ### Warning: Authentication Without Encryption

     When using authentication without enabling transport encryption using TLS or StartTLS, the password will be transmitted in the clear.

   Under *Extended Options*, you can additionally configure timeouts for BIND operations.

6. To check whether the LDAP connection works, click *Test Connection*.
7. To leave the dialog, click *OK*. Then wait for the setup to complete.
   Finally, click *Finish*.

5.4 Configuring LDAP Users and Groups in YaST

The actual registration of user and group data differs only slightly from the procedure when not using LDAP. The following instructions relate to the administration of users. The procedure for administering groups is analogous.

1. Access the YaST user administration with *Security and Users* › *User and Group Management*.

2. Use *Set Filter* to limit the view of users to the LDAP users and enter the password for Root DN.
3. Click *Add* to enter the user configuration. A dialog with four tabs opens:
    a. Specify the user's name, login name, and password in the *User Data* tab.
    b. Check the *Details* tab for the group membership, login shell, and home directory of the new user. If necessary, change the default to values that better suit your needs.
    c. Modify or accept the default *Password Settings*.
    d. Enter the *Plug-Ins* tab, select the LDAP plug-in, and click *Launch* to configure additional LDAP attributes assigned to the new user.
4. Click *OK* to apply your settings and leave the user configuration.

The initial input form of user administration offers *LDAP Options*. This allows you to apply LDAP search filters to the set of available users. Alternatively open the module for configuring LDAP users and groups by selecting *LDAP User and Group Configuration*.

5.5 Manually Configuring an LDAP Server

YaST uses OpenLDAP's dynamic configuration database (`back-config`) to store the LDAP server's configuration. For details about the dynamic configuration back-end, see the `slapd-config(5)` man page or the OpenLDAP Software 2.4 Administrator's Guide located at `/usr/share/doc/packages/openldap2/guide/admin/guide.html` on your system if the `openldap2` package is installed.

Tip: Upgrading an Old OpenLDAP Installation

YaST does not use `/etc/openldap/slapd.conf` to store the OpenLDAP configuration anymore. In case of a system upgrade, a copy of the original `/etc/openldap/slapd.conf` file will get created as `/etc/openldap/slapd.conf.YaSTsave`.

To conveniently access the configuration back-end, you use SASL external authentication. For example, the following **ldapsearch** command executed as `root` can show the complete **slapd** configuration:

```
tux > ldapsearch -Y external -H ldapi:/// -b cn=config
```

Note: LDAP Server Is Part of the Authentication Server

Basic LDAP Server initialization and configuration can be done within the Authentication Server YaST module. For more information, see *Section 4.1, "Configuring an Authentication Server with YaST"*.

When the LDAP server is fully configured and all desired entries have been made according to the pattern described in *Section 5.6, "Manually Administering LDAP Data"*, start the LDAP server as `root` by entering **sudo systemctl start slapd**. To stop the server manually, enter the command **sudo systemctl stop slapd**. Query the status of the running LDAP server with **sudo systemctl status slapd**.

Use the YaST *Services Manager*, described in *Book "Reference", Chapter 10 "The* `systemd` *Daemon", Section 10.4 "Managing Services with YaST"*, to have the server started and stopped automatically on system bootup and shutdown. You can also create the corresponding links to the start and stop scripts with the **systemctl** commands as described in *Book "Reference", Chapter 10 "The* `systemd` *Daemon", Section 10.2.1 "Managing Services in a Running System"*.

5.6 Manually Administering LDAP Data

OpenLDAP offers a series of tools for the administration of data in the LDAP directory. The four most important tools for adding to, deleting from, searching through and modifying the data stock are explained in this section.

5.6.1 Inserting Data into an LDAP Directory

Once your LDAP server is correctly configured (it features appropriate entries for `suffix`, `directory`, `rootdn`, `rootpw` and `index`), proceed to entering records. OpenLDAP offers the **ldapadd** command for this task. If possible, add the objects to the database in bundles (for

practical reasons). LDAP can process the LDIF format (LDAP data interchange format) for this. An LDIF file is a simple text file that can contain an arbitrary number of attribute and value pairs. The LDIF file for creating a rough framework for the example in *Figure 5.1, "Structure of an LDAP Directory"* would look like the one in *Example 5.2, "An LDIF File"*.

Important: Encoding of LDIF Files

LDAP works with UTF-8 (Unicode). Umlauts must be encoded correctly. Otherwise, avoid umlauts and other special characters or use **`iconv`** to convert the input to UTF-8.

EXAMPLE 5.2: AN LDIF FILE

```
# The Organization
dn: dc=example,dc=com
objectClass: dcObject
objectClass: organization
o: Example dc: example

# The organizational unit development (devel)
dn: ou=devel,dc=example,dc=com
objectClass: organizationalUnit
ou: devel

# The organizational unit documentation (doc)
dn: ou=doc,dc=example,dc=com
objectClass: organizationalUnit
ou: doc

# The organizational unit internal IT (it)
dn: ou=it,dc=example,dc=com
objectClass: organizationalUnit
ou: it
```

Save the file with the `.ldif` suffix then pass it to the server with the following command:

```
tux > ldapadd -x -D DN_OF_THE_ADMINISTRATOR -W -f FILE.ldif
```

`-x` switches off the authentication with SASL in this case. `-D` declares the user that calls the operation. The valid DN of the administrator is entered here, as it has been configured in `slapd.conf`. In the current example, this is `cn=Administrator,dc=example,dc=com`. `-W` circumvents entering the password on the command line (in clear text) and activates a separate password prompt. The `-f` option passes the file name. See the details of running **`ldapadd`** in *Example 5.3, "ldapadd with example.ldif"*.

EXAMPLE 5.3: LDAPADD WITH EXAMPLE.LDIF

```
tux > ldapadd -x -D cn=Administrator,dc=example,dc=com -W -f example.ldif

Enter LDAP password:
adding new entry "dc=example,dc=com"
adding new entry "ou=devel,dc=example,dc=com"
adding new entry "ou=doc,dc=example,dc=com"
adding new entry "ou=it,dc=example,dc=com"
```

The user data of individuals can be prepared in separate LDIF files. *Example 5.4, "LDIF Data for Tux"* adds `Tux` to the new LDAP directory.

EXAMPLE 5.4: LDIF DATA FOR TUX

```
# coworker Tux
dn: cn=Tux Linux,ou=devel,dc=example,dc=com
objectClass: inetOrgPerson
cn: Tux Linux
givenName: Tux
sn: Linux
mail: tux@example.com
uid: tux
telephoneNumber: +49 1234 567-8
```

An LDIF file can contain an arbitrary number of objects. It is possible to pass directory branches (entirely or in part) to the server in one go, as shown in the example of individual objects. If it is necessary to modify some data relatively often, a fine subdivision of single objects is recommended.

5.6.2 Modifying Data in the LDAP Directory

The tool **`ldapmodify`** is provided for modifying the data stock. The easiest way to do this is to modify the corresponding LDIF file and pass the modified file to the LDAP server. To change the telephone number of colleague Tux from `+49 1234 567-8` to `+49 1234 567-10`, edit the LDIF file like in *Example 5.5, "Modified LDIF File tux.ldif"*.

EXAMPLE 5.5: MODIFIED LDIF FILE TUX.LDIF

```
# coworker Tux
dn: cn=Tux Linux,ou=devel,dc=example,dc=com
```

```
changetype: modify
replace: telephoneNumber
telephoneNumber: +49 1234 567-10
```

Import the modified file into the LDAP directory with the following command:

```
tux > ldapmodify -x -D cn=Administrator,dc=example,dc=com -W -f tux.ldif
```

Alternatively, pass the attributes to change directly to **ldapmodify** as follows:

1. Start **ldapmodify** and enter your password:

   ```
   tux > ldapmodify -x -D cn=Administrator,dc=example,dc=com -W
   Enter LDAP password:
   ```

2. Enter the changes while carefully complying with the syntax in the order presented below:

   ```
   dn: cn=Tux Linux,ou=devel,dc=example,dc=com
   changetype: modify
   replace: telephoneNumber
   telephoneNumber: +49 1234 567-10
   ```

For more information about **ldapmodify** and its syntax, see the **ldapmodify** man page.

5.6.3 Searching or Reading Data from an LDAP Directory

OpenLDAP provides, with **ldapsearch**, a command line tool for searching data within an LDAP directory and reading data from it. This is a simple query:

```
tux > ldapsearch -x -b dc=example,dc=com "(objectClass=*)"
```

The `-b` option determines the search base (the section of the tree within which the search should be performed). In the current case, this is `dc=example,dc=com`. To perform a more finely-grained search in specific subsections of the LDAP directory (for example, only within the `devel` department), pass this section to **ldapsearch** with `-b`. `-x` requests activation of simple authentication. `(objectClass=*)` declares that all objects contained in the directory should be read. This command option can be used after the creation of a new directory tree to verify that all entries have been recorded correctly and the server responds as desired. For more information about the use of **ldapsearch**, see the `ldapsearch(1)` man page.

5.6.4 Deleting Data from an LDAP Directory

Delete unwanted entries with **ldapdelete**. The syntax is similar to that of the other commands. To delete, for example, the complete entry for `Tux Linux`, issue the following command:

```
tux > ldapdelete -x -D cn=Administrator,dc=example,dc=com -W cn=Tux \
Linux,ou=devel,dc=example,dc=com
```

5.7 For More Information

More complex subjects (like SASL configuration or establishment of a replicating LDAP server that distributes the workload among multiple slaves) were omitted from this chapter. Find detailed information about both subjects in the *OpenLDAP 2.4 Administrator's Guide*—see at *OpenLDAP 2.4 Administrator's Guide*.

The Web site of the OpenLDAP project offers exhaustive documentation for beginner and advanced LDAP users:

OpenLDAP Faq-O-Matic

A detailed question and answer collection applying to the installation, configuration, and use of OpenLDAP. Find it at http://www.openldap.org/faq/data/cache/1.html.

Quick Start Guide

Brief step-by-step instructions for installing your first LDAP server. Find it at http://www.openldap.org/doc/admin24/quickstart.html or on an installed system in Section 2 of `/usr/share/doc/packages/openldap2/guide/admin/guide.html`.

OpenLDAP 2.4 Administrator's Guide

A detailed introduction to all important aspects of LDAP configuration, including access controls and encryption. See http://www.openldap.org/doc/admin24/ or, on an installed system, `/usr/share/doc/packages/openldap2/guide/admin/guide.html`.

Understanding LDAP

A detailed general introduction to the basic principles of LDAP: http://www.redbooks.ibm.com/redbooks/pdfs/sg244986.pdf.

Printed literature about LDAP:

- *LDAP System Administration* by Gerald Carter (ISBN 1-56592-491-6)
- *Understanding and Deploying LDAP Directory Services* by Howes, Smith, and Good (ISBN 0-672-32316-8)

The ultimate reference material for the subject of LDAP are the corresponding RFCs (request for comments), 2251 to 2256.

6 Network Authentication with Kerberos

Kerberos is a network authentication protocol which also provides encryption. This chapter describes how to set up Kerberos and integrate services like LDAP and NFS.

6.1 Conceptual Overview

An open network provides no means of ensuring that a workstation can identify its users properly, except through the usual password mechanisms. In common installations, the user must enter the password each time a service inside the network is accessed. Kerberos provides an authentication method with which a user registers only once and is trusted in the complete network for the rest of the session. To have a secure network, the following requirements must be met:

- Have all users prove their identity for each desired service and make sure that no one can take the identity of someone else.
- Make sure that each network server also proves its identity. Otherwise an attacker might be able to impersonate the server and obtain sensitive information transmitted to the server. This concept is called *mutual authentication*, because the client authenticates to the server and vice versa.

Kerberos helps you meet these requirements by providing strongly encrypted authentication. Only the basic principles of Kerberos are discussed here. For detailed technical instruction, refer to the Kerberos documentation.

6.2 Kerberos Terminology

The following glossary defines some Kerberos terminology.

credential

Users or clients need to present some kind of credentials that authorize them to request services. Kerberos knows two kinds of credentials—tickets and authenticators.

ticket

A ticket is a per-server credential used by a client to authenticate at a server from which it is requesting a service. It contains the name of the server, the client's name, the client's Internet address, a time stamp, a lifetime, and a random session key. All this data is encrypted using the server's key.

authenticator

Combined with the ticket, an authenticator is used to prove that the client presenting a ticket is really the one it claims to be. An authenticator is built using the client's name, the workstation's IP address, and the current workstation's time, all encrypted with the session key known only to the client and the relevant server. An authenticator can only be used once, unlike a ticket. A client can build an authenticator itself.

principal

A Kerberos principal is a unique entity (a user or service) to which it can assign a ticket. A principal consists of the following components:

```
USER/INSTANCE@REALM
```

- **primary:** The first part of the principal. In the case of users, this is usually the same as the user name.
- **instance *(optional)*:** Additional information characterizing the *primary*. This string is separated from the *primary* by a `/`.
  `tux@example.org` and `tux/admin@example.org` can both exist on the same Kerberos system and are treated as different principals.
- **realm:** Specifies the Kerberos realm. Normally, your realm is your domain name in uppercase letters.

mutual authentication

Kerberos ensures that both client and server can be sure of each others identity. They share a session key, which they can use to communicate securely.

session key

Session keys are temporary private keys generated by Kerberos. They are known to the client and used to encrypt the communication between the client and the server for which it requested and received a ticket.

replay

Almost all messages sent in a network can be eavesdropped, stolen, and resent. In the Kerberos context, this would be most dangerous if an attacker manages to obtain your request for a service containing your ticket and authenticator. The attacker could then try to resend it (*replay*) to impersonate you. However, Kerberos implements several mechanisms to deal with this problem.

server or service

Service is used to refer to a specific action to perform. The process behind this action is called a *server*.

6.3 How Kerberos Works

Kerberos is often called a third-party trusted authentication service, which means all its clients trust Kerberos's judgment of another client's identity. Kerberos keeps a database of all its users and their private keys.

To ensure Kerberos is working correctly, run both the authentication and ticket-granting server on a dedicated machine. Make sure that only the administrator can access this machine physically and over the network. Reduce the (networking) services running on it to the absolute minimum—do not even run `sshd`.

6.3.1 First Contact

Your first contact with Kerberos is quite similar to any login procedure at a normal networking system. Enter your user name. This piece of information and the name of the ticket-granting service are sent to the authentication server (Kerberos). If the authentication server knows you, it generates a random session key for further use between your client and the ticket-granting server. Now the authentication server prepares a ticket for the ticket-granting server. The ticket contains the following information—all encrypted with a session key only the authentication server and the ticket-granting server know:

- The names of both, the client and the ticket-granting server
- The current time
- A lifetime assigned to this ticket

- The client's IP address
- The newly-generated session key

This ticket is then sent back to the client together with the session key, again in encrypted form, but this time the private key of the client is used. This private key is only known to Kerberos and the client, because it is derived from your user password. Now that the client has received this response, you are prompted for your password. This password is converted into the key that can decrypt the package sent by the authentication server. The package is “unwrapped” and password and key are erased from the workstation's memory. As long as the lifetime given to the ticket used to obtain other tickets does not expire, your workstation can prove your identity.

6.3.2 Requesting a Service

To request a service from any server in the network, the client application needs to prove its identity to the server. Therefore, the application generates an authenticator. An authenticator consists of the following components:

- The client's principal
- The client's IP address
- The current time
- A checksum (chosen by the client)

All this information is encrypted using the session key that the client has already received for this special server. The authenticator and the ticket for the server are sent to the server. The server uses its copy of the session key to decrypt the authenticator, which gives it all the information needed about the client requesting its service, to compare it to that contained in the ticket. The server checks if the ticket and the authenticator originate from the same client.

Without any security measures implemented on the server side, this stage of the process would be an ideal target for replay attacks. Someone could try to resend a request stolen off the net some time before. To prevent this, the server does not accept any request with a time stamp and ticket received previously. In addition to that, a request with a time stamp differing too much from the time the request is received is ignored.

6.3.3 Mutual Authentication

Kerberos authentication can be used in both directions. It is not only a question of the client being the one it claims to be. The server should also be able to authenticate itself to the client requesting its service. Therefore, it sends an authenticator itself. It adds one to the checksum it received in the client's authenticator and encrypts it with the session key, which is shared between it and the client. The client takes this response as a proof of the server's authenticity and they both start cooperating.

6.3.4 Ticket Granting—Contacting All Servers

Tickets are designed to be used for one server at a time. Therefore, you need to get a new ticket each time you request another service. Kerberos implements a mechanism to obtain tickets for individual servers. This service is called the “ticket-granting service”. The ticket-granting service is a service (like any other service mentioned before) and uses the same access protocols that have already been outlined. Any time an application needs a ticket that has not already been requested, it contacts the ticket-granting server. This request consists of the following components:

- The requested principal
- The ticket-granting ticket
- An authenticator

Like any other server, the ticket-granting server now checks the ticket-granting ticket and the authenticator. If they are considered valid, the ticket-granting server builds a new session key to be used between the original client and the new server. Then the ticket for the new server is built, containing the following information:

- The client's principal
- The server's principal
- The current time
- The client's IP address
- The newly-generated session key

The new ticket has a lifetime, which is either the remaining lifetime of the ticket-granting ticket or the default for the service. The lesser of both values is assigned. The client receives this ticket and the session key, which are sent by the ticket-granting service. But this time the answer is encrypted with the session key that came with the original ticket-granting ticket. The client can decrypt the response without requiring the user's password when a new service is contacted. Kerberos can thus acquire ticket after ticket for the client without bothering the user.

6.4 User View of Kerberos

Ideally, a user only contact with Kerberos happens during login at the workstation. The login process includes obtaining a ticket-granting ticket. At logout, a user's Kerberos tickets are automatically destroyed, which makes it difficult for anyone else to impersonate this user.

The automatic expiration of tickets can lead to a situation when a user's login session lasts longer than the maximum lifespan given to the ticket-granting ticket (a reasonable setting is 10 hours). However, the user can get a new ticket-granting ticket by running **`kinit`**. Enter the password again and Kerberos obtains access to desired services without additional authentication. To get a list of all the tickets silently acquired for you by Kerberos, run **`klist`**.

Here is a short list of applications that use Kerberos authentication. These applications can be found under `/usr/lib/mit/bin` or `/usr/lib/mit/sbin` after installing the package `krb5-apps-clients`. They all have the full functionality of their common Unix and Linux brothers plus the additional bonus of transparent authentication managed by Kerberos:

- **`telnet`**, `telnetd`
- **`rlogin`**
- **`rsh`**, **`rcp`**, `rshd`
- **`ftp`**, `ftpd`
- **`ksu`**

You no longer need to enter your password for using these applications because Kerberos has already proven your identity. **`ssh`**, if compiled with Kerberos support, can even forward all the tickets acquired for one workstation to another one. If you use **`ssh`** to log in to another workstation, **`ssh`** makes sure that the encrypted contents of the tickets are adjusted to the new situation. Simply copying tickets between workstations is not sufficient because the

ticket contains workstation-specific information (the IP address). XDM and GDM offer Kerberos support, too. Read more about the Kerberos network applications in *Kerberos V5 UNIX User's Guide* at http://web.mit.edu/kerberos.

6.5 Installing and Administering Kerberos

A Kerberos environment consists of several components. A key distribution center (KDC) holds the central database with all Kerberos-relevant data. All clients rely on the KDC for proper authentication across the network. Both the KDC and the clients need to be configured to match your setup:

General Preparations

Check your network setup and make sure it meets the minimum requirements outlined in *Section 6.5.1, "Kerberos Network Topology"*. Choose an appropriate realm for your Kerberos setup, see *Section 6.5.2, "Choosing the Kerberos Realms"*. Carefully set up the machine that is to serve as the KDC and apply tight security, see *Section 6.5.3, "Setting Up the KDC Hardware"*. Set up a reliable time source in your network to make sure all tickets contain valid time stamps, see *Section 6.5.4, "Configuring Time Synchronization"*.

Basic Configuration

Configure the KDC and the clients, see *Section 6.5.5, "Configuring the KDC"* and *Section 6.5.6, "Configuring Kerberos Clients"*. Enable remote administration for your Kerberos service, so you do not need physical access to your KDC machine, see *Section 6.5.7, "Configuring Remote Kerberos Administration"*. Create service principals for every service in your realm, see *Section 6.5.8, "Creating Kerberos Service Principals"*.

Enabling Kerberos Authentication

Various services in your network can use Kerberos. To add Kerberos password-checking to applications using PAM, proceed as outlined in *Section 6.5.9, "Enabling PAM Support for Kerberos"*. To configure SSH or LDAP with Kerberos authentication, proceed as outlined in *Section 6.5.10, "Configuring SSH for Kerberos Authentication"* and *Section 6.5.11, "Using LDAP and Kerberos"*.

6.5.1 Kerberos Network Topology

Any Kerberos environment must meet the following requirements to be fully functional:

- Provide a DNS server for name resolution across your network, so clients and servers can locate each other. Refer to *Book "Reference", Chapter 19 "The Domain Name System"* for information on DNS setup.
- Provide a time server in your network. Using exact time stamps is crucial to a Kerberos setup, because valid Kerberos tickets must contain correct time stamps. Refer to *Book "Reference", Chapter 18 "Time Synchronization with NTP"* for information on NTP setup.
- Provide a key distribution center (KDC) as the center piece of the Kerberos architecture. It holds the Kerberos database. Use the tightest possible security policy on this machine to prevent any attacks on this machine compromising your entire infrastructure.
- Configure the client machines to use Kerberos authentication.

The following figure depicts a simple example network with only the minimum components needed to build a Kerberos infrastructure. Depending on the size and topology of your deployment, your setup may vary.

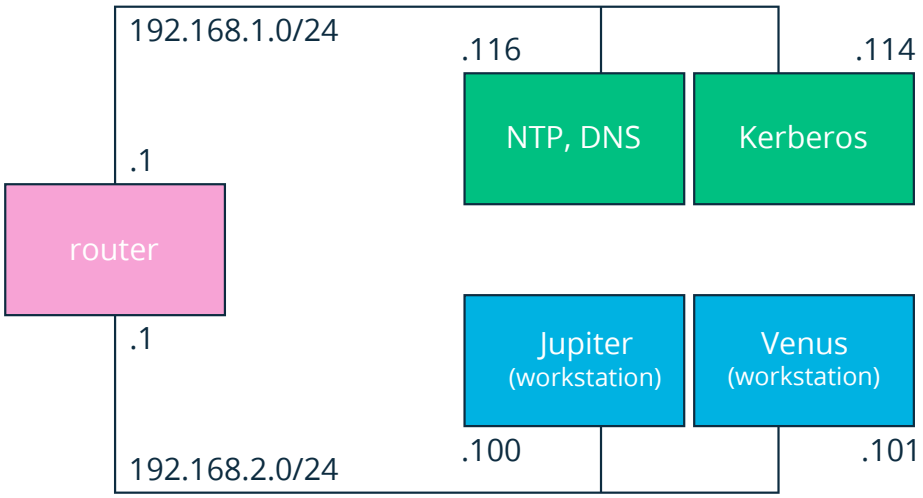


FIGURE 6.1: KERBEROS NETWORK TOPOLOGY

Tip: Configuring Subnet Routing

For a setup similar to the one in *Figure 6.1, "Kerberos Network Topology"*, configure routing between the two subnets (192.168.1.0/24 and 192.168.2.0/24). Refer to *Book "Reference", Chapter 13 "Basic Networking", Section 13.4.1.5 "Configuring Routing"* for more information on configuring routing with YaST.

6.5.2 Choosing the Kerberos Realms

The domain of a Kerberos installation is called a realm and is identified by a name, such as `EXAMPLE.COM` or simply `ACCOUNTING`. Kerberos is case-sensitive, so `example.com` is actually a different realm than `EXAMPLE.COM`. Use the case you prefer. It is common practice, however, to use uppercase realm names.

It is also a good idea to use your DNS domain name (or a subdomain, such as `ACCOUNTING.EXAMPLE.COM`). As shown below, your life as an administrator can be much easier if you configure your Kerberos clients to locate the KDC and other Kerberos services via DNS. To do so, it is helpful if your realm name is a subdomain of your DNS domain name.

Unlike the DNS name space, Kerberos is not hierarchical. So if you have a realm named `EXAMPLE.COM` with two "subrealms" named `DEVELOPMENT` and `ACCOUNTING`, these subordinate realms do not inherit principals from `EXAMPLE.COM`. Instead, you would have three separate realms, and you would need to configure cross-realm authentication for each realm, so that users from one realm can interact with servers or other users from another realm.

For the sake of simplicity, let us assume you are setting up only one realm for your entire organization. For the remainder of this section, the realm name `EXAMPLE.COM` is used in all examples.

6.5.3 Setting Up the KDC Hardware

The first thing required to use Kerberos is a machine that acts as the key distribution center, or KDC for short. This machine holds the entire Kerberos user database with passwords and all information.

The KDC is the most important part of your security infrastructure—if someone breaks into it, all user accounts and all of your infrastructure protected by Kerberos is compromised. An attacker with access to the Kerberos database can impersonate any principal in the database. Tighten security for this machine as much as possible:

1. Put the server machine into a physically secured location, such as a locked server room to which only a very few people have access.
2. Do not run any network applications on it except the KDC. This includes servers and clients —for example, the KDC should not import any file systems via NFS or use DHCP to retrieve its network configuration.
3. Install a minimal system first then check the list of installed packages and remove any unneeded packages. This includes servers, such as `inetd`, `portmap`, and CUPS, plus anything X-based. Even installing an SSH server should be considered a potential security risk.
4. No graphical login is provided on this machine as an X server is a potential security risk. Kerberos provides its own administration interface.
5. Configure `/etc/nsswitch.conf` to use only local files for user and group lookup. Change the lines for `passwd` and `group` to look like this:

   ```
   passwd:         files
   group:          files
   ```

   Edit the `passwd`, `group`, and `shadow` files in `/etc` and remove the lines that start with a `+` character (these are for NIS lookups).
6. Disable all user accounts except `root`'s account by editing `/etc/shadow` and replacing the hashed passwords with `*` or `!` characters.

6.5.4 Configuring Time Synchronization

To use Kerberos successfully, make sure that all system clocks within your organization are synchronized within a certain range. This is important because Kerberos protects against replayed credentials. An attacker might be able to observe Kerberos credentials on the network and reuse them to attack the server. Kerberos employs several defenses to prevent this. One of them is that it puts time stamps into its tickets. A server receiving a ticket with a time stamp that differs from the current time rejects the ticket.

Kerberos allows a certain leeway when comparing time stamps. However, computer clocks can be very inaccurate in keeping time—it is not unheard of for PC clocks to lose or gain half an hour during a week. For this reason, configure all hosts on the network to synchronize their clocks with a central time source.

A simple way to do so is by installing an NTP time server on one machine and having all clients synchronize their clocks with this server. Do this by running an NTP daemon `chronyd` as a client on all these machines. The KDC itself needs to be synchronized to the common time source as well. Because running an NTP daemon on this machine would be a security risk, it is probably a good idea to do this by running `chronyd -q` via a cron job. To configure your machine as an NTP client, proceed as outlined in *Book "Reference", Chapter 18 "Time Synchronization with NTP", Section 18.1 "Configuring an NTP Client with YaST"*.

A different way to secure the time service and still use the NTP daemon is to attach a hardware reference clock to a dedicated NTP server and an additional hardware reference clock to the KDC.

It is also possible to adjust the maximum deviation Kerberos allows when checking time stamps. This value (called *clock skew*) can be set in the `krb5.conf` file as described in *Section 6.5.6.3, "Adjusting the Clock Skew"*.

6.5.5 Configuring the KDC

This section covers the initial configuration and installation of the KDC, including the creation of an administrative principal. This procedure consists of several steps:

1. **Install the RPMs.** On a machine designated as the KDC, install the following software packages: `krb5`, `krb5-server` and `krb5-client` packages.
2. **Adjust the Configuration Files.** The `/etc/krb5.conf` and `/var/lib/kerberos/krb5kdc/kdc.conf` configuration files must be adjusted for your scenario. These files contain all information on the KDC.
3. **Create the Kerberos Database.** Kerberos keeps a database of all principal identifiers and the secret keys of all principals that need to be authenticated. Refer to *Section 6.5.5.1, "Setting Up the Database"* for details.
4. **Adjust the ACL Files: Add Administrators.** The Kerberos database on the KDC can be managed remotely. To prevent unauthorized principals from tampering with the database, Kerberos uses access control lists. You must explicitly enable remote access for the

administrator principal to enable them to manage the database. The Kerberos ACL file is located under `/var/lib/kerberos/krb5kdc/kadm5.acl`. Refer to *Section 6.5.7, "Configuring Remote Kerberos Administration"* for details.

5. **Adjust the Kerberos Database: Add Administrators.** You need at least one administrative principal to run and administer Kerberos. This principal must be added before starting the KDC. Refer to *Section 6.5.5.2, "Creating a Principal"* for details.

6. **Start the Kerberos Daemon.** After the KDC software is installed and properly configured, start the Kerberos daemon to provide Kerberos service for your realm. Refer to *Section 6.5.5.3, "Starting the KDC"* for details.

7. **Create a Principal for Yourself.** You need a principal for yourself. Refer to *Section 6.5.5.2, "Creating a Principal"* for details.

6.5.5.1 Setting Up the Database

Your next step is to initialize the database where Kerberos keeps all information about principals. Set up the database master key, which is used to protect the database from accidental disclosure (in particular if it is backed up to tape). The master key is derived from a pass phrase and is stored in a file called the stash file. This is so you do not need to enter the password every time the KDC is restarted. Make sure that you choose a good pass phrase, such as a sentence from a book opened to a random page.

When you make tape backups of the Kerberos database (`/var/lib/kerberos/krb5kdc/principal`), do not back up the stash file (which is in `/var/lib/kerberos/krb5kdc/.k5.EXAMPLE.COM`). Otherwise, everyone able to read the tape could also decrypt the database. Therefore, keep a copy of the pass phrase in a safe or some other secure location, because you will need it to restore your database from backup tape after a crash.

To create the stash file and the database, run:

```
tux > sudo kdb5_util create -r EXAMPLE.COM -s
```

You will see the following output:

```
Initializing database '/var/lib/kerberos/krb5kdc/principal' for realm 'EXAMPLE.COM',
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:  ❶
```

```
Re-enter KDC database master key to verify:  ❷
```

❶ Type the master password.

❷ Type the password again.

To verify, use the list command:

```
tux > kadmin.local

kadmin> listprincs
```

You will see several principals in the database, which are for internal use by Kerberos:

```
K/M@EXAMPLE.COM
kadmin/admin@EXAMPLE.COM
kadmin/changepw@EXAMPLE.COM
krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

6.5.5.2 Creating a Principal

Create two Kerberos principals for yourself: one normal principal for everyday work and one for administrative tasks relating to Kerberos. Assuming your login name is `geeko`, proceed as follows:

```
tux > kadmin.local

kadmin> ank geeko
```

You will see the following output:

```
geeko@EXAMPLE.COM's Password: ❶
Verifying password: ❷
```

❶ Type `geeko`'s password.

❶ Type `geeko`'s password again.

Next, create another principal named `geeko/admin` by typing **`ank`** `geeko/admin` at the **`kadmin`** prompt. The `admin` suffixed to your user name is a *role*. Later, use this role when administering the Kerberos database. A user can have several roles for different purposes. Roles act like completely different accounts that have similar names.

6.5.5.3 Starting the KDC

Start the KDC daemon and the kadmin daemon. To start the daemons manually, enter:

```
tux > sudo systemctl start krb5kdc
sudo systemctl start kadmind
```

Also make sure that the services KDC (`krb5kdc`) and kadmind (`kadmind`) are started by default when the server machine is rebooted. Enable them by entering:

```
tux > sudo systemctl enable krb5kdc kadmind
```

or by using the YaST *Services Manager*.

6.5.6 Configuring Kerberos Clients

When the supporting infrastructure is in place (DNS, NTP) and the KDC has been properly configured and started, configure the client machines. To configure a Kerberos client, use one of the two manual approaches described below.

When configuring Kerberos, there are two approaches you can take—static configuration in the `/etc/krb5.conf` file or dynamic configuration with DNS. With DNS configuration, Kerberos applications try to locate the KDC services using DNS records. With static configuration, add the host names of your KDC server to `krb5.conf` (and update the file whenever you move the KDC or reconfigure your realm in other ways).

DNS-based configuration is generally a lot more flexible and the amount of configuration work per machine is a lot less. However, it requires that your realm name is either the same as your DNS domain or a subdomain of it. Configuring Kerberos via DNS also creates a security issue: An attacker can seriously disrupt your infrastructure through your DNS (by shooting down the name server, spoofing DNS records, etc.). However, this amounts to a denial of service at worst. A similar scenario applies to the static configuration case unless you enter IP addresses in `krb5.conf` instead of host names.

6.5.6.1 Static Configuration

One way to configure Kerberos is to edit `/etc/krb5.conf`. The file installed by default contains various sample entries. Erase all of these entries before starting. `krb5.conf` is made up of several sections (stanzas), each introduced by the section name in brackets like `[this]`.

To configure your Kerberos clients, add the following stanza to `krb5.conf` (where `kdc.example.com` is the host name of the KDC):

```
[libdefaults]
        default_realm = EXAMPLE.COM

[realms]
        EXAMPLE.COM = {
                kdc = kdc.example.com
                admin_server = kdc.example.com
        }
```

The `default_realm` line sets the default realm for Kerberos applications. If you have several realms, add additional statements to the `[realms]` section.

Also add a statement to this file that tells applications how to map host names to a realm. For example, when connecting to a remote host, the Kerberos library needs to know in which realm this host is located. This must be configured in the `[domain_realms]` section:

```
[domain_realm]
.example.com = EXAMPLE.COM
www.example.org = EXAMPLE.COM
```

This tells the library that all hosts in the `example.com` DNS domains are in the `EXAMPLE.COM` Kerberos realm. In addition, one external host named `www.example.org` should also be considered a member of the `EXAMPLE.COM` realm.

6.5.6.2 DNS-Based Configuration

DNS-based Kerberos configuration makes heavy use of SRV records. See *(RFC2052) A DNS RR for specifying the location of services* at http://www.ietf.org.

The name of an SRV record, as far as Kerberos is concerned, is always in the format `_service._proto.realm`, where realm is the Kerberos realm. Domain names in DNS are case-insensitive, so case-sensitive Kerberos realms would break when using this configuration method. `_service` is a service name (different names are used when trying to contact the KDC or the password service, for example). `_proto` can be either `_udp` or `_tcp`, but not all services support both protocols.

The data portion of SRV resource records consists of a priority value, a weight, a port number, and a host name. The priority defines the order in which hosts should be tried (lower values indicate a higher priority). The weight value is there to support some sort of load balancing among servers of equal priority. You probably do not need any of this, so it is okay to set these to zero.

MIT Kerberos currently looks up the following names when looking for services:

_kerberos

This defines the location of the KDC daemon (the authentication and ticket granting server). Typical records look like this:

```
_kerberos._udp.EXAMPLE.COM.  IN  SRV    0 0 88 kdc.example.com.
_kerberos._tcp.EXAMPLE.COM.  IN  SRV    0 0 88 kdc.example.com.
```

_kerberos-adm

This describes the location of the remote administration service. Typical records look like this:

```
_kerberos-adm._tcp.EXAMPLE.COM. IN  SRV    0 0 749 kdc.example.com.
```

Because kadmind does not support UDP, there should be no `_udp` record.

As with the static configuration file, there is a mechanism to inform clients that a specific host is in the `EXAMPLE.COM` realm, even if it is not part of the `example.com` DNS domain. This can be done by attaching a TXT record to `_kerberos.host_name`, as shown here:

```
_kerberos.www.example.org.  IN TXT "EXAMPLE.COM"
```

6.5.6.3 Adjusting the Clock Skew

The *clock skew* is the tolerance for accepting tickets with time stamps that do not exactly match the host's system clock. Usually, the clock skew is set to 300 seconds (five minutes). This means a ticket can have a time stamp somewhere between five minutes behind and five minutes ahead of the server's clock.

When using NTP to synchronize all hosts, you can reduce this value to about one minute. The clock skew value can be set in `/etc/krb5.conf` like this:

```
[libdefaults]
```

```
        clockskew = 60
```

6.5.7 Configuring Remote Kerberos Administration

To be able to add and remove principals from the Kerberos database without accessing the KDC's console directly, tell the Kerberos administration server which principals are allowed to do what by editing `/var/lib/kerberos/krb5kdc/kadm5.acl`. The ACL (access control list) file allows you to specify privileges with a precise degree of control. For details, refer to the manual page with **`man`** `8 kadmind`.

For now, grant yourself the privilege to administer the database by putting the following line into the file:

```
geeko/admin              *
```

Replace the user name `geeko` with your own. Restart `kadmind` for the change to take effect.

You should now be able to perform Kerberos administration tasks remotely using the kadmin tool. First, obtain a ticket for your admin role and use that ticket when connecting to the kadmin server:

```
tux > kadmin -p geeko/admin
Authenticating as principal geeko/admin@EXAMPLE.COM with password.
Password for geeko/admin@EXAMPLE.COM:
kadmin:  getprivs
current privileges: GET ADD MODIFY DELETE
kadmin:
```

Using the **`getprivs`** command, verify which privileges you have. The list shown above is the full set of privileges.

As an example, modify the principal `geeko`:

```
tux > kadmin -p geeko/admin
Authenticating as principal geeko/admin@EXAMPLE.COM with password.
Password for geeko/admin@EXAMPLE.COM:

kadmin:  getprinc geeko
Principal: geeko@EXAMPLE.COM
Expiration date: [never]
Last password change: Wed Jan 12 17:28:46 CET 2005
Password expiration date: [none]
Maximum ticket life: 0 days 10:00:00
Maximum renewable life: 7 days 00:00:00
```

```
Last modified: Wed Jan 12 17:47:17 CET 2005 (admin/admin@EXAMPLE.COM)
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 2
Key: vno 1, Triple DES cbc mode with HMAC/sha1, no salt
Key: vno 1, DES cbc mode with CRC-32, no salt
Attributes:
Policy: [none]

kadmin:  modify_principal -maxlife "8 hours" geeko
Principal "geeko@EXAMPLE.COM" modified.
kadmin:  getprinc geeko
Principal: geeko@EXAMPLE.COM
Expiration date: [never]
Last password change: Wed Jan 12 17:28:46 CET 2005
Password expiration date: [none]
Maximum ticket life: 0 days 08:00:00
Maximum renewable life: 7 days 00:00:00
Last modified: Wed Jan 12 17:59:49 CET 2005 (geeko/admin@EXAMPLE.COM)
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 2
Key: vno 1, Triple DES cbc mode with HMAC/sha1, no salt
Key: vno 1, DES cbc mode with CRC-32, no salt
Attributes:
Policy: [none]
kadmin:
```

This changes the maximum ticket life time to eight hours. For more information about the **`kadmin`** command and the options available, see the `krb5-doc` package or refer to the **`man`** `8 kadmin` manual page.

6.5.8 Creating Kerberos Service Principals

So far, only user credentials have been discussed. However, Kerberos-compatible services usually need to authenticate themselves to the client user, too. Therefore, special service principals must be in the Kerberos database for each service offered in the realm. For example, if ldap.example.com offers an LDAP service, you need a service principal, `ldap/ldap.example.com@EXAMPLE.COM`, to authenticate this service to all clients.

The naming convention for service principals is *`SERVICE/HOSTNAME@REALM`*, where *`HOSTNAME`* is the host's fully qualified host name.

Valid service descriptors are:

Service Descriptor	Service
`host`	Telnet, RSH, SSH
`nfs`	NFSv4 (with Kerberos support)
`HTTP`	HTTP (with Kerberos authentication)
`imap`	IMAP
`pop`	POP3
`ldap`	LDAP

Service principals are similar to user principals, but have significant differences. The main difference between a user principal and a service principal is that the key of the former is protected by a password. When a user obtains a ticket-granting ticket from the KDC, they needs to type their password, so Kerberos can decrypt the ticket. It would be inconvenient for system administrators to obtain new tickets for the SSH daemon every eight hours or so.

Instead, the key required to decrypt the initial ticket for the service principal is extracted by the administrator from the KDC only once and stored in a local file called the *keytab*. Services such as the SSH daemon read this key and use it to obtain new tickets automatically, when needed. The default keytab file resides in `/etc/krb5.keytab`.

To create a host service principal for `jupiter.example.com` enter the following commands during your kadmin session:

```
tux > kadmin -p geeko/admin
Authenticating as principal geeko/admin@EXAMPLE.COM with password.
Password for geeko/admin@EXAMPLE.COM:
kadmin:  addprinc -randkey host/jupiter.example.com
WARNING: no policy specified for host/jupiter.example.com@EXAMPLE.COM;
defaulting to no policy
Principal "host/jupiter.example.com@EXAMPLE.COM" created.
```

Instead of setting a password for the new principal, the `-randkey` flag tells **`kadmin`** to generate a random key. This is used here because no user interaction is wanted for this principal. It is a server account for the machine.

Finally, extract the key and store it in the local keytab file `/etc/krb5.keytab`. This file is owned by the superuser, so you must be `root` to execute the next command in the kadmin shell:

```
kadmin:  ktadd host/jupiter.example.com
Entry for principal host/jupiter.example.com with kvno 3, encryption type Triple
DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/jupiter.example.com with kvno 3, encryption type DES
cbc mode with CRC-32 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:
```

When completed, make sure that you destroy the admin ticket obtained with kinit above with **`kdestroy`**.

6.5.9 Enabling PAM Support for Kerberos

Warning: Incomplete Configuration Locks Users Out

An incomplete Kerberos configuration may completely lock you out of your system, including the root user. To prevent this, add the `ignore_unknown_principals` directive to the `pam_krb5` module *after* you have added the `pam_krb5` module to the existing PAM configuration files as described below.

```
tux > sudo pam-config --add --krb5-ignore_unknown_principals
```

This will direct the `pam_krb5` module to ignore some errors that would otherwise cause the account phase to fail.

openSUSE® Leap comes with a PAM module named `pam_krb5`, which supports Kerberos login and password update. This module can be used by applications such as console login, **`su`**, and graphical login applications like GDM. That is, it can be used in all cases where the user enters a password and expects the authenticating application to obtain an initial Kerberos ticket on their behalf. To configure PAM support for Kerberos, use the following command:

```
tux > sudo pam-config --add --krb5
```

The above command adds the `pam_krb5` module to the existing PAM configuration files and makes sure it is called in the right order. To make precise adjustments to the way in which `pam_krb5` is used, edit the file `/etc/krb5.conf` and add default applications to PAM. For details, refer to the manual page with **`man 5 pam_krb5`**.

The `pam_krb5` module was specifically not designed for network services that accept Kerberos tickets as part of user authentication. This is an entirely different matter, and is discussed below.

6.5.10 Configuring SSH for Kerberos Authentication

OpenSSH supports Kerberos authentication in both protocol version 1 and 2. In version 1, there are special protocol messages to transmit Kerberos tickets. Version 2 does not use Kerberos directly anymore, but relies on GSSAPI, the General Security Services API. This is a programming interface that is not specific to Kerberos—it was designed to hide the peculiarities of the underlying authentication system, be it Kerberos, a public-key authentication system like SPKM, or others. However, the included GSSAPI library only supports Kerberos.

To use sshd with Kerberos authentication, edit `/etc/ssh/sshd_config` and set the following options:

```
# These are for protocol version 1
#
# KerberosAuthentication yes
# KerberosTicketCleanup yes

# These are for version 2 - better to use this
GSSAPIAuthentication yes
GSSAPICleanupCredentials yes
```

Then restart your SSH daemon using **`sudo systemctl restart sshd`**.

To use Kerberos authentication with protocol version 2, enable it on the client side as well. Do this either in the system wide configuration file `/etc/ssh/ssh_config` or on a per-user level by editing `~/.ssh/config`. In both cases, add the option `GSSAPIAuthentication yes`.

You should now be able to connect using Kerberos authentication. Use **`klist`** to verify that you have a valid ticket, then connect to the SSH server. To force SSH protocol version 1, specify the `-1` option on the command line.

Tip: Additional Information

The file `/usr/share/doc/packages/openssh/README.kerberos` discusses the interaction of OpenSSH and Kerberos in more detail.

Tip: Additional Directives for Protocol Version 2

The `GSSAPIKeyExchange` mechanism (RFC 4462) is supported. This directive specifies how host keys are exchanged. For more information, see the sshd_config manual page (`man sshd_config`).

6.5.11 Using LDAP and Kerberos

When using Kerberos, one way to distribute the user information (such as user ID, groups, and home directory) in your local network is to use LDAP. This requires a strong authentication mechanism that prevents packet spoofing and other attacks. One solution is to use Kerberos for LDAP communication, too.

OpenLDAP implements most authentication flavors through SASL, the simple authentication session layer. SASL is a network protocol designed for authentication. The SASL implementation is cyrus-sasl, which supports several authentication flavors. Kerberos authentication is performed through GSSAPI (General Security Services API). By default, the SASL plug-in for GSSAPI is not installed. Install the `cyrus-sasl-gssapi` with YaST.

To enable Kerberos to bind to the OpenLDAP server, create a principal `ldap/ldap.example.com` and add that to the keytab.

By default, the LDAP server slapd runs as user and group `ldap`, while the keytab file is readable by `root` only. Therefore, either change the LDAP configuration so the server runs as `root` or make the keytab file readable by the group `ldap`. The latter is done automatically by the OpenLDAP start script (`/usr/lib/openldap/start`) if the keytab file has been specified in the `OPENLDAP_KRB5_KEYTAB` variable in `/etc/sysconfig/openldap` and the `OPENLDAP_CHOWN_DIRS` variable is set to `yes`, which is the default setting. If `OPENLDAP_KRB5_KEYTAB` is left empty, the default keytab under `/etc/krb5.keytab` is used and you must adjust the privileges yourself as described below.

To run slapd as `root`, edit `/etc/sysconfig/openldap`. Disable the `OPENLDAP_USER` and `OPENLDAP_GROUP` variables by putting a comment character in front of them.

To make the keytab file readable by group LDAP, execute

```
tux > sudo chgrp ldap /etc/krb5.keytab
tux > sudo chmod 640 /etc/krb5.keytab
```

A third (and maybe the best) solution is to tell OpenLDAP to use a special keytab file. To do this, start kadmin, and enter the following command after you have added the principal ldap/ldap.example.com:

```
tux > sudo ktadd -k /etc/openldap/ldap.keytab ldap/ldap.example.com@EXAMPLE.COM
```

Then in the shell run:

```
tux > sudo chown ldap:ldap /etc/openldap/ldap.keytab
tux > sudo chmod 600 /etc/openldap/ldap.keytab
```

To tell OpenLDAP to use a different keytab file, change the following variable in `/etc/sysconfig/openldap`:

```
OPENLDAP_KRB5_KEYTAB="/etc/openldap/ldap.keytab"
```

Finally, restart the LDAP server using **`sudo systemctl restart slapd`**.

6.5.11.1 Using Kerberos Authentication with LDAP

You are now able to automatically use tools such as ldapsearch with Kerberos authentication.

```
tux > ldapsearch -b ou=people,dc=example,dc=com '(uid=geeko)'

SASL/GSSAPI authentication started
SASL SSF: 56
SASL installing layers
[...]

# geeko, people, example.com
dn: uid=geeko,ou=people,dc=example,dc=com
uid: geeko
cn: Suzanne Geeko
[...]
```

As you can see, **`ldapsearch`** prints a message that it started GSSAPI authentication. The next message is very cryptic, but it shows that the *security strength factor* (SSF for short) is 56 (The value 56 is somewhat arbitrary. Most likely it was chosen because this is the number of bits in a DES encryption key). This means that GSSAPI authentication was successful and that encryption is being used to protect integrity and provide confidentiality for the LDAP connection.

In Kerberos, authentication is always mutual. This means that not only have you authenticated yourself to the LDAP server, but also the LDAP server has authenticated itself to you. In particular, this means communication is with the desired LDAP server, rather than some bogus service set up by an attacker.

6.5.11.2 Kerberos Authentication and LDAP Access Control

There is one minor piece of the puzzle missing—how the LDAP server can find out that the Kerberos user `tux@EXAMPLE.COM` corresponds to the LDAP distinguished name `uid=tux,ou=people,dc=example,dc=com`. This sort of mapping must be configured manually using the `saslExpr` directive. In this example, the "authz-regexp" change in LDIF would look as follows:

```
dn: cn=config
add: olcAuthzRegexp
olcAuthzRegexp: uid=(.*),cn=GSSAPI,cn=auth uid=$1,ou=people,dc=example,dc=com
```

All these changes can be applied via **`ldapmodify`** on the command line.

When SASL authenticates a user, OpenLDAP forms a distinguished name from the name given to it by SASL (such as `tux`) and the name of the SASL flavor (`GSSAPI`). The result would be `uid=tux,cn=GSSAPI,cn=auth`.

If a `authz-regexp` has been configured, it checks the DN formed from the SASL information using the first argument as a regular expression. If this regular expression matches, the name is replaced with the second argument of the `authz-regexp` statement. The placeholder `$1` is replaced with the substring matched by the `(.*)` expression.

More complicated match expressions are possible. If you have a more complicated directory structure or a schema in which the user name is not part of the DN, you can even use search expressions to map the SASL DN to the user DN.

For more information, see the `slapd-config` man page.

6.6 Setting up Kerberos using *LDAP and Kerberos Client*

YaST includes the module *LDAP and Kerberos Client* that helps define authentication scenarios involving either LDAP or Kerberos.

It can also be used to join Kerberos and LDAP separately. However, in many such cases, using this module may not be the first choice, such as for joining Active Directory (which uses a combination of LDAP and Kerberos). For more information, see *Section 4.2, "Configuring an Authentication Client with YaST"*.

Start the module by selecting *Network Services* › *LDAP and Kerberos Client*.

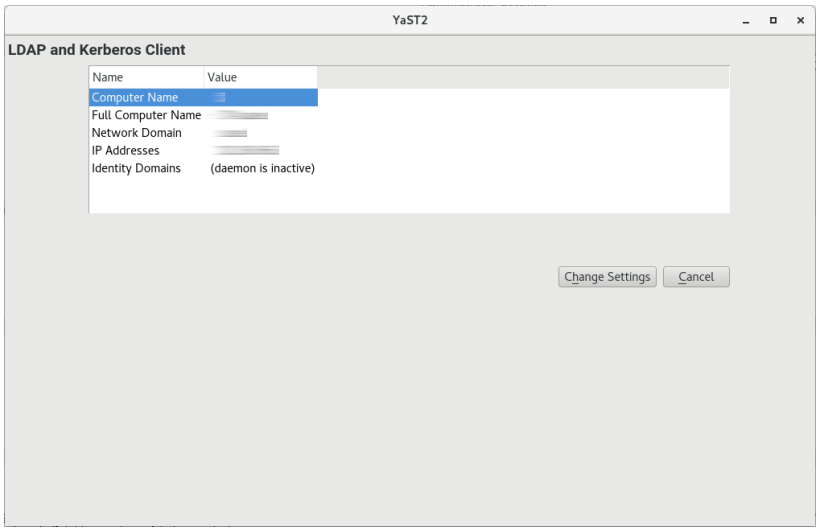
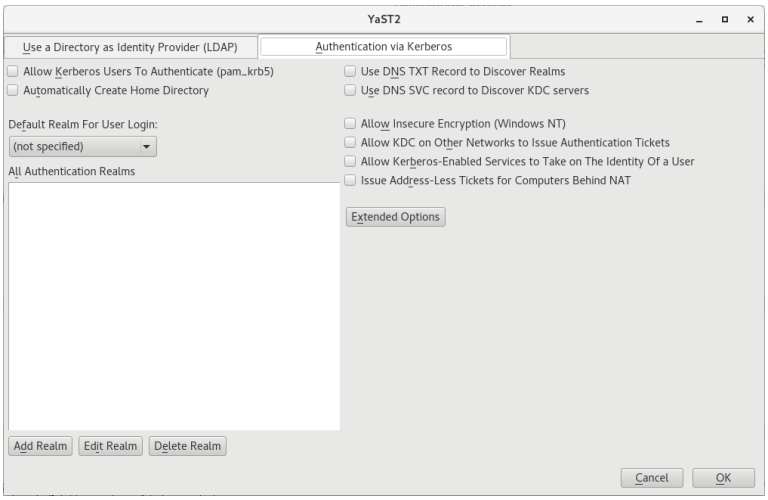


FIGURE 6.2: *LDAP AND KERBEROS CLIENT* WINDOW

To configure a Kerberos client, follow the procedure below:

1. In the window *LDAP and Kerberos Client*, click *Change Settings*. Choose the tab *Authentication via Kerberos*.



2. Click *Add Realm*.

3. In the appearing dialog, specify the correct *Realm name*. Usually, the realm name is an uppercase version of the domain name. Additionally, you can specify the following:

   - To apply mappings from the realm name to the domain name, activate *Map Domain Name to the Realm* and/or *Map Wildcard Domain Name to the Realm*.

   - You can specify the *Host Name of Administration Server*, the *Host Name of Master Key Distribution Server* and additional *Key Distribution Centers*.
     All of these items are optional if they can be automatically discovered via the `SRV` and `TXT` records in DNS.

   - To manually map Principals to local user names, use *Custom Mappings of Principal Names to User Names*.
     You can also use `auth_to_local` rules to supply such mappings using *Custom Rules for Mapping Principal Names to User Names*. For more information about using such rules, see the official documentation at https://web.mit.edu/kerberos/krb5-current/doc/admin/conf_files/krb5_conf.html#realms and an article at https://community.hortonworks.com/articles/14463/auth-to-local-rules-syntax.html.

   Continue with *OK*.

4. To add more realms, repeat from *Step 2*.

5. Enable Kerberos users logging in and creation of home directories by activating *Allow Kerberos Users to Authenticate* and *Automatically Create Home Directory*.

6. If you left empty the optional text boxes in *Step 3*, make sure to enable automatic discovery of realms and key distribution centers by activating *Use DNS TXT Record to Discover Realms* and *Use DNS SRV Record to Discover KDC Servers*.

7. You can additionally activate the following:

   - *Allow Insecure Encryption (for Windows NT)* allows the encryption types listed as weak at http://web.mit.edu/kerberos/krb5-current/doc/admin/conf_files/kdc_conf.html#encryption-types.

   - *Allow KDC on Other Networks to Issue Authentication Tickets* allows forwarding of tickets.

- *Allow Kerberos-Enabled Services to Take on The Identity Of a User* allows the use of proxies between the computer of the user and the key distribution center.
- *Issue Address-Less Tickets for Computers Behind NAT* allows granting tickets to users behind networks using network address translation.

8. To set up allowed encryption types and define the name of the keytab file which lists the names of principals and their encrypted keys, use the *Extended Options*.
9. Finish with *OK* and *Finish*.
   YaST may now install extra packages.

6.7 Kerberos and NFS

Most NFS servers can export file systems using any combination of the default "trust the network" form of security, known as `sec=sys`, and three different levels of Kerberos-based security, `sec=krb5`, `sec=krb5i`, and `sec=krb5p`. The `sec` option is set as a mount option on the client. It is often the case that the NFS service will first be configured and used with `sec=sys`, and then Kerberos can be imposed afterwards. In this case it is likely that the server will be configured to support both `sec=sys` and one of the Kerberos levels, and then after all clients have transitioned, the `sec=sys` support will be removed, thus achieving true security. The transition to Kerberos should be fairly transparent if done in an orderly manner. However there is one subtle detail of NFS behavior that works differently when Kerberos is used, and the implications of this need to be understood and possibly addressed. See *Section 6.7.1, "Group Membership"*.

The three Kerberos levels indicate different levels of security. With more security comes a need for more processor power to encrypt and decrypt messages. Choosing the right balance is an important consideration when planning a roll-out of Kerberos for NFS.

`krb5` provides only authentication. The server can know who sent a request, and the client can know that the server sent a reply. No security is provided for the content of the request or reply, so an attacker with physical network access could transform the request or reply, or both, in various ways to deceive either server or client. They cannot directly read or change any file that the authenticated user could not read or change, but almost anything is theoretically possible.

`krb5i` adds integrity checks to all messages. With `krb5i`, an attacker cannot modify any request or reply, but they can view all the data exchanged, and so could discover the content of any file that is read.

`krb5p` adds privacy to the protocol. As well as reliable authentication and integrity checking, messages are fully encrypted so an attacker can only know that messages were exchanged between client and server, and cannot extract other information directly from the message. Whether information can be extracted from message timing is a separate question that Kerberos does not address.

6.7.1 Group Membership

The one behavioral difference between `sec=sys` and the various Kerberos security levels that might be visible is related to group membership. In Unix and Linux, each file system access comes from a process that is owned by a particular user and has a particular group owner and a number of supplemental groups. Access rights to files can vary based on the owner and the various groups.

With `sec=sys`, the user-id, group-id, and a list of up to 16 supplementary groups are sent to the server in each request.

If a user is a member of more than 16 supplementary groups, the extra groups are lost and some files may not be accessible over NFS that the user would normally expect to have access to. For this reason, most sites that use NFS find a way to limit all users to at most 16 supplementary groups.

If the user runs the **`newgrp`** command or runs a set-group-id program, either of which can change the list of groups they are a member of, these changes take effect immediately and provide different accesses over NFS.

With Kerberos, group information is not sent in requests at all. Only the user is identified (using a Kerberos “principal”), and the server performs a lookup to determine the user ID and group list for that principal. This means that if the user is a member of more than 16 groups, all of these group memberships will be used in determining file access permissions. However it also means that if the user changes a group-id on the client in some way, the server will not notice the change and will not take it into account in determining access rights.

In most cases, the improvement of having access to more groups brings a real benefit, and the loss of not being able to change groups is not noticed as it is not widely used. A site administrator considering the use of Kerberos should be aware of the difference though and ensure that it will not actually cause problems.

6.7.2 Performance and Scalability

Using Kerberos for security requires extra CPU power for encrypting and decrypting messages. How much extra CPU power is required and whether the difference is noticeable will vary with different hardware and different applications. If the server or client are already saturating the available CPU power, it is likely that a performance drop will be measurable when switching from `sec=sys` to Kerberos. If there is spare CPU capacity available, it is quite possible that the transition will not result in any throughput change. The only way to be sure how much impact the use of Kerberos will have is to test your load on your hardware.

The only configuration options that might reduce the load will also reduce the quality of the protection offered. `sec=krb5` should produce noticeably less load than `sec=krb5p` but, as discussed above, it doesn't produce very strong security. Similarly it is possible to adjust the list of ciphers that Kerberos can choose from, and this might change the CPU requirement. However the defaults are carefully chosen and should not be changed without similar careful consideration.

The other possible performance issue when configuring NFS to use Kerberos involves availability of the Kerberos authentication servers, known as the KDC or Key Distribution Center.

The use of NFS adds load to such servers to the same degree that adding the use of Kerberos for any other services adds some load. Every time a given user (Kerberos principal) establishes a session with a service, for example by accessing files exported by a particular NFS server, the client needs to negotiate with the KDC. Once a session key has been negotiated, the client server can communicate without further help for many hours, depending on details of the Kerberos configuration, particularly the `ticket_lifetime` setting.

The concerns most likely to affect the provisioning of Kerberos KDC servers are availability and peak usage.

As with other core services such as DNS, LDAP or similar name-lookup services, having two servers that are reasonably "close" to every client provides good availability for modest resources. Kerberos allows for multiple KDC servers with flexible models for database propagation, so distributing servers as needed around campuses, buildings, and even cabinets is fairly straightforward. The best mechanism to ensure each client finds a nearby Kerberos server is to use split-horizon DNS with each building (or similar) getting different details from the DNS server. If this is not possible, then managing the `/etc/krb5.conf` file to be different at different locations is a suitable alternative.

As access to the Kerberos KDC is infrequent, load is only likely to be a problem at peak times. If thousands of people all log in between 9:00 and 9:05, then the servers will receive many more requests-per-minute than they might in the middle of the night. The load on the Kerberos server is likely to be more than that on an LDAP server, but not orders of magnitude more. A sensible guideline is to provision Kerberos replicas in the same manner that you provision LDAP replicas, and then monitor performance to determine if demand ever exceeds capacity.

6.7.3 Master KDC, Multiple Domains, and Trust Relationships

One service of the Kerberos KDC that is not easily distributed is the handling of updates, such as password changes and new user creation. These must happen at a single master KDC.

These updates are not likely to happen with such frequency that any significant load will be generated, but availability could be an issue. It can be annoying if you want to create a new user or change a password, and the master KDC on the other side of the world is temporarily unavailable.

When an organization is geographically distributed and has a policy of handling administration tasks locally at each site, it can be beneficial to create multiple Kerberos domains, one for each administrative center. Each domain would then have its own master KDC which would be geographically local. Users in one domain can still get access to resources in another domain by setting up trust relationships between domains.

The easiest arrangement for multiple domains is to have a global domain (for example EXAMPLE.COM) and various local domains (for example ASIA.EXAMPLE.COM, EUROPE.EXAMPLE.COM, etc). If the global domain is configured to trust each local domain, and each local domain is configured to trust the global domain, then fully transitive trust is available between any pair of domains, and any principal can establish a secure connection with any service. Ensuring appropriate access rights to resources, for example files provided by that service, will be dependent on the user name lookup service used, and the functionality of the NFS file server, and is beyond the scope of this document.

6.8 For More Information

The official site of MIT Kerberos is http://web.mit.edu/kerberos. There, find links to any other relevant resource concerning Kerberos, including Kerberos installation, user, and administration guides.

The book *Kerberos—A Network Authentication System* by Brian Tung (ISBN 0-201-37924-4) offers extensive information.

7 Active Directory Support

Active Directory* (AD) is a directory-service based on LDAP, Kerberos, and other services. It is used by Microsoft* Windows* to manage resources, services, and people. In a Microsoft Windows network, Active Directory provides information about these objects, restricts access to them, and enforces policies. openSUSE® Leap lets you join existing Active Directory domains and integrate your Linux machine into a Windows environment.

7.1 Integrating Linux and Active Directory Environments

With a Linux client (configured as an Active Directory client) that is joined to an existing Active Directory domain, benefit from various features not available on a pure openSUSE Leap Linux client:

Browsing Shared Files and Directories with SMB

GNOME Files (previously called Nautilus) supports browsing shared resources through SMB.

Sharing Files and Directories with SMB

GNOME Files supports sharing directories and files as in Windows.

Accessing and Manipulating User Data on the Windows Server

Through GNOME Files, users can access their Windows user data and can edit, create, and delete files and directories on the Windows server. Users can access their data without having to enter their password multiple times.

Offline Authentication

Users can log in and access their local data on the Linux machine even if they are offline or the Active Directory server is unavailable for other reasons.

Windows Password Change

This port of Active Directory support in Linux enforces corporate password policies stored in Active Directory. The display managers and console support password change messages and accept your input. You can even use the Linux **`passwd`** command to set Windows passwords.

Single-Sign-On through Kerberized Applications

Many desktop applications are Kerberos-enabled (*kerberized*), which means they can transparently handle authentication for the user without the need for password reentry at Web servers, proxies, groupware applications, or other locations.

Note: Managing Unix Attributes from Windows Server* 2016 and Later

In Windows Server 2016 and later, Microsoft has removed the role *IDMU/NIS Server* and along with it the *Unix Attributes* plug-in for the *Active Directory Users and Computers* MMC snap-in.

However, Unix attributes can still be managed manually when *Advanced Options* are enabled in the *Active Directory Users and Computers* MMC snap-in. For more information, see Clarification regarding the status of Identity Management for Unix (IDMU) & NIS Server Role in Windows Server 2016 Technical Preview and beyond (https://blogs.technet.microsoft.com/activedirectoryua/2016/02/09/identity-management-for-unix-idmu-is-deprecated-in-windows-server/).

Alternatively, use the method described in *Procedure 7.1, "Joining an Active Directory Domain Using User Logon Management"* to complete attributes on the client side (in particular, see *Step 6.c*).

The following section contains technical background for most of the previously named features. For more information about file and printer sharing using Active Directory, see *Book "*GNOME User Guide*"*.

7.2 Background Information for Linux Active Directory Support

Many system components need to interact flawlessly to integrate a Linux client into an existing Windows Active Directory domain. The following sections focus on the underlying processes of the key events in Active Directory server and client interaction.

To communicate with the directory service, the client needs to share at least two protocols with the server:

LDAP

LDAP is a protocol optimized for managing directory information. A Windows domain controller with Active Directory can use the LDAP protocol to exchange directory information with the clients. To learn more about LDAP in general and about the open source port of it, OpenLDAP, refer to *Chapter 5, LDAP—A Directory Service*.

Kerberos

Kerberos is a third-party trusted authentication service. All its clients trust Kerberos's authorization of another client's identity, enabling kerberized single-sign-on (SSO) solutions. Windows supports a Kerberos implementation, making Kerberos SSO possible even with Linux clients. To learn more about Kerberos in Linux, refer to *Chapter 6, Network Authentication with Kerberos*.

Depending on which YaST module you use to set up Kerberos authentication, different client components process account and authentication data:

Solutions Based on SSSD

- The `sssd` daemon is the central part of this solution. It handles all communication with the Active Directory server.
- To gather name service information, `sssd_nss` is used.
- To authenticate users, the `pam_sss` module for PAM is used. The creation of user homes for the Active Directory users on the Linux client is handled by `pam_mkhomedir`.
  For more information about PAM, see *Chapter 2, Authentication with PAM*.

Solution Based On Winbind (Samba)

- The `winbindd` daemon is the central part of this solution. It handles all communication with the Active Directory server.
- To gather name service information, `nss_winbind` is used.
- To authenticate users, the `pam_winbind` module for PAM is used. The creation of user homes for the Active Directory users on the Linux client is handled by `pam_mkhomedir`.
  For more information about PAM, see *Chapter 2, Authentication with PAM*.

Figure 7.1, "Schema of Winbind-based Active Directory Authentication" highlights the most prominent components of Winbind-based Active Directory authentication.

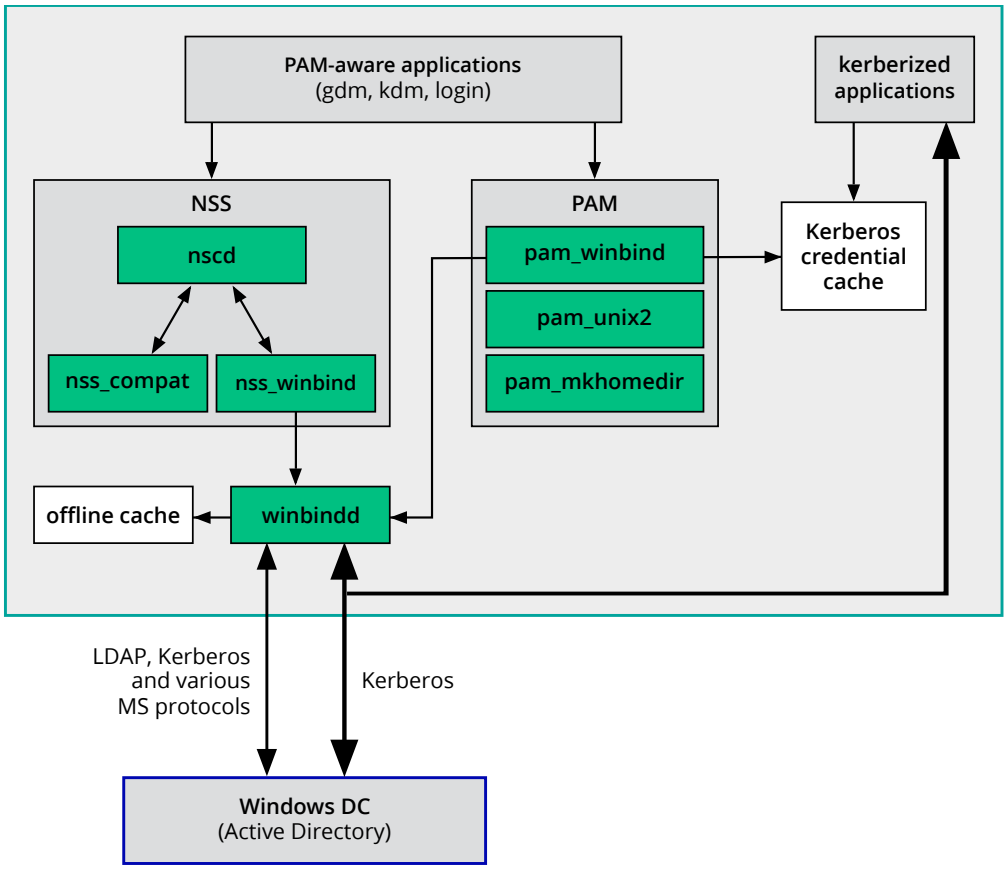


FIGURE 7.1: SCHEMA OF WINBIND-BASED ACTIVE DIRECTORY AUTHENTICATION

Applications that are PAM-aware, like the login routines and the GNOME display manager, interact with the PAM and NSS layer to authenticate against the Windows server. Applications supporting Kerberos authentication (such as file managers, Web browsers, or e-mail clients) use the Kerberos credential cache to access user's Kerberos tickets, making them part of the SSO framework.

7.2.1 Domain Join

During domain join, the server and the client establish a secure relation. On the client, the following tasks need to be performed to join the existing LDAP and Kerberos SSO environment provided by the Windows domain controller. The entire join process is handled by the YaST Domain Membership module, which can be run during installation or in the installed system:

1. The Windows domain controller providing both LDAP and KDC (Key Distribution Center) services is located.

2. A machine account for the joining client is created in the directory service.
3. An initial ticket granting ticket (TGT) is obtained for the client and stored in its local Kerberos credential cache. The client needs this TGT to get further tickets allowing it to contact other services, like contacting the directory server for LDAP queries.
4. NSS and PAM configurations are adjusted to enable the client to authenticate against the domain controller.

During client boot, the winbind daemon is started and retrieves the initial Kerberos ticket for the machine account. winbindd automatically refreshes the machine's ticket to keep it valid. To keep track of the current account policies, winbindd periodically queries the domain controller.

7.2.2 Domain Login and User Homes

The login manager of GNOME (GDM) has been extended to allow the handling of Active Directory domain login. Users can choose to log in to the primary domain the machine has joined or to one of the trusted domains with which the domain controller of the primary domain has established a trust relationship.

User authentication is mediated by several PAM modules as described in *Section 7.2, "Background Information for Linux Active Directory Support"*. If there are errors, the error codes are translated into user-readable error messages that PAM gives at login through any of the supported methods (GDM, console, and SSH):

`Password has expired`
: The user sees a message stating that the password has expired and needs to be changed. The system prompts for a new password and informs the user if the new password does not comply with corporate password policies (for example the password is too short, too simple, or already in the history). If a user's password change fails, the reason is shown and a new password prompt is given.

`Account disabled`
: The user sees an error message stating that the account has been disabled and to contact the system administrator.

`Account locked out`
: The user sees an error message stating that the account has been locked and to contact the system administrator.

`Password has to be changed`

The user can log in but receives a warning that the password needs to be changed soon. This warning is sent three days before that password expires. After expiration, the user cannot log in.

`Invalid workstation`

When a user is restricted to specific workstations and the current openSUSE Leap machine is not among them, a message appears that this user cannot log in from this workstation.

`Invalid logon hours`

When a user is only allowed to log in during working hours and tries to log in outside working hours, a message informs the user that logging in is not possible at that time.

`Account expired`

An administrator can set an expiration time for a specific user account. If that user tries to log in after expiration, the user gets a message that the account has expired and cannot be used to log in.

During a successful authentication, the client acquires a ticket granting ticket (TGT) from the Kerberos server of Active Directory and stores it in the user's credential cache. It also renews the TGT in the background, requiring no user interaction.

openSUSE Leap supports local home directories for Active Directory users. If configured through YaST as described in *Section 7.3, "Configuring a Linux Client for Active Directory"*, user home directories are created when a Windows/Active Directory user first logs in to the Linux client. These home directories look and feel identical to standard Linux user home directories and work independently of the Active Directory Domain Controller.

Using a local user home, it is possible to access a user's data on this machine (even when the Active Directory server is disconnected) as long as the Linux client has been configured to perform offline authentication.

7.2.3 Offline Service and Policy Support

Users in a corporate environment must have the ability to become roaming users (for example, to switch networks or even work disconnected for some time). To enable users to log in to a disconnected machine, extensive caching was integrated into the winbind daemon. The winbind daemon enforces password policies even in the offline state. It tracks the number of failed login attempts and reacts according to the policies configured in Active Directory. Offline support is disabled by default and must be explicitly enabled in the YaST Domain Membership module.

When the domain controller has become unavailable, the user can still access network resources (other than the Active Directory server itself) with valid Kerberos tickets that have been acquired before losing the connection (as in Windows). Password changes cannot be processed unless the domain controller is online. While disconnected from the Active Directory server, a user cannot access any data stored on this server. When a workstation has become disconnected from the network entirely and connects to the corporate network again later, openSUSE Leap acquires a new Kerberos ticket when the user has locked and unlocked the desktop (for example, using a desktop screen saver).

7.3 Configuring a Linux Client for Active Directory

Before your client can join an Active Directory domain, some adjustments must be made to your network setup to ensure the flawless interaction of client and server.

DNS

Configure your client machine to use a DNS server that can forward DNS requests to the Active Directory DNS server. Alternatively, configure your machine to use the Active Directory DNS server as the name service data source.

NTP

To succeed with Kerberos authentication, the client must have its time set accurately. It is highly recommended to use a central NTP time server for this purpose (this can be also the NTP server running on your Active Directory domain controller). If the clock skew between your Linux host and the domain controller exceeds a certain limit, Kerberos authentication fails and the client is logged in using the weaker NTLM (NT LAN Manager) authentication. For more details about using Active Directory for time synchronization, see *Procedure 7.2, "Joining an Active Directory Domain Using Windows Domain Membership"*.

Firewall

To browse your network neighborhood, either disable the firewall entirely or mark the interface used for browsing as part of the internal zone.

To change the firewall settings on your client, log in as `root` and start the YaST firewall module. Select *Interfaces*. Select your network interface from the list of interfaces and click *Change*. Select *Internal Zone* and apply your settings with *OK*. Leave the firewall settings with *Next* › *Finish*. To disable the firewall, check the *Disable Firewall Automatic Starting* option, and leave the firewall module with *Next* › *Finish*.

Active Directory Account

You cannot log in to an Active Directory domain unless the Active Directory administrator has provided you with a valid user account for that domain. Use the Active Directory user name and password to log in to the Active Directory domain from your Linux client.

7.3.1 Choosing Which YaST Module to Use for Connecting to Active Directory

YaST contains multiple modules that allow connecting to an Active Directory:

- ***User Logon Management***. Use both an identity service (usually LDAP) and a user authentication service (usually Kerberos). This option is based on SSSD and in the majority of cases is best suited for joining Active Directory domains.
  This module is described in *Section 7.3.2, "Joining Active Directory Using User Logon Management"*.

- ***Windows Domain Membership***. Join an Active Directory (which entails use of Kerberos and LDAP). This option is based on **`winbind`** and is best suited for joining an Active Directory domain if support for NTLM or cross-forest trusts is necessary.
  This module is described in *Section 7.3.3, "Joining Active Directory Using Windows Domain Membership"*.

- ***LDAP and Kerberos Authentication***. Allows setting up LDAP identities and Kerberos authentication independently from each other and provides fewer options. While this module also uses SSSD, it is not as well suited for connecting to Active Directory as the previous two options.
  This module is described in:

  - LDAP: *Section 5.3, "Configuring an LDAP Client with YaST"*
  - Kerberos: *Section 6.6, "Setting up Kerberos using LDAP and Kerberos Client"*

7.3.2 Joining Active Directory Using *User Logon Management*

The YaST module *User Logon Management* supports authentication at an Active Directory. Additionally, it also supports the following related authentication and identification providers:

Identification Providers

- ***Delegate to third-party software library***. Support for legacy NSS providers via a proxy.
- ***FreeIPA***. FreeIPA and Red Hat Enterprise Identity Management provider.
- ***Generic directory service (LDAP)***. An LDAP provider. For more information about configuring LDAP, see `man 5 sssd-ldap`.
- ***Local SSSD file database***. An SSSD-internal provider for local users.

Authentication Providers

- ***Delegate to third-party software library***. Relay authentication to another PAM target via a proxy.
- ***FreeIPA***. FreeIPA and Red Hat Enterprise Identity Management provider.
- ***Generic Kerberos service***. An LDAP provider.
- ***Generic directory service (LDAP)***. Kerberos authentication.
- ***Local SSSD file database***. An SSSD-internal provider for local users.
- ***This domain does not provide authentication service***. Disables authentication explicitly.

To join an Active Directory domain using SSSD and the *User Logon Management* module of YaST, proceed as follows:

PROCEDURE 7.1: JOINING AN ACTIVE DIRECTORY DOMAIN USING *USER LOGON MANAGEMENT*

1. Open YaST.
2. To be able to use DNS auto-discovery later, set up the Active Directory Domain Controller (the Active Directory server) as the name server for your client.
   a. In YaST, click *Network Settings*.
   b. Select *Hostname/DNS*, then enter the IP address of the Active Directory Domain Controller into the text box *Name Server 1*.
   Save the setting with *OK*.
3. From the YaST main window, start the module *User Logon Management*.
   The module opens with an overview showing different network properties of your computer and the authentication method currently in use.

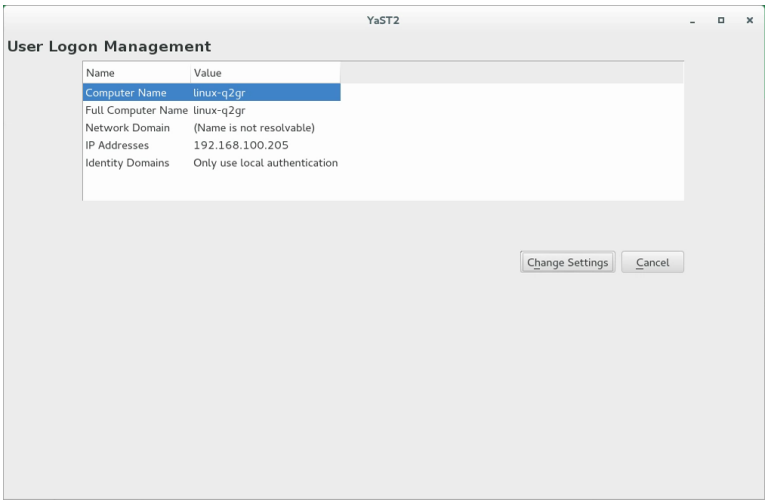


FIGURE 7.2: MAIN WINDOW OF *USER LOGON MANAGEMENT*

4. To start editing, click *Change Settings*.

5. Now join the domain.

   a. Click *Join Domain*.

   b. In the appearing dialog, specify the correct *Domain name*. Then specify the services to use for identity data and authentication: Select *Microsoft Active Directory* for both. Ensure that *Enable the domain* is activated.
   Click *OK*.

   c. *(Optional)* Usually, you can keep the default settings in the following dialog. However, there are reasons to make changes:

      - **If the Local Host Name Does Not Match the Host Name Set on the Domain Controller.** Find out if the host name of your computer matches what the name your computer is known as to the Active Directory Domain Controller. In a terminal, run the command `hostname`, then compare its output to the configuration of the Active Directory Domain Controller.
      If the values differ, specify the host name from the Active Directory configuration under *AD hostname*. Otherwise, leave the appropriate text box empty.

      - **If You Do Not Want to Use DNS Auto-Discovery.** Specify the *Host names of Active Directory servers* that you want to use. If there are multiple Domain Controllers, separate their host names with commas.

d. To continue, click *OK*.
   If not all software is installed already, the computer will now install missing software. It will then check whether the configured Active Directory Domain Controller is available.

e. If everything is correct, the following dialog should now show that it has discovered an *Active Directory Server* but that you are *Not yet enrolled*.
   In the dialog, specify the *Username* and *Password* of the Active Directory administrator account (usually `Administrator`).
   To make sure that the current domain is enabled for Samba, activate *Overwrite Samba configuration to work with this AD*.
   To enroll, click *OK*.

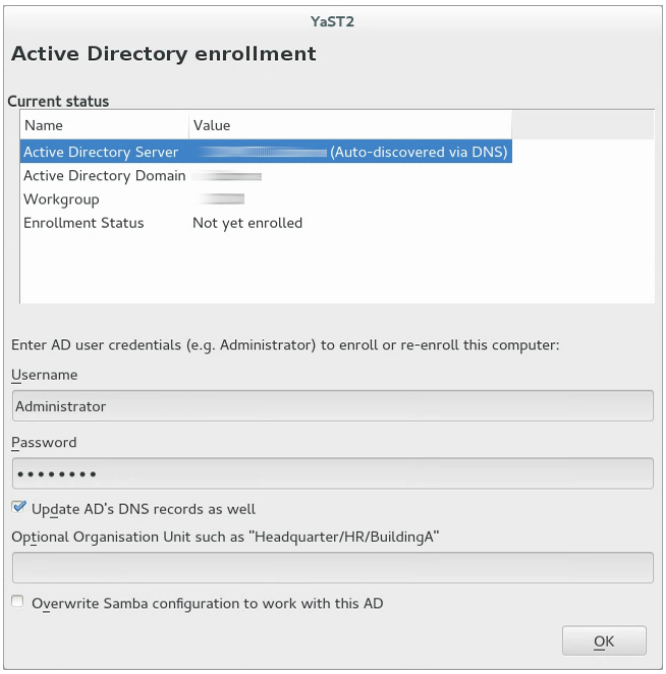


FIGURE 7.3: ENROLLING INTO A DOMAIN

f. You should now see a message confirming that you have enrolled successfully. Finish with *OK*.

6. After enrolling, configure the client using the window *Manage Domain User Logon*.

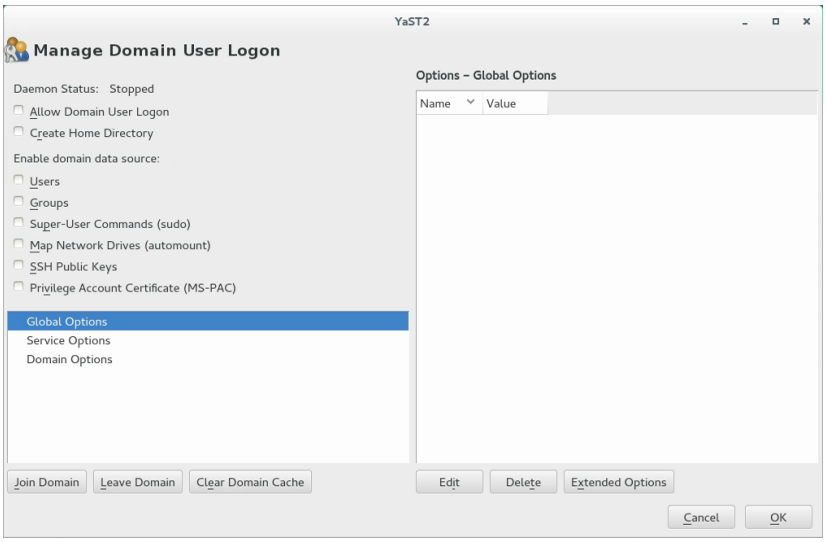


FIGURE 7.4: CONFIGURATION WINDOW OF *USER LOGON MANAGEMENT*

a. To allow logging in to the computer using login data provided by Active Directory, activate *Allow Domain User Logon*.

b. *(Optional)* Optionally, under *Enable domain data source*, activate additional data sources such as information on which users are allowed to use **`sudo`** or which network drives are available.

c. To allow Active Directory users to have home directories, activate *Create Home Directories*. The path for home directories can be set in multiple ways—on the client, on the server, or both ways:

   - To configure the home directory paths on the Domain Controller, set an appropriate value for the attribute `UnixHomeDirectory` for each user. Additionally, make sure that this attribute replicated to the global catalog. For information on achieving that under Windows, see https://support.microsoft.com/en-us/kb/248717.

   - To configure home directory paths on the client in such a way that precedence will be given to the path set on the domain controller, use the option `fallback_homedir`.

   - To configure home directory paths on the client in such a way that the client setting will override the server setting, use `override_homedir`.

As settings on the Domain Controller are outside of the scope of this documentation, only the configuration of the client-side options will be described in the following.
From the side bar, select *Service Options* › *Name switch*, then click *Extended Options*.
From that window, select either `fallback_homedir` or `override_homedir`, then click *Add*.
Specify a value. To have home directories follow the format `/home/USER_NAME`, use `/home/%u`. For more information about possible variables, see the man page `sssd.conf` (**`man 5 sssd.conf`**), section *override_homedir*.
Click *OK*.

7. Save the changes by clicking *OK*. Then make sure that the values displayed now are correct. To leave the dialog, click *Cancel*.

7.3.3 Joining Active Directory Using *Windows Domain Membership*

To join an Active Directory domain using **`winbind`** and the *Windows Domain Membership* module of YaST, proceed as follows:

PROCEDURE 7.2: JOINING AN ACTIVE DIRECTORY DOMAIN USING *WINDOWS DOMAIN MEMBERSHIP*

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 (see *Figure 7.5, "Determining Windows Domain Membership"*). If the DNS settings on your host are properly integrated with the Windows DNS server, enter the Active Directory domain name in its DNS format (`mydomain.mycompany.com`). If you enter the short name of your domain (also known as the pre–Windows 2000 domain name), YaST must rely on NetBIOS name resolution instead of DNS to find the correct domain controller.

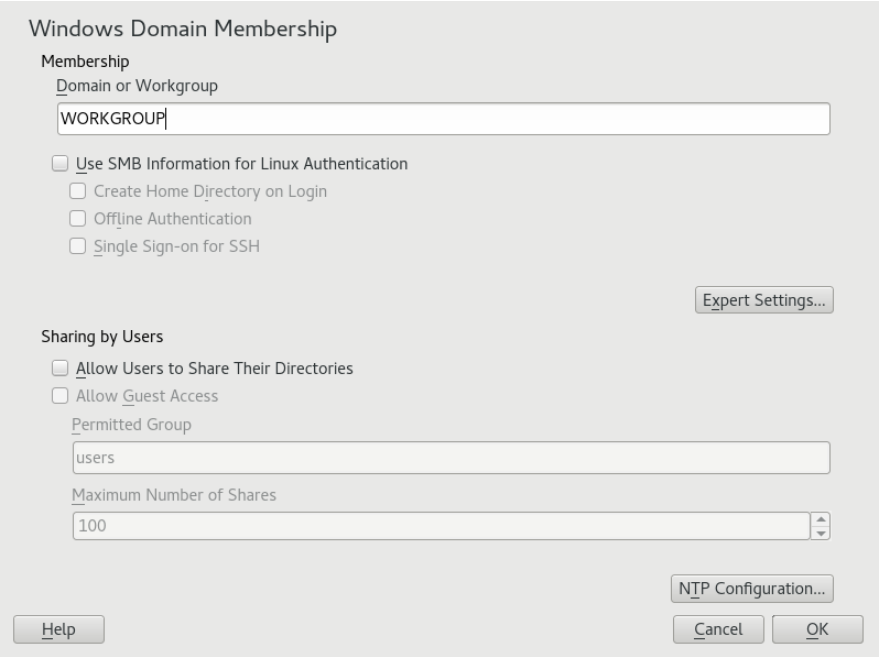


FIGURE 7.5: DETERMINING WINDOWS DOMAIN MEMBERSHIP

4. To use the SMB source for Linux authentication, activate *Also Use SMB Information for Linux Authentication*.

5. To automatically create a local home directory for Active Directory users on the Linux machine, activate *Create Home Directory on Login*.

6. Check *Offline Authentication* to allow your domain users to log in even if the Active Directory server is temporarily unavailable, or if you do not have a network connection.

7. To change the UID and GID ranges for the Samba users and groups, select *Expert Settings*. Let DHCP retrieve the WINS server only if you need it. This is the case when some machines are resolved only by the WINS system.

8. Configure NTP time synchronization for your Active Directory environment by selecting *NTP Configuration* and entering an appropriate server name or IP address. This step is obsolete if you have already entered the appropriate settings in the stand-alone YaST NTP configuration module.

9. Click *OK* and confirm the domain join when prompted for it.

10. Provide the password for the Windows administrator on the Active Directory server and click *OK* (see *Figure 7.6, "Providing Administrator Credentials"*).

FIGURE 7.6: PROVIDING ADMINISTRATOR CREDENTIALS

After you have joined the Active Directory domain, you can log in to it from your workstation using the display manager of your desktop or the console.

Important: Domain Name

Joining a domain may not succeed if the domain name ends with `.local`. Names ending in `.local` cause conflicts with Multicast DNS (MDNS) where `.local` is reserved for link-local host names.

Note: Only Administrators Can Enroll a Computer

Only a domain administrator account, such as `Administrator`, can join openSUSE Leap into Active Directory.

7.3.4 Checking Active Directory Connection Status

To check whether you are successfully enrolled in an Active Directory domain, use the following commands:

- **`klist`** shows whether the current user has a valid Kerberos ticket.
- **`getent passwd`** shows published LDAP data for all users.

7.4 Logging In to an Active Directory Domain

Provided your machine has been configured to authenticate against Active Directory and you have a valid Windows user identity, you can log in to your machine using the Active Directory credentials. Login is supported for GNOME, the console, SSH, and any other PAM-aware application.

Important: Offline Authentication

openSUSE Leap supports offline authentication, allowing you to log in to your client machine even when it is offline. See *Section 7.2.3, "Offline Service and Policy Support"* for details.

7.4.1 GDM

To authenticate a GNOME client machine against an Active Directory server, proceed as follows:

1. Click *Not listed*.
2. In the text box *Username*, enter the domain name and the Windows user name in this form: *DOMAIN_NAME\USER_NAME*.
3. Enter your Windows password.

If configured to do so, openSUSE Leap creates a user home directory on the local machine on the first login of each user authenticated via Active Directory. This allows you to benefit from the Active Directory support of openSUSE Leap while still having a fully functional Linux machine at your disposal.

7.4.2 Console Login

Besides logging in to the Active Directory client machine using a graphical front-end, you can log in using the text-based console or even remotely using SSH.

To log in to your Active Directory client from a console, enter *DOMAIN_NAME\USER_NAME* at the `login:` prompt and provide the password.

To remotely log in to your Active Directory client machine using SSH, proceed as follows:

1. At the login prompt, enter:

   ```
   tux > ssh DOMAIN_NAME\\USER_NAME@HOST_NAME
   ```

   The `\` domain and login delimiter is escaped with another `\` sign.

2. Provide the user's password.

7.5 Changing Passwords

openSUSE Leap helps the user choose a suitable new password that meets the corporate security policy. The underlying PAM module retrieves the current password policy settings from the domain controller, informing the user about the specific password quality requirements a user account typically has by means of a message on login. Like its Windows counterpart, openSUSE Leap presents a message describing:

- Password history settings
- Minimum password length requirements
- Minimum password age
- Password complexity

The password change process cannot succeed unless all requirements have been successfully met. Feedback about the password status is given both through the display managers and the console.

GDM provides feedback about password expiration and the prompt for new passwords in an interactive mode. To change passwords in the display managers, provide the password information when prompted.

To change your Windows password, you can use the standard Linux utility, **`passwd`**, instead of having to manipulate this data on the server. To change your Windows password, proceed as follows:

1. Log in at the console.
2. Enter `passwd`.

3. Enter your current password when prompted.
4. Enter the new password.
5. Reenter the new password for confirmation. If your new password does not comply with the policies on the Windows server, this feedback is given to you and you are prompted for another password.

To change your Windows password from the GNOME desktop, proceed as follows:

1. Click the *Computer* icon on the left edge of the panel.
2. Select *Control Center*.
3. From the *Personal* section, select *About Me* › *Change Password*.
4. Enter your old password.
5. Enter and confirm the new password.
6. Leave the dialog with *Close* to apply your settings.

II Local Security

8 Configuring Security Settings with YaST

The YaST module *Security Center and Hardening* offers a central clearinghouse to configure security-related settings for openSUSE Leap. Use it to configure security aspects such as settings for the login procedure and for password creation, for boot permissions, user creation or for default file permissions. Launch it from the YaST control center by *Security and Users* › *Security Center and Hardening*. The *Security Center* dialog always starts with the *Security Overview*, and other configuration dialogs are available from the right pane.

8.1 *Security Overview*

The *Security Overview* displays a comprehensive list of the most important security settings for your system. The security status of each entry in the list is clearly visible. A green check mark indicates a secure setting while a red cross indicates an entry as being insecure. Click *Help* to open an overview of the setting and information on how to make it secure. To change a setting, click the corresponding link in the Status column. Depending on the setting, the following entries are available:

Enabled/Disabled

Click this entry to toggle the status of the setting to either enabled or disabled.

Configure

Click this entry to launch another YaST module for configuration. You will return to the Security Overview when leaving the module.

Unknown

A setting's status is set to unknown when the associated service is not installed. Such a setting does not represent a potential security risk.

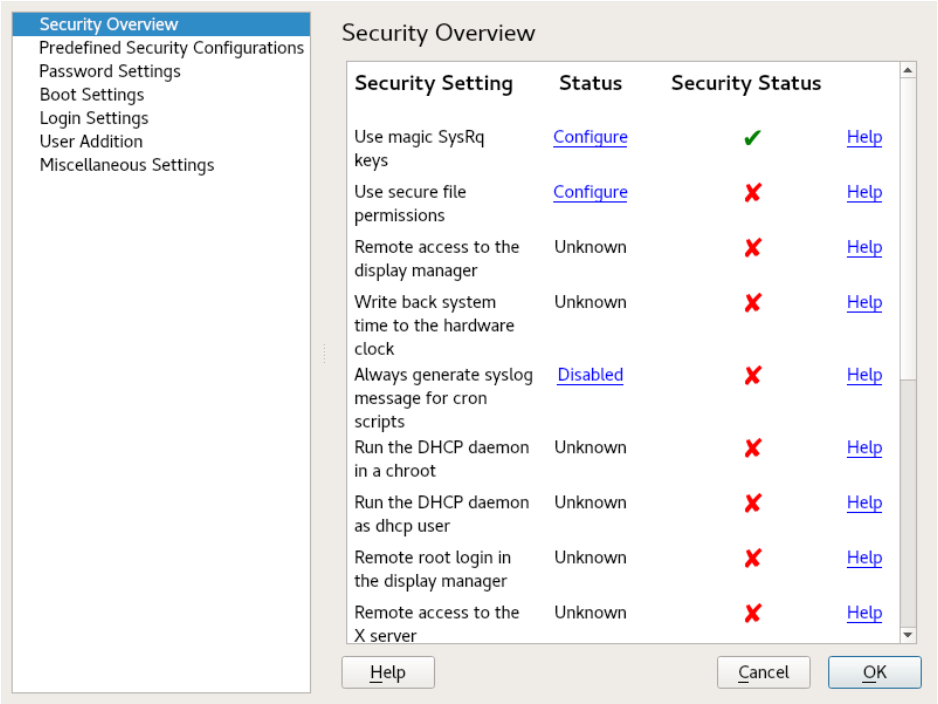


FIGURE 8.1: YAST SECURITY CENTER AND HARDENING: SECURITY OVERVIEW

8.2 *Predefined Security Configurations*

openSUSE Leap comes with three *Predefined Security Configurations*. These configurations affect all the settings available in the *Security Center* module. Each configuration can be modified to your needs using the dialogs available from the right pane changing its state to *Custom Settings*:

Workstation

A configuration for a workstation with any kind of network connection (including a connection to the Internet).

Roaming Device

This setting is designed for a laptop or tablet that connects to different networks.

Network Server

Security settings designed for a machine providing network services such as a Web server, file server, name server, etc. This set provides the most secure configuration of the predefined settings.

Custom Settings

A pre-selected *Custom Settings* (when opening the *Predefined Security Configurations* dialog) indicates that one of the predefined sets has been modified. Actively choosing this option does not change the current configuration—you will need to change it using the *Security Overview*.

8.3 *Password Settings*

Passwords that are easy to guess are a major security issue. The *Password Settings* dialog provides the means to ensure that only secure passwords can be used.

Check New Passwords

By activating this option, a warning will be issued if new passwords appear in a dictionary, or if they are proper names (proper nouns).

Minimum Acceptable Password Length

If the user chooses a password with a length shorter than specified here, a warning will be issued.

Number of Passwords to Remember

When password expiration is activated (via *Password Age*), this setting stores the given number of a user's previous passwords, preventing their reuse.

Password Encryption Method

Choose a password encryption algorithm. Normally there is no need to change the default (Blowfish).

Password Age

Activate password expiration by specifying a minimum and a maximum time limit (in days). By setting the minimum age to a value greater than `0` days, you can prevent users from immediately changing their passwords again (and in doing so circumventing the password expiration). Use the values `0` and `99999` to deactivate password expiration.

Days Before Password Expires Warning

When a password expires, the user receives a warning in advance. Specify the number of days prior to the expiration date that the warning should be issued.

8.4 Boot Settings

Configure which users can shut down the machine via the graphical login manager in this dialog. You can also specify how Ctrl–Alt–Del will be interpreted and who can hibernate the system.

8.5 Login Settings

This dialog lets you configure security-related login settings:

Delay after Incorrect Login Attempt

To make it difficult to guess a user's password by repeatedly logging in, it is recommended to delay the display of the login prompt that follows an incorrect login. Specify the value in seconds. Make sure that users who have mistyped their passwords do not need to wait too long.

Allow Remote Graphical Login

When checked, the graphical login manager (GDM) can be accessed from the network. This is a potential security risk.

8.6 User Addition

Set minimum and maximum values for user and group IDs. These default settings would rarely need to be changed.

8.7 Miscellaneous Settings

Other security settings that do not fit the above-mentioned categories are listed here:

File Permissions

openSUSE Leap comes with three predefined sets of file permissions for system files. These permission sets define whether a regular user may read log files or start certain programs. *Easy* file permissions are suitable for stand-alone machines. These settings allow regular users to, for example, read most system files. See the file `/etc/permissions.easy` for the complete configuration. The *Secure* file permissions are designed for multiuser machines

with network access. A thorough explanation of these settings can be found in `/etc/permissions.secure`. The *Paranoid* settings are the most restrictive ones and should be used with care. See `/etc/permissions.paranoid` for more information.

User Launching updatedb

The program **`updatedb`** scans the system and creates a database of all file locations which can be queried with the command **`locate`**. When **`updatedb`** is run as user nobody, only world-readable files will be added to the database. When run as user `root`, almost all files (except the ones root is not allowed to read) will be added.

Enable Magic SysRq Keys

The magic SysRq key is a key combination that enables you to have some control over the system even when it has crashed. The complete documentation can be found at https://www.kernel.org/doc/html/latest/admin-guide/sysrq.html.

9 Authorization with PolKit

PolKit (formerly known as PolicyKit) is an application framework that acts as a negotiator between the unprivileged user session and the privileged system context. Whenever a process from the user session tries to carry out an action in the system context, PolKit is queried. Based on its configuration—specified in a so-called “policy”—the answer could be “yes”, “no”, or “needs authentication”. Unlike classical privilege authorization programs such as sudo, PolKit does not grant `root` permissions to an entire session, but only to the action in question.

9.1 Conceptual Overview

PolKit works by limiting specific actions by users, by group, or by name. It then defines how those users are allowed to perform this action.

9.1.1 Available Authentication Agents

When a user starts a session (using the graphical environment or on the console), each session consists of the *authority* and an *authentication agent*. The authority is implemented as a service on the system message bus, whereas the authentication agent is used to authenticate the current user, which started the session. The current user needs to prove their authenticity, for example, using a passphrase.

Each desktop environment has its own authentication agent. Usually it is started automatically, whatever environment you choose.

9.1.2 Structure of PolKit

PolKit's configuration depends on *actions* and *authorization rules*:

Actions (file extension `*.policy`)

Written as XML files and located in `/usr/share/polkit-1/actions`. Each file defines one or more actions, and each action contains descriptions and default permissions. Although a system administrator can write their own rules, PolKit's files must not be edited.

Authorization Rules (file extension `*.rules`)

Written as JavaScript files and located in two places: `/usr/share/polkit-1/rules.d` is used for third party packages and `/etc/polkit-1/rules.d` for local configurations. Each rule file refers to the action specified in the action file. A rule determines what restrictions are allowed to a subset of users. For example, a rule file could overrule a restrictive permission and allow some users to allow it.

9.1.3 Available Commands

PolKit contains several commands for specific tasks (see also the specific man page for further details):

`pkaction`

Get details about a defined action. See *Section 9.3, "Querying Privileges"* for more information.

`pkcheck`

Checks whether a process is authorized, specified by either `--process` or `--system-bus-name`.

`pkexec`

Allows an authorized user to execute the specific program as another user.

`pkttyagent`

Starts a textual authentication agent. This agent is used if a desktop environment does not have its own authentication agent.

9.1.4 Available Policies and Supported Applications

At the moment, not all applications requiring privileges use PolKit. Find the most important policies available on openSUSE® Leap below, sorted into the categories where they are used.

PulseAudio

Set scheduling priorities for the PulseAudio daemon

CUPS

Add, remove, edit, enable or disable printers

Backup Manager

Modify schedule

GNOME

Modify system and mandatory values with GConf
Change the system time

libvirt

Manage and monitor local virtualized systems

NetworkManager

Apply and modify connections

PolKit

Read and change privileges for other users
Modify defaults

PackageKit

Update and remove packages
Change and refresh repositories
Install local files
Rollback
Import repository keys
Accepting EULAs
Setting the network proxy

System

Wake on LAN
Mount or unmount fixed, hotpluggable and encrypted devices
Eject and decrypt removable media
Enable or disable WLAN
Enable or disable Bluetooth
Device access
Stop, suspend, hibernate and restart the system
Undock a docking station

Change power-management settings

YaST

Register product
Change the system time and language

9.2 Authorization Types

Every time a PolKit-enabled process carries out a privileged operation, PolKit is asked whether this process is entitled to do so. PolKit answers according to the policy defined for this process. The answers can be `yes`, `no`, or `authentication needed`. By default, a policy contains `implicit` privileges, which automatically apply to all users. It is also possible to specify `explicit` privileges which apply to a specific user.

9.2.1 Implicit Privileges

Implicit privileges can be defined for any active and inactive sessions. An active session is the one in which you are currently working. It becomes inactive when you switch to another console for example. When setting implicit privileges to “no”, no user is authorized, whereas “yes” authorizes all users. However, usually it is useful to demand authentication.

A user can either authorize by authenticating as `root` or by authenticating as self. Both authentication methods exist in four variants:

Authentication

The user always needs to authenticate.

One Shot Authentication

The authentication is bound to the instance of the program currently running. After the program is restarted, the user is required to authenticate again.

Keep Session Authentication

The authentication dialog offers a check button *Remember authorization for this session*. If checked, the authentication is valid until the user logs out.

Keep Indefinitely Authentication

The authentication dialog offers a check button *Remember authorization*. If checked, the user needs to authenticate only once.

9.2.2 Explicit Privileges

Explicit privileges can be granted to specific users. They can either be granted without limitations, or, when using constraints, limited to an active session and/or a local console.

It is not only possible to grant privileges to a user, a user can also be blocked. Blocked users cannot carry out an action requiring authorization, even though the default implicit policy allows authorization by authentication.

9.2.3 Default Privileges

Each application supporting PolKit comes with a default set of implicit policies defined by the application's developers. Those policies are the so-called “upstream defaults”. The privileges defined by the upstream defaults are not necessarily the ones that are activated by default on SUSE systems. openSUSE Leap comes with a predefined set of privileges that override the upstream defaults:

`/etc/polkit-default-privs.standard`
: Defines privileges suitable for most desktop systems

`/etc/polkit-default-privs.restrictive`
: Designed for machines administrated centrally

To switch between the two sets of default privileges, adjust the value of `POLKIT_DEFAULT_PRIVS` to either `restrictive` or `standard` in `/etc/sysconfig/security`. Then run the command **`set_polkit_default_privs`** as `root`.

Do not modify the two files in the list above. To define your own custom set of privileges, use `/etc/polkit-default-privs.local`. For details, refer to *Section 9.4.3, “Modifying Configuration Files for Implicit Privileges”*.

9.3 Querying Privileges

To query privileges use the command **`pkaction`** included in PolKit.

PolKit comes with command line tools for changing privileges and executing commands as another user (see *Section 9.1.3, “Available Commands”* for a short overview). Each existing policy has a speaking, unique name with which it can be identified. List all available policies with the command **`pkaction`**.

When invoked with no parameters, the command **pkaction** lists all policies. By adding the `--show-overrides` option, you can list all policies that differ from the default values. To reset the privileges for a given action to the (upstream) defaults, use the option `--reset-defaults` *ACTION*. See **man pkaction** for more information.

If you want to display the needed authorization for a given policy (for example, `org.freedesktop.login1.reboot`) use **pkaction** as follows:

```
tux > pkaction -v --action-id org.freedesktop.login1.reboot
org.freedesktop.login1.reboot:
  description:       Reboot the system
  message:           Authentication is required to allow rebooting the system
  vendor:            The systemd Project
  vendor_url:        http://www.freedesktop.org/wiki/Software/systemd
  icon:
  implicit any:      auth_admin_keep
  implicit inactive: auth_admin_keep
  implicit active:   yes
```

The keyword `auth_admin_keep` means that users need to enter a passphrase.

Note: Restrictions of **pkaction** on openSUSE Leap

pkaction always operates on the upstream defaults. Therefore it cannot be used to list or restore the defaults shipped with openSUSE Leap. To do so, refer to *Section 9.5, "Restoring the Default Privileges"*.

9.4 Modifying Configuration Files

Adjusting privileges by modifying configuration files is useful when you want to deploy the same set of policies to different machines, for example to the computers of a specific team. It is possible to change implicit and explicit privileges by modifying configuration files.

9.4.1 Adding Action Rules

The available actions depend on what additional packages you have installed on your system. For a quick overview, use **pkaction** to list all defined rules.

To get an idea, the following example describes how the command **gparted** ("GNOME Partition Editor") is integrated into PolKit.

The file `/usr/share/polkit-1/actions/org.opensuse.policykit.gparted.policy` contains the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE policyconfig PUBLIC
 "-//freedesktop//DTD PolicyKit Policy Configuration 1.0//EN"
 "http://www.freedesktop.org/standards/PolicyKit/1.0/policyconfig.dtd">
<policyconfig> (1)

  <action id="org.opensuse.policykit.gparted"> (2)
    <message>Authentication is required to run the GParted Partition Editor</message>
    <icon_name>gparted</icon_name>
    <defaults> (3)
      <allow_any>auth_admin</allow_any>
      <allow_inactive>auth_admin</allow_inactive>
     < allow_active>auth_admin</allow_active>
    </defaults>
    <annotate (4)
      key="org.freedesktop.policykit.exec.path">/usr/sbin/gparted</annotate>
    <annotate (4)
      key="org.freedesktop.policykit.exec.allow_gui">true</annotate>
  </action>

</policyconfig>
```

1. Root element of the policy file.
2. Contains one single action.
3. The `defaults` element contains several permissions used in remote sessions like SSH, VNC (element `allow_inactive`), when logged directly into the machine on a TTY or X display (element `allow_active`), or for both (element `allow_any`). The value `auth_admin` indicates authentication is required as an administrative user.
4. The `annotate` element contains specific information regarding how PolKit performs an action. In this case, it contains the path to the executable and states whether a GUI is allowed to open a X display.

To add your own policy, create a `.policy` file with the structure above, add the appropriate value into the `id` attribute, and define the default permissions.

9.4.2 Adding Authorization Rules

Your own authorization rules overrule the default settings. To add your own settings, store your files under `/etc/polkit-1/rules.d/`.

The files in this directory start with a two-digit number, followed by a descriptive name, and end with `.rules`. Functions inside these files are executed in the order they have been sorted in. For example, `00-foo.rules` is sorted (and hence executed) before `60-bar.rules` or even `90-default-privs.rules`.

Inside the file, the script checks for the specified action ID, which is defined in the `.policy` file. For example, if you want to allow the command **gparted** to be executed by any member of the `admin` group, check for the action ID `org.opensuse.policykit.gparted`:

```
/* Allow users in admin group to run GParted without authentication */
polkit.addRule(function(action, subject) {
    if (action.id == "org.opensuse.policykit.gparted" &&
        subject.isInGroup("admin")) {
        return polkit.Result.YES;
    }
});
```

Find the description of all classes and methods of the functions in the PolKit API at http://www.freedesktop.org/software/polkit/docs/latest/ref-api.html.

9.4.3 Modifying Configuration Files for Implicit Privileges

openSUSE Leap ships with two sets of default authorizations, located in `/etc/polkit-default-privs.standard` and `/etc/polkit-default-privs.restrictive`. For more information, refer to *Section 9.2.3, "Default Privileges"*.

Custom privileges are defined in `/etc/polkit-default-privs.local`. Privileges defined here will always take precedence over the ones defined in the other configuration files. To define your custom set of privileges, do the following:

1. Open `/etc/polkit-default-privs.local`. To define a privilege, add a line for each policy with the following format:

   ```
   <privilege_identifier>     <any session>:<inactive session>:<active session>
   ```

   For example:

   ```
   org.freedesktop.policykit.modify-defaults     auth_admin_keep_always
   ```

The following values are valid for the *SESSION* placeholders:

`yes`
: grant privilege

`no`
: block

`auth_self`
: user needs to authenticate with own password every time the privilege is requested

`auth_self_keep_session`
: user needs to authenticate with own password once per session, privilege is granted for the whole session

`auth_self_keep_always`
: user needs to authenticate with own password once, privilege is granted for the current and for future sessions

`auth_admin`
: user needs to authenticate with `root` password every time the privilege is requested

`auth_admin_keep_session`
: user needs to authenticate with `root` password once per session, privilege is granted for the whole session

`auth_admin_keep_always`
: user needs to authenticate with `root` password once, privilege is granted for the current and for future sessions

2. Run as `root` for changes to take effect:

```
# /sbin/set_polkit_default_privs
```

3. Optionally check the list of all privilege identifiers with the command **`pkaction`**.

9.5 Restoring the Default Privileges

openSUSE Leap comes with a predefined set of privileges that is activated by default and thus overrides the upstream defaults. For details, refer to *Section 9.2.3, "Default Privileges"*.

Since the graphical PolKit tools and the command line tools always operate on the upstream defaults, openSUSE Leap includes an additional command-line tool, **`set_polkit_default_privs`**. It resets privileges to the values defined in `/etc/polkit-default-privs.*`. However, the command **`set_polkit_default_privs`** will only reset policies that are set to the upstream defaults.

PROCEDURE 9.1: RESTORING THE OPENSUSE LEAP DEFAULTS

1. Make sure `/etc/polkit-default-privs.local` does not contain any overrides of the default policies.

   ### Important: Custom Policy Configuration

   Policies defined in `/etc/polkit-default-privs.local` will be applied on top of the defaults during the next step.

2. To reset all policies to the upstream defaults first and then apply the openSUSE Leap defaults:

   ```
   tux > sudo rm -f /var/lib/polkit/* && set_polkit_default_privs
   ```

10 Access Control Lists in Linux

POSIX ACLs (access control lists) can be used as an expansion of the traditional permission concept for file system objects. With ACLs, permissions can be defined more flexibly than with the traditional permission concept.

The term *POSIX ACL* suggests that this is a true POSIX (*portable operating system interface*) standard. The respective draft standards POSIX 1003.1e and POSIX 1003.2c have been withdrawn for several reasons. Nevertheless, ACLs (as found on many systems belonging to the Unix family) are based on these drafts and the implementation of file system ACLs (as described in this chapter) follows these two standards.

10.1 Traditional File Permissions

The permissions of all files included in openSUSE Leap are carefully chosen. When installing additional software or files, take great care when setting the permissions. Always use the `-l` option with the command **`ls`** to detect any incorrect file permissions immediately. An incorrect file attribute does not only mean that files could be changed or deleted. Modified files could be executed by `root` or services could be hijacked by modifying configuration files. This significantly increases the danger of an attack.

A openSUSE® Leap system includes the files `permissions`, `permissions.easy`, `permissions.secure`, and `permissions.paranoid`, all in the directory `/etc`. The purpose of these files is to define special permissions, such as world-writable directories or, for files, the setuser ID bit. Programs with the setuser ID bit set do not run with the permissions of the user that launched it, but with the permissions of the file owner, usually `root`. An administrator can use the file `/etc/permissions.local` to add their own settings.

To define one of the available profiles, select *Local Security* in the *Security and Users* section of YaST. To learn more about the topic, read the comments in `/etc/permissions` or consult **`man chmod`**.

Find detailed information about the traditional file permissions in the GNU Coreutils Info page, Node *File permissions* (**`info coreutils "File permissions"`**). More advanced features are the setuid, setgid, and sticky bit.

10.1.1 The setuid Bit

In certain situations, the access permissions may be too restrictive. Therefore, Linux has additional settings that enable the temporary change of the current user and group identity for a specific action. For example, the **passwd** program normally requires root permissions to access `/etc/passwd`. This file contains some important information, like the home directories of users and user and group IDs. Thus, a normal user would not be able to change `passwd`, because it would be too dangerous to grant all users direct access to this file. A possible solution to this problem is the *setuid* mechanism. setuid (set user ID) is a special file attribute that instructs the system to execute programs marked accordingly under a specific user ID. Consider the **passwd** command:

```
-rwsr-xr-x  1 root shadow 80036 2004-10-02 11:08 /usr/bin/passwd
```

You can see the `s` that denotes that the setuid bit is set for the user permission. By means of the setuid bit, all users starting the **passwd** command execute it as `root`.

10.1.2 The setgid Bit

The setuid bit applies to users. However, there is also an equivalent property for groups: the *setgid* bit. A program for which this bit was set runs under the group ID under which it was saved, no matter which user starts it. Therefore, in a directory with the setgid bit, all newly created files and subdirectories are assigned to the group to which the directory belongs. Consider the following example directory:

```
drwxrws--- 2 tux archive 48 Nov 19 17:12  backup
```

You can see the `s` that denotes that the setgid bit is set for the group permission. The owner of the directory and members of the group `archive` may access this directory. Users that are not members of this group are "mapped" to the respective group. The effective group ID of all written files will be `archive`. For example, a backup program that runs with the group ID `archive` can access this directory even without root privileges.

10.1.3 The Sticky Bit

There is also the *sticky bit*. It makes a difference whether it belongs to an executable program or a directory. If it belongs to a program, a file marked in this way is loaded to RAM to avoid needing to get it from the hard disk each time it is used. This attribute is used rarely, because modern hard disks are fast enough. If this bit is assigned to a directory, it prevents users from deleting each other's files. Typical examples include the `/tmp` and `/var/tmp` directories:

```
drwxrwxrwt 2 root root 1160 2002-11-19 17:15 /tmp
```

10.2 Advantages of ACLs

Traditionally, three permission sets are defined for each file object on a Linux system. These sets include the read (`r`), write (`w`), and execute (`x`) permissions for each of three types of users—the file owner, the group, and other users. In addition to that, it is possible to set the *set user id*, the *set group id*, and the *sticky* bit. This lean concept is fully adequate for most practical cases. However, for more complex scenarios or advanced applications, system administrators formerly needed to use several workarounds to circumvent the limitations of the traditional permission concept.

ACLs can be used as an extension of the traditional file permission concept. They allow the assignment of permissions to individual users or groups even if these do not correspond to the original owner or the owning group. Access control lists are a feature of the Linux kernel and are currently supported by Ext2, Ext3, Ext4, JFS, and XFS. Using ACLs, complex scenarios can be realized without implementing complex permission models on the application level.

The advantages of ACLs are evident if you want to replace a Windows server with a Linux server. Some connected workstations may continue to run under Windows even after the migration. The Linux system offers file and print services to the Windows clients with Samba. With Samba supporting access control lists, user permissions can be configured both on the Linux server and in Windows with a graphical user interface (only Windows NT and later). With **`winbindd`**, part of the Samba suite, it is even possible to assign permissions to users only existing in the Windows domain without any account on the Linux server.

10.3 Definitions

User Class

The conventional POSIX permission concept uses three *classes* of users for assigning permissions in the file system: the owner, the owning group, and other users. Three permission bits can be set for each user class, giving permission to read (`r`), write (`w`), and execute (`x`).

ACL

The user and group access permissions for all kinds of file system objects (files and directories) are determined by means of ACLs.

Default ACL

Default ACLs can only be applied to directories. They determine the permissions a file system object inherits from its parent directory when it is created.

ACL Entry

Each ACL consists of a set of ACL entries. An ACL entry contains a type, a qualifier for the user or group to which the entry refers, and a set of permissions. For some entry types, the qualifier for the group or users is undefined.

10.4 Handling ACLs

Table 10.1, "ACL Entry Types" summarizes the six possible types of ACL entries, each defining permissions for a user or a group of users. The *owner* entry defines the permissions of the user owning the file or directory. The *owning group* entry defines the permissions of the file's owning group. The superuser can change the owner or owning group with **`chown`** or **`chgrp`**, in which case the owner and owning group entries refer to the new owner and owning group. Each *named user* entry defines the permissions of the user specified in the entry's qualifier field. Each *named group* entry defines the permissions of the group specified in the entry's qualifier field. Only the named user and named group entries have a qualifier field that is not empty. The *other* entry defines the permissions of all other users.

The *mask* entry further limits the permissions granted by named user, named group, and owning group entries by defining which of the permissions in those entries are effective and which are masked. If permissions exist in one of the mentioned entries and in the mask, they are effective. Permissions contained only in the mask or only in the actual entry are not effective—meaning the permissions are not granted. All permissions defined in the owner and owning group entries are always effective. The example in *Table 10.2, "Masking Access Permissions"* demonstrates this mechanism.

There are two basic classes of ACLs: A *minimum* ACL contains only the entries for the types owner, owning group, and other, which correspond to the conventional permission bits for files and directories. An *extended* ACL goes beyond this. It must contain a mask entry and may contain several entries of the named user and named group types.

TABLE 10.1: ACL ENTRY TYPES

Type	Text Form
owner	`user::rwx`
named user	`user:name:rwx`
owning group	`group::rwx`
named group	`group:name:rwx`
mask	`mask::rwx`
other	`other::rwx`

TABLE 10.2: MASKING ACCESS PERMISSIONS

Entry Type	Text Form	Permissions
named user	`user:geeko:r-x`	`r-x`
mask	`mask::rw-`	`rw-`
	effective permissions:	`r--`

10.4.1 ACL Entries and File Mode Permission Bits

Figure 10.1, "Minimum ACL: ACL Entries Compared to Permission Bits" and *Figure 10.2, "Extended ACL: ACL Entries Compared to Permission Bits"* illustrate the two cases of a minimum ACL and an extended ACL. The figures are structured in three blocks—the left block shows the type specifications of the ACL entries, the center block displays an example ACL, and the right block shows the respective permission bits according to the conventional permission concept (for example, as displayed by `ls -l`). In both cases, the *owner class* permissions are mapped to the ACL entry owner. *Other class* permissions are mapped to the respective ACL entry. However, the mapping of the *group class* permissions is different in the two cases.

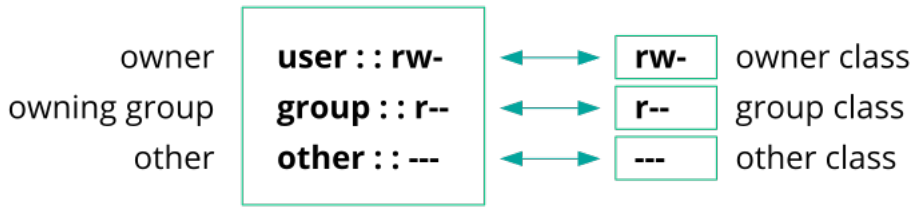


FIGURE 10.1: MINIMUM ACL: ACL ENTRIES COMPARED TO PERMISSION BITS

In the case of a minimum ACL—without mask—the group class permissions are mapped to the ACL entry owning group. This is shown in *Figure 10.1, "Minimum ACL: ACL Entries Compared to Permission Bits"*. In the case of an extended ACL—with mask—the group class permissions are mapped to the mask entry. This is shown in *Figure 10.2, "Extended ACL: ACL Entries Compared to Permission Bits"*.

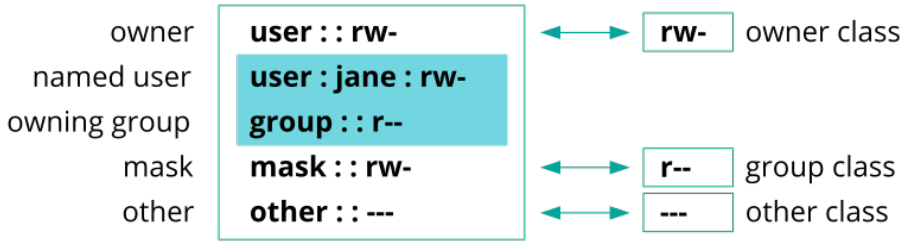


FIGURE 10.2: EXTENDED ACL: ACL ENTRIES COMPARED TO PERMISSION BITS

This mapping approach ensures the smooth interaction of applications, regardless of whether they have ACL support. The access permissions that were assigned by means of the permission bits represent the upper limit for all other "fine adjustments" made with an ACL. Changes made to the permission bits are reflected by the ACL and vice versa.

10.4.2 A Directory with an ACL

With `getfacl` and `setfacl` on the command line, you can access ACLs. The usage of these commands is demonstrated in the following example.

Before creating the directory, use the `umask` command to define which access permissions should be masked each time a file object is created. The command `umask 027` sets the default permissions by giving the owner the full range of permissions (`0`), denying the group write access (`2`), and giving other users no permissions (`7`). `umask` actually masks the corresponding permission bits or turns them off. For details, consult the `umask` man page.

`mkdir mydir` creates the `mydir` directory with the default permissions as set by `umask`. Use `ls -dl mydir` to check whether all permissions were assigned correctly. The output for this example is:

```
drwxr-x--- ... tux project3 ... mydir
```

With `getfacl mydir`, check the initial state of the ACL. This gives information like:

```
# file: mydir
# owner: tux
# group: project3
user::rwx
group::r-x
other::---
```

The first three output lines display the name, owner, and owning group of the directory. The next three lines contain the three ACL entries owner, owning group, and other. In fact, in the case of this minimum ACL, the `getfacl` command does not produce any information you could not have obtained with `ls`.

Modify the ACL to assign read, write, and execute permissions to an additional user `geeko` and an additional group `mascots` with:

```
root # setfacl -m user:geeko:rwx,group:mascots:rwx mydir
```

The option `-m` prompts `setfacl` to modify the existing ACL. The following argument indicates the ACL entries to modify (multiple entries are separated by commas). The final part specifies the name of the directory to which these modifications should be applied. Use the `getfacl` command to take a look at the resulting ACL.

```
# file: mydir
# owner: tux
# group: project3
user::rwx
user:geeko:rwx
group::r-x
group:mascots:rwx
mask::rwx
other::---
```

In addition to the entries initiated for the user `geeko` and the group `mascots`, a mask entry has been generated. This mask entry is set automatically so that all permissions are effective. `setfacl` automatically adapts existing mask entries to the settings modified, unless

you deactivate this feature with `-n`. The mask entry defines the maximum effective access permissions for all entries in the group class. This includes named user, named group, and owning group. The group class permission bits displayed by **`ls -dl mydir`** now correspond to the `mask` entry.

```
drwxrwx---+ ... tux project3 ... mydir
```

The first column of the output contains an additional `+` to indicate that there is an *extended* ACL for this item.

According to the output of the **`ls`** command, the permissions for the mask entry include write access. Traditionally, such permission bits would mean that the owning group (here `project3`) also has write access to the directory `mydir`.

However, the effective access permissions for the owning group correspond to the overlapping portion of the permissions defined for the owning group and for the mask—which is `r-x` in our example (see *Table 10.2, "Masking Access Permissions"*). As far as the effective permissions of the owning group in this example are concerned, nothing has changed even after the addition of the ACL entries.

Edit the mask entry with **`setfacl`** or **`chmod`**. For example, use **`chmod g-w mydir`**. **`ls -dl mydir`** then shows:

```
drwxr-x---+ ... tux project3 ... mydir
```

`getfacl mydir` provides the following output:

```
# file: mydir
# owner: tux
# group: project3
user::rwx
user:geeko:rwx          # effective: r-x
group::r-x
group:mascots:rwx       # effective: r-x
mask::r-x
other::---
```

After executing **`chmod`** to remove the write permission from the group class bits, the output of **`ls`** is sufficient to see that the mask bits must have changed accordingly: write permission is again limited to the owner of `mydir`. The output of the **`getfacl`** confirms this. This output includes a comment for all those entries in which the effective permission bits do not correspond to the original permissions, because they are filtered according to the mask entry. The original permissions can be restored at any time with **`chmod g+w mydir`**.

10.4.3 A Directory with a Default ACL

Directories can have a default ACL, which is a special kind of ACL defining the access permissions that objects in the directory inherit when they are created. A default ACL affects both subdirectories and files.

10.4.3.1 Effects of a Default ACL

There are two ways in which the permissions of a directory's default ACL are passed to the files and subdirectories:

- A subdirectory inherits the default ACL of the parent directory both as its default ACL and as an ACL.
- A file inherits the default ACL as its ACL.

All system calls that create file system objects use a `mode` parameter that defines the access permissions for the newly created file system object. If the parent directory does not have a default ACL, the permission bits as defined by the `umask` are subtracted from the permissions as passed by the `mode` parameter, with the result being assigned to the new object. If a default ACL exists for the parent directory, the permission bits assigned to the new object correspond to the overlapping portion of the permissions of the `mode` parameter and those that are defined in the default ACL. The `umask` is disregarded in this case.

10.4.3.2 Application of Default ACLs

The following three examples show the main operations for directories and default ACLs:

1. Add a default ACL to the existing directory `mydir` with:

   ```
   tux > setfacl -d -m group:mascots:r-x mydir
   ```

   The option `-d` of the **`setfacl`** command prompts **`setfacl`** to perform the following modifications (option `-m`) in the default ACL.
   Take a closer look at the result of this command:

   ```
   tux > getfacl mydir
   ```

```
# file: mydir
# owner: tux
# group: project3
user::rwx
user:geeko:rwx
group::r-x
group:mascots:rwx
mask::rwx
other::---
default:user::rwx
default:group::r-x
default:group:mascots:r-x
default:mask::r-x
default:other::---
```

getfacl returns both the ACL and the default ACL. The default ACL is formed by all lines that start with `default`. Although you merely executed the **setfacl** command with an entry for the `mascots` group for the default ACL, **setfacl** automatically copied all other entries from the ACL to create a valid default ACL. Default ACLs do not have an immediate effect on access permissions. They only come into play when file system objects are created. These new objects inherit permissions only from the default ACL of their parent directory.

2. In the next example, use **mkdir** to create a subdirectory in `mydir`, which inherits the default ACL.

```
tux > mkdir mydir/mysubdir

getfacl mydir/mysubdir

# file: mydir/mysubdir
# owner: tux
# group: project3
user::rwx
group::r-x
group:mascots:r-x
mask::r-x
other::---
default:user::rwx
default:group::r-x
default:group:mascots:r-x
default:mask::r-x
default:other::---
```

As expected, the newly-created subdirectory `mysubdir` has the permissions from the default ACL of the parent directory. The ACL of `mysubdir` is an exact reflection of the default ACL of `mydir`. The default ACL that this directory will hand down to its subordinate objects is also the same.

3. Use **`touch`** to create a file in the `mydir` directory, for example, **`touch`** `mydir/myfile`. **`ls`** `-l mydir/myfile` then shows:

```
-rw-r-----+ ... tux project3 ... mydir/myfile
```

The output of **`getfacl`** `mydir/myfile` is:

```
# file: mydir/myfile
# owner: tux
# group: project3
user::rw-
group::r-x          # effective:r--
group:mascots:r-x   # effective:r--
mask::r--
other::---
```

`touch` uses a `mode` with the value `0666` when creating new files, which means that the files are created with read and write permissions for all user classes, provided no other restrictions exist in **`umask`** or in the default ACL (see *Section 10.4.3.1, "Effects of a Default ACL"*). In effect, this means that all access permissions not contained in the `mode` value are removed from the respective ACL entries. Although no permissions were removed from the ACL entry of the group class, the mask entry was modified to mask permissions not set in `mode`.

This approach ensures the smooth interaction of applications (such as compilers) with ACLs. You can create files with restricted access permissions and subsequently mark them as executable. The **`mask`** mechanism guarantees that the right users and groups can execute them as desired.

10.4.4 The ACL Check Algorithm

A check algorithm is applied before any process or application is granted access to an ACL-protected file system object. As a basic rule, the ACL entries are examined in the following sequence: owner, named user, owning group or named group, and other. The access is handled in accordance with the entry that best suits the process. Permissions do not accumulate.

Things are more complicated if a process belongs to more than one group and would potentially suit several group entries. An entry is randomly selected from the suitable entries with the required permissions. It is irrelevant which of the entries triggers the final result “access granted”. Likewise, if none of the suitable group entries contain the required permissions, a randomly selected entry triggers the final result “access denied”.

10.5 ACL Support in Applications

ACLs can be used to implement very complex permission scenarios that meet the requirements of modern applications. The traditional permission concept and ACLs can be combined in a smart manner. The basic file commands (`cp`, `mv`, `ls`, etc.) support ACLs, as do Samba and Nautilus. Vi/Vim and emacs both fully support ACLs by preserving the permissions on writing files including backups. Unfortunately, many other editors and file managers still lack ACL support. When modifying files with an editor, the ACLs of files are sometimes preserved and sometimes not, depending on the backup mode of the editor used. If the editor writes the changes to the original file, the ACL is preserved. If the editor saves the updated contents to a new file that is subsequently renamed to the old file name, the ACLs may be lost, unless the editor supports ACLs. Except for the `star` archiver, there are currently no backup applications that preserve ACLs.

10.6 For More Information

For more information about ACLs, see the man pages for `getfacl(1)`, `acl(5)`, and `setfacl(1)`.

11 Encrypting Partitions and Files

Encrypting files, partitions, and entire disks prevents unauthorized access to your data and protects your confidential files and documents.

You can choose between the following encryption options:

Encrypting a Hard Disk Partition

It is possible to create an encrypted partition with YaST during installation or in an already installed system. For further info, see *Section 11.1.1, "Creating an Encrypted Partition during Installation"* and *Section 11.1.2, "Creating an Encrypted Partition on a Running System"*. This option can also be used for removable media, such as external hard disks, as described in *Section 11.1.4, "Encrypting the Content of Removable Media"*.

Creating an Encrypted Virtual Disk

You can create a file-based encrypted virtual disk on your hard disk or a removable medium with YaST. The encrypted virtual disk can then be used as a regular folder for storing files or directories. For more information, refer to *Section 11.1.3, "Creating an Encrypted Virtual Disk"*.

Encrypting Single Files with GPG

To quickly encrypt one or more files, you can use the GPG tool. See *Section 11.2, "Encrypting Files with GPG"* for more information.

Warning: Encryption Offers Limited Protection

Encryption methods described in this chapter cannot protect your running system from being compromised. After the encrypted volume is successfully mounted, everybody with appropriate permissions can access it. However, encrypted media are useful in case of loss or theft of your computer, or to prevent unauthorized individuals from reading your confidential data.

11.1 Setting Up an Encrypted File System with YaST

Use YaST to encrypt partitions or parts of your file system during installation or in an already installed system. However, encrypting a partition in an already-installed system is more difficult, because you need to resize and change existing partitions. In such cases, it may be more convenient to create an encrypted file of a defined size, in which to *store* other files or parts

of your file system. To encrypt an entire partition, dedicate a partition for encryption in the partition layout. The standard partitioning proposal, as suggested by YaST, does not include an encrypted partition by default. Add an encrypted partition manually in the partitioning dialog.

11.1.1 Creating an Encrypted Partition during Installation

Warning: Password Input

Make sure to memorize the password for your encrypted partitions well. Without that password, you cannot access or restore the encrypted data.

The YaST expert dialog for partitioning offers the options needed for creating an encrypted partition. To create a new encrypted partition proceed as follows:

1. Run the YaST Expert Partitioner with *System* › *Partitioner*.
2. Select a hard disk, click *Add*, and select a primary or an extended partition.
3. Select the partition size or the region to use on the disk.
4. Select the file system, and mount point of this partition.
5. Activate the *Encrypt device* check box.

   #### Note: Additional Software Required

   After checking *Encrypt device*, a pop-up window asking for installing additional software may appear. Confirm to install all the required packages to ensure that the encrypted partition works well.

6. If the encrypted file system needs to be mounted only when necessary, enable *Do not mount partition* in the *Fstab Options*. otherwise enable *Mount partition* and enter the mount point.
7. Click *Next* and enter a password which is used to encrypt this partition. This password is not displayed. To prevent typing errors, you need to enter the password twice.
8. Complete the process by clicking *Finish*. The newly-encrypted partition is now created.

During the boot process, the operating system asks for the password before mounting any encrypted partition which is set to be auto-mounted in `/etc/fstab`. Such a partition is then available to all users when it has been mounted.

To skip mounting the encrypted partition during start-up, press `Enter` when prompted for the password. Then decline the offer to enter the password again. In this case, the encrypted file system is not mounted and the operating system continues booting, blocking access to your data.

To mount an encrypted partition which is not mounted during the boot process, open a file manager and click the partition entry in the pane listing common places on your file system. You will be prompted for a password and the partition will be mounted.

When you are installing your system on a machine where partitions already exist, you can also decide to encrypt an existing partition during installation. In this case follow the description in *Section 11.1.2, "Creating an Encrypted Partition on a Running System"* and be aware that this action destroys all data on the existing partition.

11.1.2 Creating an Encrypted Partition on a Running System

Warning: Activating Encryption on a Running System

It is also possible to create encrypted partitions on a running system. However, encrypting an existing partition destroys all data on it, and requires re-sizing and restructuring of existing partitions.

On a running system, select *System* › *Partitioner* in the YaST control center. Click *Yes* to proceed. In the *Expert Partitioner*, select the partition to encrypt and click *Edit*. The rest of the procedure is the same as described in *Section 11.1.1, "Creating an Encrypted Partition during Installation"*.

11.1.3 Creating an Encrypted Virtual Disk

Instead of encrypting an entire disk or partition, you can use YaST to set up a file-based encrypted virtual disk. It will appear as a regular file in the file system, but can be mounted and used like a regular folder. Unlike encrypted partitions, encrypted virtual disks can be created without re-partitioning the hard disk.

To set up an encrypted virtual disk, you need to create an empty file first. This file is called a loop file and is going to contain the encrypted data. In the terminal, switch to the desired directory and run the **`touch`** *`LOOP_FILE`* command (where *`LOOP_FILE`* is the desired name, for example: `secret`). We recommend to create an empty directory that will act as a mount point for the encrypted virtual disk. To do this, use the **`mkdir`** *`MOUNT_DIR`* command (replace *`MOUNT_DIR`* with the actual path and directory name, for example `~/my_docs`). *`LOOP_FILE`* must reside outside of *`MOUNT_DIR`*.

To set up an encrypted virtual disk, launch YaST, switch to the *System* section, and start the Partitioner. Switch to the *Crypt Files* section and press *Add Crypt File*. Enter the path to the created loop file (*`LOOP_FILE`*) into the *Path Name of Loop File* field. Enable the *Create Loop File* option, specify the desired size, and press *Next*. In the *Mount Point* field, enter the path to the directory that serves as a mount point (*`MOUNT_DIR`*, in this example it is `~/my_docs`). Make sure that the *Encrypt Device* option is enabled and press *Next*. Provide the desired password and press *Finish*.

YaST changes the owner of the mount point to `root` by default. If the content should be accessible to other users, change the group and permissions, for example with **`chgrp users`** *`MOUNT_DIR`* and **`chmod 775`** *`MOUNT_DIR`*.

11.1.4 Encrypting the Content of Removable Media

YaST treats removable media (like external hard disks or flash disks) the same as any other storage device. Virtual disks or partitions on external media can be encrypted as described above. However, you should disable mounting at boot time, because removable media is usually connected only when the system is up and running.

If you encrypted your removable device with YaST, the GNOME desktop automatically recognizes the encrypted partition and prompts for the password when the device is detected. If you plug in a FAT-formatted removable device when running GNOME, the desktop user entering the password automatically becomes the owner of the device. For devices with a file system other than FAT, change the ownership explicitly for users other than `root` to give them read-write access to the device.

If you have created a virtual disk as described in *Section 11.1.3, “Creating an Encrypted Virtual Disk”* but with the loop file on a removable disk, then you need to mount the file manually as follows:

```
tux > sudo cryptsetup luksOpen  FILE NAME
sudo mount /dev/mapper/NAME DIR
```

In the commands above, *FILE* refers to the path to the loop file, *NAME* is a user-defined name, and *DIR* is the path to the mount point. For example:

```
tux > sudo cryptsetup luksOpen /run/media/tux/usbstick/secret my_secret
tux > sudo mount /dev/mapper/my_secret /home/tux/my_docs
```

11.2 Encrypting Files with GPG

The GPG encryption software can be used to encrypt individual files and documents.

To encrypt a file with GPG, you need to generate a key pair first. To do this, run the **gpg --gen-key** and follow the on-screen instructions. When generating the key pair, GPG creates a user ID (UID) to identify the key based on your real name, comments, and email address. You need this UID (or just a part of it like your first name or email address) to specify the key you want to use to encrypt a file. To find the UID of an existing key, use the **gpg --list-keys** command. To encrypt a file use the following command:

```
tux > gpg -e -r UID
  FILE
```

Replace *UID* with part of the UID (for example, your first name) and *FILE* with the file you want to encrypt. For example:

```
tux > gpg -e -r Tux secret.txt
```

This command creates an encrypted version of the specified file recognizable by the `.gpg` file extension (in this example, it is `secret.txt.gpg`).

To decrypt an encrypted file, use the following command:

```
tux > gpg -d -o DECRYPTED_FILE
  ENCRYPTED_FILE
```

Replace *DECRYPTED_FILE* with the desired name for the decrypted file and *ENCRYPTED_FILE* with the encrypted file you want to decrypt.

Keep in mind that the encrypted file can be only decrypted using the same key that was used for encryption. If you want to share an encrypted file with another person, you have to use that person's public key to encrypt the file.

12 Certificate Store

Certificates play an important role in the authentication of companies and individuals. Usually certificates are administered by the application itself. In some cases, it makes sense to share certificates between applications. The certificate store is a common ground for Firefox, Evolution, and NetworkManager. This chapter explains some details.

The certificate store is a common database for Firefox, Evolution, and NetworkManager at the moment. Other applications that use certificates are not covered but may be in the future. If you have such an application, you can continue to use its private, separate configuration.

12.1 Activating Certificate Store

The configuration is mostly done in the background. To activate it, proceed as follows:

1. Decide if you want to activate the certificate store globally (for every user on your system) or specifically to a certain user:
   - **For every user.** Use the file `/etc/profile.local`
   - **For a specific user.** Use the file `~/.bashrc`
2. Open the file from the previous step and insert the following line:

   ```
   export NSS_USE_SHARED_DB=1
   ```

   Save the file
3. Log out of and log in to your desktop.

All the certificates are stored under `$HOME/.local/var/pki/nssdb/`.

12.2 Importing Certificates

To import a certificate into the certificate store, do the following:

1. Start Firefox.

2. Open the dialog from *Edit* › *Preferences*. Change to *Advanced* › *Encryption* and click *View Certificates*.
3. Import your certificate depending on your type: use *Servers* to import server certificate, *People* to identify other, and *Your Certificates* to identify yourself.

13 Intrusion Detection with AIDE

Securing your systems is a mandatory task for any mission-critical system administrator. Because it is impossible to always guarantee that the system is not compromised, it is very important to do extra checks regularly (for example with `cron`) to ensure that the system is still under your control. This is where AIDE, the *Advanced Intrusion Detection Environment*, comes into play.

13.1 Why Use AIDE?

An easy check that often can reveal unwanted changes can be done by means of RPM. The package manager has a built-in verify function that checks all the managed files in the system for changes. To verify of all files, run the command `rpm -Va`. However, this command will also display changes in configuration files and you will need to do some filtering to detect important changes.

An additional problem to the method with RPM is that an intelligent attacker will modify `rpm` itself to hide any changes that might have been done by some kind of root-kit which allows the attacker to mask its intrusion and gain root privilege. To solve this, you should implement a secondary check that can also be run completely independent of the installed system.

13.2 Setting Up an AIDE Database

Important: Initialize AIDE Database After Installation

Before you install your system, verify the checksum of your medium (see *Book "Start-Up", Chapter 4 "Troubleshooting", Section 4.1 "Checking Media"*) to make sure you do not use a compromised source. After you have installed the system, initialize the AIDE database. To make sure that all went well during and after the installation, do an installation directly on the console, without any network attached to the computer. Do not leave the computer unattended or connected to any network before AIDE creates its database.

AIDE is not installed by default on openSUSE Leap. To install it, either use *Computer › Install Software*, or enter `zypper install aide` on the command line as `root`.

To tell AIDE which attributes of which files should be checked, use the `/etc/aide.conf` configuration file. It must be modified to become the actual configuration. The first section handles general parameters like the location of the AIDE database file. More relevant for local configurations are the `Custom Rules` and the `Directories and Files` sections. A typical rule looks like the following:

```
Binlib     = p+i+n+u+g+s+b+m+c+md5+sha1
```

After defining the variable `Binlib`, the respective check boxes are used in the files section. Important options include the following:

TABLE 13.1: IMPORTANT AIDE CHECK BOXES

Option	Description
p	Check for the file permissions of the selected files or directories.
i	Check for the inode number. Every file name has a unique inode number that should not change.
n	Check for the number of links pointing to the relevant file.
u	Check if the owner of the file has changed.
g	Check if the group of the file has changed.
s	Check if the file size has changed.
b	Check if the block count used by the file has changed.
m	Check if the modification time of the file has changed.
c	Check if the files access time has changed.

Option	Description
S	Check for a changed file size.
I	Ignore changes of the file name.
md5	Check if the md5 checksum of the file has changed. We recommend to use sha256 or sha512.
sha1	Check if the sha1 (160 Bit) checksum of the file has changed. We recommend to use sha256 or sha512.
sha256	Check if the sha256 checksum of the file has changed.
sha512	Check if the sha512 checksum of the file has changed.

This is a configuration that checks for all files in `/sbin` with the options defined in `Binlib` but omits the `/sbin/conf.d/` directory:

```
/sbin  Binlib
!/sbin/conf.d
```

To create the AIDE database, proceed as follows:

1. Open `/etc/aide.conf`.
2. Define which files should be checked with which check boxes. For a complete list of available check boxes, see `/usr/share/doc/packages/aide/manual.html`. The definition of the file selection needs some knowledge about regular expressions. Save your modifications.
3. To check whether the configuration file is valid, run:

   ```
   root # aide --config-check
   ```

Any output of this command is a hint that the configuration is not valid. For example, if you get the following output:

```
root # aide --config-check
35:syntax error:!
35:Error while reading configuration:!
Configuration error
```

The error is to be expected in line 36 of `/etc/aide.conf`. Note that the error message contains the last successfully read line of the configuration file.

4. Initialize the AIDE database. Run the command:

```
root # aide -i
```

5. Copy the generated database to a save location like a CD-R or DVD-R, a remote server or a flash disk for later use.

> **Important:**
>
> This step is essential as it avoids compromising your database. It is recommended to use a medium which can be written only once to prevent the database being modified. *Never* leave the database on the computer which you want to monitor.

13.3 Local AIDE Checks

To perform a file system check, proceed as follows:

1. Rename the database:

```
root # mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db
```

2. After any configuration change, you always need to re-initialize the AIDE database and subsequently move the newly generated database. It is also a good idea to make a backup of this database. See *Section 13.2, "Setting Up an AIDE Database"* for more information.

3. Perform the check with the following command:

```
root # aide --check
```

If the output is empty, everything is fine. If AIDE found changes, it displays a summary of changes, for example:

```
root # aide --check
AIDE found differences between database and filesystem!!

Summary:
  Total number of files:        1992
  Added files:                  0
  Removed files:                0
  Changed files:                1
```

To learn about the actual changes, increase the verbose level of the check with the parameter `-V`. For the previous example, this could look like the following:

```
root # aide --check -V
AIDE found differences between database and filesystem!!
Start timestamp: 2009-02-18 15:14:10

Summary:
  Total number of files:        1992
  Added files:                  0
  Removed files:                0
  Changed files:                1


---------------------------------------------------
Changed files:
---------------------------------------------------

changed: /etc/passwd

---------------------------------------------------
Detailed information about changes:
---------------------------------------------------


File: /etc/passwd
  Mtime    : 2009-02-18 15:11:02              , 2009-02-18 15:11:47
  Ctime    : 2009-02-18 15:11:02              , 2009-02-18 15:11:47
```

In this example, the file `/etc/passwd` was touched to demonstrate the effect.

13.4 System Independent Checking

To avoid risk, it is advisable to also run the AIDE binary from a trusted source. This excludes the risk that some attacker also modified the aide binary to hide its traces.

To accomplish this task, AIDE must be run from a rescue system that is independent of the installed system. With openSUSE Leap it is relatively easy to extend the rescue system with arbitrary programs, and thus add the needed functionality.

Before you can start using the rescue system, you need to provide two packages to the system. These are included with the same syntax as you would add a driver update disk to the system. For a detailed description about the possibilities of linuxrc that are used for this purpose, see http://en.opensuse.org/SDB:Linuxrc. In the following, one possible way to accomplish this task is discussed.

PROCEDURE 13.1: STARTING A RESCUE SYSTEM WITH AIDE

1. Provide an FTP server as a second machine.

2. Copy the packages `aide` and `mhash` to the FTP server directory, in our case `/srv/ftp/`. Replace the placeholders *`ARCH`* and *`VERSION`* with the corresponding values:

```
root # cp DVD1/suse/ARCH/aideVERSION.ARCH.rpm /srv/ftp
root # cp DVD1/suse/ARCH/mhashVERSION.ARCH.rpm /srv/ftp
```

3. Create an info file `/srv/ftp/info.txt` that provides the needed boot parameters for the rescue system:

```
dud:ftp://ftp.example.com/aideVERSION.ARCH.rpm
dud:ftp://ftp.example.com/mhashVERSION.ARCH.rpm
```

   Replace your FTP domain name, *`VERSION`* and *`ARCH`* with the values used on your system.

4. Restart the server that needs to go through an AIDE check with the Rescue system from your DVD. Add the following string to the boot parameters:

```
info=ftp://ftp.example.com/info.txt
```

   This parameter tells **`linuxrc`** to also read in all information from the `info.txt` file.

After the rescue system has booted, the AIDE program is ready for use.

13.5 For More Information

Information about AIDE is available at the following places:

- The home page of AIDE: http://aide.sourceforge.net
- In the documented template configuration `/etc/aide.conf`.
- In several files below `/usr/share/doc/packages/aide` after installing the `aide` package.
- On the AIDE user mailing list at https://www.ipi.fi/mailman/listinfo/aide.

III Network Security

14 X Window System and X Authentication

As mentioned at the beginning, network transparency is one of the central characteristics of a Unix system. X, the windowing system of Unix operating systems, can use this feature in an impressive way. With X, it is no problem to log in to a remote host and start a graphical program that is then sent over the network to be displayed on your computer.

When an X client needs to be displayed remotely using an X server, the latter should protect the resource managed by it (the display) from unauthorized access. In more concrete terms, certain permissions must be given to the client program. With the X Window System, there are two ways to do this, called host-based access control and cookie-based access control. The former relies on the IP address of the host where the client should run. The program to control this is **`xhost`**. **`xhost`** enters the IP address of a legitimate client into a database belonging to the X server. However, relying on IP addresses for authentication is not very secure. For example, if there were a second user working on the host sending the client program, that user would have access to the X server as well—like someone spoofing the IP address. Because of these shortcomings, this authentication method is not described in more detail here, but you can learn about it with **`man`** `xhost`.

In the case of cookie-based access control, a character string is generated that is only known to the X server and to the legitimate user, like an ID card of some kind. This cookie is stored on login in the file `.Xauthority` in the user's home directory and is available to any X client wanting to use the X server to display a window. The file `.Xauthority` can be examined by the user with the tool **`xauth`**. If you rename `.Xauthority`, or if you delete the file from your home directory by accident, you will not be able to open any new windows or X clients.

SSH (secure shell) can be used to encrypt a network connection and forward it to an X server transparently. This is also called X forwarding. X forwarding is achieved by simulating an X server on the server side and setting a DISPLAY variable for the shell on the remote host. Further details about SSH can be found in *Chapter 15, SSH: Secure Network Operations*.

Warning: X Forwarding Can Be Insecure

If you do not consider the computer where you log in to be a secure host, do not use X forwarding. If X forwarding is enabled, an attacker could authenticate via your SSH connection. The attacker could then intrude on your X server and, for example, read your keyboard input.

15 SSH: Secure Network Operations

In networked environments, it is often necessary to access hosts from a remote location. If a user sends login and password strings for authentication purposes as plain text, they could be intercepted and misused to gain access to that user account. This would open all the user's files to an attacker and the illegal account could be used to obtain administrator or `root` access, or to penetrate other systems. In the past, remote connections were established with **`telnet`**, **`rsh`** or **`rlogin`**, which offered no guards against eavesdropping in the form of encryption or other security mechanisms. There are other unprotected communication channels, like the traditional FTP protocol and some remote copying programs like **`rcp`**.

The SSH suite provides the necessary protection by encrypting the authentication strings (usually a login name and a password) and all the other data exchanged between the hosts. With SSH, the data flow could still be recorded by a third party, but the contents are encrypted and cannot be reverted to plain text unless the encryption key is known. So SSH enables secure communication over insecure networks, such as the Internet. The SSH implementation coming with openSUSE Leap is OpenSSH.

openSUSE Leap installs the OpenSSH package by default providing the commands **`ssh`**, **`scp`**, and **`sftp`**. In the default configuration, remote access of a openSUSE Leap system is only possible with the OpenSSH utilities, and only if the `sshd` is running and the firewall permits access.

SSH on openSUSE Leap uses cryptographic hardware acceleration if available. As a result, the transfer of large quantities of data through an SSH connection is considerably faster than without cryptographic hardware. As an additional benefit, the CPU will see a significant reduction in load.

15.1 **`ssh`**—Secure Shell

With **`ssh`** it is possible to log in to remote systems and to work interactively. To log in to the host `sun` as user `tux` enter one of the following commands:

```
tux > ssh tux@sun
tux > ssh -l tux sun
```

If the user name is the same on both machines, you can omit it. Using `ssh sun` is sufficient. The remote host prompts for the remote user's password. After a successful authentication, you can work on the remote command line or use interactive applications, such as YaST in text mode.

Furthermore, `ssh` offers the possibility to run non-interactive commands on remote systems using `ssh` *HOST COMMAND*. *COMMAND* needs to be properly quoted. Multiple commands can be concatenated as on a local shell.

```
tux > ssh root@sun "dmesg -T | tail -n 25"
tux > ssh root@sun "cat /etc/issue && uptime"
```

15.1.1 Starting X Applications on a Remote Host

SSH also simplifies the use of remote X applications. If you run `ssh` with the `-X` option, the `DISPLAY` variable is automatically set on the remote machine and all X output is exported to the local machine over the existing SSH connection. At the same time, X applications started remotely cannot be intercepted by unauthorized individuals.

15.1.2 Agent Forwarding

By adding the `-A` option, the ssh-agent authentication mechanism is carried over to the next machine. This way, you can work from different machines without having to enter a password, but only if you have distributed your public key to the destination hosts and properly saved it there. Refer to *Section 15.5.2, "Copying an SSH Key"* for details.

This mechanism is deactivated in the default settings, but can be permanently activated at any time in the system wide configuration file `/etc/ssh/sshd_config` by setting `AllowAgentForwarding yes`.

15.2 `scp`—Secure Copy

`scp` copies files to or from a remote machine. If the user name on jupiter is different than the user name on sun, specify the latter using the *USER_NAME*`@host` format. If the file should be copied into a directory other than the remote user's home directory, specify it as sun:*DIRECTORY*. The following examples show how to copy a file from a local to a remote machine and vice versa.

```
tux > scp ~/MyLetter.tex tux@sun:/tmp ❶
```

```
tux > scp tux@sun:/tmp/MyLetter.tex ~ ❷
```

❶ local to remote

❷ remote to local

Tip: The `-l` Option

With the **ssh** command, the option `-l` can be used to specify a remote user (as an alternative to the *USER_NAME*`@host` format). With **scp** the option `-l` is used to limit the bandwidth consumed by **scp**.

After the correct password is entered, **scp** starts the data transfer. It displays a progress bar and the time remaining for each file that is copied. Suppress all output with the `-q` option.

scp also provides a recursive copying feature for entire directories. The command

```
tux > scp -r src/ sun:backup/
```

copies the entire contents of the directory `src` including all subdirectories to the `~/backup` directory on the host sun. If this subdirectory does not exist, it is created automatically.

The `-p` option tells **scp** to leave the time stamp of files unchanged. `-C` compresses the data transfer. This minimizes the data volume to transfer, but creates a heavier burden on the processors of both machines.

15.3 **sftp**—Secure File Transfer

15.3.1 Using **sftp**

If you want to copy several files from or to different locations, **sftp** is a convenient alternative to **scp**. It opens a shell with a set of commands similar to a regular FTP shell. Type **help** at the sftp-prompt to get a list of available commands. More details are available from the **sftp** man page.

```
tux > sftp sun
Enter passphrase for key '/home/tux/.ssh/id_rsa':
Connected to sun.
sftp> help
Available commands:
```

```
bye                                 Quit sftp
cd path                             Change remote directory to 'path'
[...]
```

15.3.2 Setting Permissions for File Uploads

As with a regular FTP server, a user cannot only download, but also upload files to a remote machine running an SFTP server by using the **`put`** command. By default the files will be uploaded to the remote host with the same permissions as on the local host. There are two options to automatically alter these permissions:

Setting a umask

A umask works as a filter against the permissions of the original file on the local host. It can only withdraw permissions:

TABLE 15.1:

permissions original	umask	permissions uploaded
0666	0002	0664
0600	0002	0600
0775	0025	0750

To apply a umask on an SFTP server, edit the file `/etc/ssh/sshd_configuration`. Search for the line beginning with `Subsystem sftp` and add the `-u` parameter with the desired setting, for example:

```
Subsystem sftp /usr/lib/ssh/sftp-server -u 0002
```

Explicitly Setting the Permissions

Explicitly setting the permissions sets the same permissions for all files uploaded via SFTP. Specify a three-digit pattern such as `600`, `644`, or `755` with `-u`. When both `-m` and `-u` are specified, `-u` is ignored.

To apply explicit permissions for uploaded files on an SFTP server, edit the file `/etc/ssh/sshd_configuration`. Search for the line beginning with `Subsystem sftp` and add the `-m` parameter with the desired setting, for example:

```
Subsystem sftp /usr/lib/ssh/sftp-server -m 600
```

15.4 The SSH Daemon (sshd)

To work with the SSH client programs `ssh` and `scp`, a server (the SSH daemon) must be running in the background, listening for connections on `TCP/IP port 22`. The daemon generates three key pairs when starting for the first time. Each key pair consists of a private and a public key. Therefore, this procedure is called public key-based. To guarantee the security of the communication via SSH, access to the private key files must be restricted to the system administrator. The file permissions are set accordingly by the default installation. The private keys are only required locally by the SSH daemon and must not be given to anyone else. The public key components (recognizable by the name extension `.pub`) are sent to the client requesting the connection. They are readable for all users.

A connection is initiated by the SSH client. The waiting SSH daemon and the requesting SSH client exchange identification data to compare the protocol and software versions, and to prevent connections through the wrong port. Because a child process of the original SSH daemon replies to the request, several SSH connections can be made simultaneously.

For the communication between SSH server and SSH client, OpenSSH supports versions 1 and 2 of the SSH protocol. Version 2 of the SSH protocol is used by default. Override this to use version 1 of protocol with the `-1` option.

When using version 1 of SSH, the server sends its public host key and a server key, which is regenerated by the SSH daemon every hour. Both allow the SSH client to encrypt a freely chosen session key, which is sent to the SSH server. The SSH client also tells the server which encryption method (cipher) to use. Version 2 of the SSH protocol does not require a server key. Both sides use an algorithm according to Diffie-Hellman to exchange their keys.

The private host and server keys are absolutely required to decrypt the session key and cannot be derived from the public parts. Only the contacted SSH daemon can decrypt the session key using its private keys. This initial connection phase can be watched closely by turning on verbose debugging using the `-v` option of the SSH client.

Tip: Viewing the SSH Daemon Log File

To watch the log entries from the `sshd` use the following command:

```
tux > sudo journalctl -u sshd
```

15.4.1 Maintaining SSH Keys

It is recommended to back up the private and public keys stored in `/etc/ssh/` in a secure, external location. In this way, key modifications can be detected or the old ones can be used again after having installed a new system.

Tip: Existing SSH Host Keys

If you install openSUSE Leap on a machine with existing Linux installations, the installation routine automatically imports the SSH host key with the most recent access time from an existing installation.

When establishing a secure connection with a remote host for the first time, the client stores all public host keys in `~/.ssh/known_hosts`. This prevents any man-in-the-middle attacks—attempts by foreign SSH servers to use spoofed names and IP addresses. Such attacks are detected either by a host key that is not included in `~/.ssh/known_hosts`, or by the server's inability to decrypt the session key in the absence of an appropriate private counterpart.

If the public keys of a host have changed (that needs to be verified before connecting to such a server), the offending keys can be removed with **`ssh-keygen -r`** *`HOSTNAME`*.

15.4.2 Rotating Host Keys

As of version 6.8, OpenSSH comes with a protocol extension that supports host key rotation. It makes sense to replace keys, if you are still using weak keys such as 1024-bit RSA keys. It is strongly recommended to replace such a key and go for 2048-bit DSA keys or something even better. The client will then use the “best” host key.

Tip: Restarting sshd

After installing new host keys on the server, restart sshd.

This protocol extension can inform a client of all the new host keys on the server, if the user initiates a connection with **`ssh`**. Then, the software on the client updates `~/.ssh/known_hosts`, and the user is not required to accept new keys of previously known and trusted hosts manually. The local `known_hosts` file will contain all the host keys of the remote hosts, in addition to the one that authenticated the host during this session.

Once the administrator of the server knows that all the clients have fetched the new keys, they can remove the old keys. The protocol extension ensures that the obsolete keys will be removed from the client's configuration, too. The key removal occurs while initiating an **ssh** session.

For more information, see:

- http://blog.djm.net.au/2015/02/key-rotation-in-openssh-68.html
- http://heise.de/-2540907 („Endlich neue Schlüssel für SSH-Server“, German only)

15.5 SSH Authentication Mechanisms

In its simplest form, authentication is done by entering the user's password just as if logging in locally. However, having to memorize passwords of several users on remote machines is inefficient. What is more, these passwords may change. On the other hand—when granting `root` access—an administrator needs to be able to quickly revoke such a permission without having to change the `root` password.

To accomplish a login that does not require to enter the remote user's password, SSH uses another key pair, which needs to be generated by the user. It consists of a public (`id_rsa.pub` or `id_dsa.pub`) and a private key (`id_rsa` or `id_dsa`).

To be able to log in without having to specify the remote user's password, the public key of the “SSH user” must be in `~/.ssh/authorized_keys`. This approach also ensures that the remote user has got full control: adding the key requires the remote user's password and removing the key revokes the permission to log in from remote.

For maximum security such a key should be protected by a passphrase which needs to be entered every time you use **ssh**, **scp**, or **sftp**. Contrary to the simple authentication, this passphrase is independent from the remote user and therefore always the same.

An alternative to the key-based authentication described above, SSH also offers a host-based authentication. With host-based authentication, users on a trusted host can log in to another host on which this feature is enabled using the same user name. openSUSE Leap is set up for using key-based authentication, covering setting up host-based authentication on openSUSE Leap is beyond the scope of this manual.

Note: File Permissions for Host-Based Authentication

If the host-based authentication is to be used, the file `/usr/lib/ssh/ssh-keysign` (32-bit systems) or `/usr/lib64/ssh/ssh-keysign` (64-bit systems) should have the setuid bit set, which is not the default setting in openSUSE Leap. In such case, set the file permissions manually. You should use `/etc/permissions.local` for this purpose, to make sure that the setuid bit is preserved after security updates of `openssh`.

15.5.1 Generating an SSH Key

1. To generate a key with default parameters (RSA, 2048 bits), enter the command **`ssh-keygen`**.
2. Accept the default location to store the key (`~/.ssh/id_rsa`) by pressing `Enter` (strongly recommended) or enter an alternative location.
3. Enter a passphrase consisting of 10 to 30 characters. The same rules as for creating safe passwords apply. It is strongly advised to refrain from specifying no passphrase.

You should make absolutely sure that the private key is not accessible by anyone other than yourself (always set its permissions to `0600`). The private key must never fall into the hands of another person.

To change the password of an existing key pair, use the command **`ssh-keygen -p`**.

15.5.2 Copying an SSH Key

To copy a public SSH key to `~/.ssh/authorized_keys` of a user on a remote machine, use the command **`ssh-copy-id`**. To copy your personal key stored under `~/.ssh/id_rsa.pub` you may use the short form. To copy DSA keys or keys of other users, you need to specify the path:

```
tux > ~/.ssh/id_rsa.pub
ssh-copy-id -i tux@sun

tux > ~/.ssh/id_dsa.pub
ssh-copy-id -i ~/.ssh/id_dsa.pub  tux@sun

tux > ~notme/.ssh/id_rsa.pub
```

```
ssh-copy-id -i ~notme/.ssh/id_rsa.pub  tux@sun
```

To successfully copy the key, you need to enter the remote user's password. To remove an existing key, manually edit `~/.ssh/authorized_keys`.

15.5.3 Using the `ssh-agent`

When doing lots of secure shell operations it is cumbersome to type the SSH passphrase for each such operation. Therefore, the SSH package provides another tool, **`ssh-agent`**, which retains the private keys for the duration of an X or terminal session. All other windows or programs are started as clients to the **`ssh-agent`**. By starting the agent, a set of environment variables is set, which will be used by **`ssh`**, **`scp`**, or **`sftp`** to locate the agent for automatic login. See the **`ssh-agent`** man page for details.

After the **`ssh-agent`** is started, you need to add your keys by using **`ssh-add`**. It will prompt for the passphrase. After the password has been provided once, you can use the secure shell commands within the running session without having to authenticate again.

15.5.3.1 Using `ssh-agent` in an X Session

On openSUSE Leap, the **`ssh-agent`** is automatically started by the GNOME display manager. To also invoke **`ssh-add`** to add your keys to the agent at the beginning of an X session, do the following:

1. Log in as the desired user and check whether the file `~/.xinitrc` exists.
2. If it does not exist, use an existing template or copy it from `/etc/skel`:

   ```
   if [ -f ~/.xinitrc.template ]; then mv ~/.xinitrc.template ~/.xinitrc; \
   else cp /etc/skel/.xinitrc.template ~/.xinitrc; fi
   ```

3. If you have copied the template, search for the following lines and uncomment them. If `~/.xinitrc` already existed, add the following lines (without comment signs).

   ```
   # if test -S "$SSH_AUTH_SOCK" -a -x "$SSH_ASKPASS"; then
   #       ssh-add < /dev/null
   # fi
   ```

4. When starting a new X session, you will be prompted for your SSH passphrase.

15.5.3.2 Using `ssh-agent` in a Terminal Session

In a terminal session you need to manually start the **`ssh-agent`** and then call **`ssh-add`** afterward. There are two ways to start the agent. The first example given below starts a new Bash shell on top of your existing shell. The second example starts the agent in the existing shell and modifies the environment as needed.

```
tux > ssh-agent -s /bin/bash
eval $(ssh-agent)
```

After the agent has been started, run **`ssh-add`** to provide the agent with your keys.

15.6 Port Forwarding

`ssh` can also be used to redirect TCP/IP connections. This feature, also called `SSH tunneling`, redirects TCP connections to a certain port to another machine via an encrypted channel.

With the following command, any connection directed to jupiter port 25 (SMTP) is redirected to the SMTP port on sun. This is especially useful for those using SMTP servers without SMTP-AUTH or POP-before-SMTP features. From any arbitrary location connected to a network, e-mail can be transferred to the "home" mail server for delivery.

```
root # ssh -L 25:sun:25 jupiter
```

Similarly, all POP3 requests (port 110) on jupiter can be forwarded to the POP3 port of sun with this command:

```
root # ssh -L 110:sun:110 jupiter
```

Both commands must be executed as `root`, because the connection is made to privileged local ports. E-mail is sent and retrieved by normal users in an existing SSH connection. The SMTP and POP3 host must be set to `localhost` for this to work. Additional information can be found in the manual pages for each of the programs described above and in the OpenSSH package documentation under `/usr/share/doc/packages/openssh`.

15.7 Adding and Removing Public Keys on an Installed System

In some environment, it is convenient or necessary to log in over SSH. As such, the user needs to provide a public SSH key. To add or remove an SSH key, proceed as follows:

1. Open YaST.
2. Under *Security and Users*, open the *User and Group Management* module.
3. Select the user you want to change and press *Edit*.
4. Switch to the *SSH Public Key* tab.
5. Add or remove your public key(s). If you add a public SSH key, look for the file extension `.pub`.
6. Confirm with *Ok*.

Your public SSH key is saved in `~/.ssh/authorized_keys`.

15.8 For More Information

http://www.openssh.com
: The home page of OpenSSH

http://en.wikibooks.org/wiki/OpenSSH
: The OpenSSH Wikibook

`man sshd`
: The man page of the OpenSSH daemon

`man ssh_config`
: The man page of the OpenSSH SSH client configuration files

man scp,
man sftp,
man slogin,
man ssh,
man ssh-add,
man ssh-agent,
man ssh-copy-id,
man ssh-keyconvert,
man ssh-keygen,
man ssh-keyscan

Man pages of several binary files to securely copy files (**scp**, **sftp**), to log in (**slogin**, **ssh**), and to manage keys.

`/usr/share/doc/packages/openssh/README.SUSE`,
`/usr/share/doc/packages/openssh/README.FIPS`

SUSE package specific documentation; changes in defaults with respect to upstream, notes on FIPS mode etc.

16 Masquerading and Firewalls

Whenever Linux is used in a network environment, you can use the kernel functions that allow the manipulation of network packets to maintain a separation between internal and external network areas. The Linux `netfilter` framework provides the means to establish an effective firewall that keeps different networks apart. Using iptables—a generic table structure for the definition of rule sets—precisely controls the packets allowed to pass a network interface. Such a packet filter can be set up using `firewalld` and its graphical interface **`firewall-config`**.

16.1 Packet Filtering with iptables

This section discusses the low-level details of packet filtering. The components `netfilter` and `iptables` are responsible for the filtering and manipulation of network packets and for network address translation (NAT). The filtering criteria and any actions associated with them are stored in chains, which must be matched one after another by individual network packets as they arrive. The chains to match are stored in tables. The **`iptables`** command allows you to alter these tables and rule sets.

The Linux kernel maintains three tables, each for a particular category of functions of the packet filter:

filter

This table holds the bulk of the filter rules, because it implements the *packet filtering* mechanism in the stricter sense, which determines whether packets are let through (`ACCEPT`) or discarded (`DROP`), for example.

nat

This table defines any changes to the source and target addresses of packets. Using these functions also allows you to implement *masquerading*, which is a special case of NAT used to link a private network with the Internet.

mangle

The rules held in this table make it possible to manipulate values stored in IP headers (such as the type of service).

These tables contain several predefined chains to match packets:

PREROUTING

This chain is applied to all incoming packets.

INPUT

This chain is applied to packets destined for the system's internal processes.

FORWARD

This chain is applied to packets that are only routed through the system.

OUTPUT

This chain is applied to packets originating from the system itself.

POSTROUTING

This chain is applied to all outgoing packets.

Figure 16.1, "iptables: A Packet's Possible Paths" illustrates the paths along which a network packet may travel on a given system. For the sake of simplicity, the figure lists tables as parts of chains, but in reality these chains are held within the tables themselves.

In the simplest case, an incoming packet destined for the system itself arrives at the `eth0` interface. The packet is first referred to the `PREROUTING` chain of the `mangle` table then to the `PREROUTING` chain of the `nat` table. The following step, concerning the routing of the packet, determines that the actual target of the packet is a process of the system itself. After passing the `INPUT` chains of the `mangle` and the `filter` table, the packet finally reaches its target, provided that the rules of the `filter` table allow this.

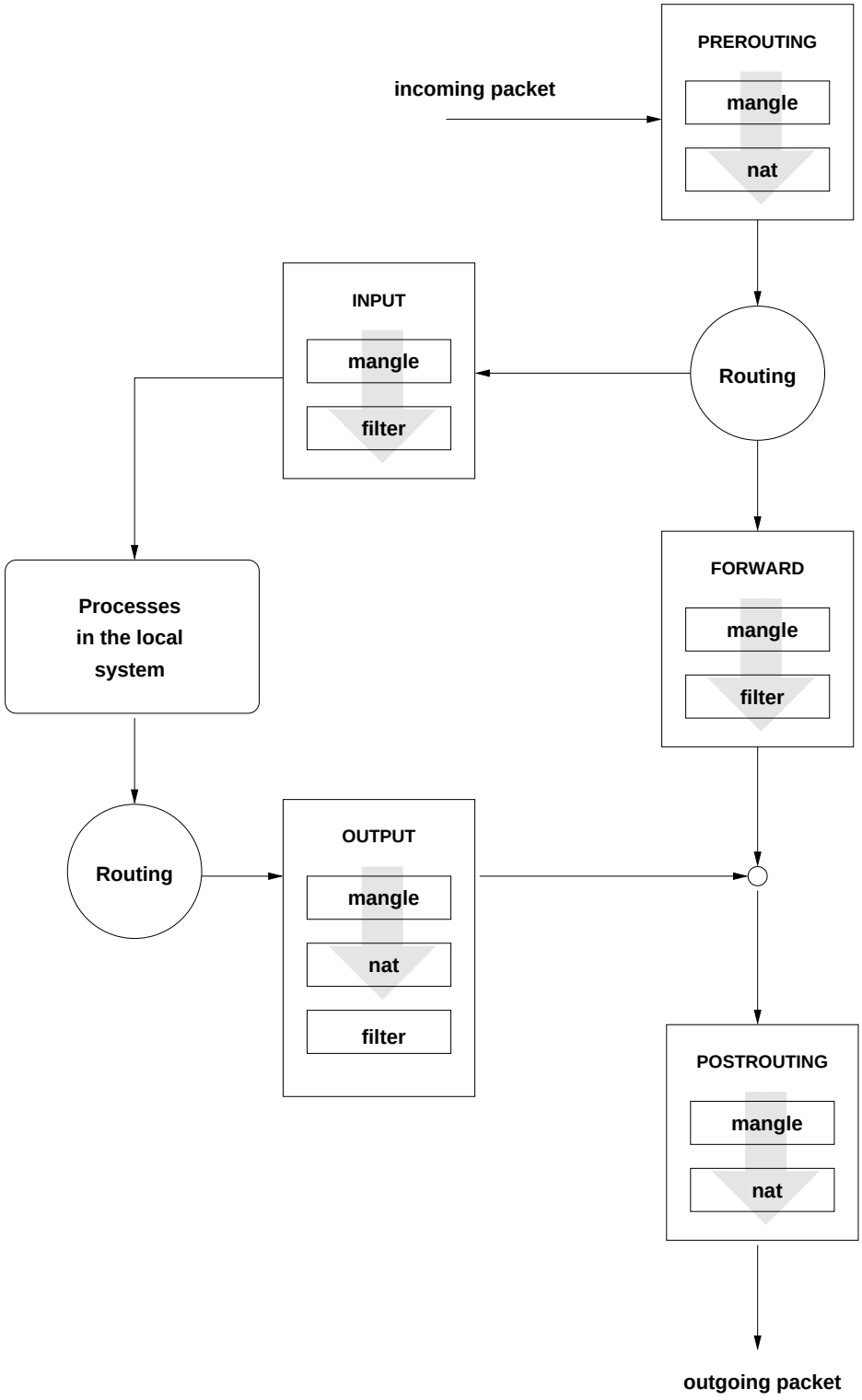


FIGURE 16.1: IPTABLES: A PACKET'S POSSIBLE PATHS

16.2 Masquerading Basics

Masquerading is the Linux-specific form of NAT (network address translation) and can be used to connect a small LAN with the Internet. LAN hosts use IP addresses from the private range (see *Book "Reference", Chapter 13 "Basic Networking", Section 13.1.2 "Netmasks and Routing"*) and on the Internet official IP addresses are used. To be able to connect to the Internet, a LAN host's private address is translated to an official one. This is done on the router, which acts as the gateway between the LAN and the Internet. The underlying principle is a simple one: The router has more than one network interface, typically a network card and a separate interface connecting with the Internet. While the latter links the router with the outside world, one or several others link it with the LAN hosts. With these hosts in the local network connected to the network card (such as `eth0`) of the router, they can send any packets not destined for the local network to their default gateway or router.

Important: Using the Correct Network Mask

When configuring your network, make sure both the broadcast address and the netmask are the same for all local hosts. Failing to do so prevents packets from being routed properly.

As mentioned, whenever one of the LAN hosts sends a packet destined for an Internet address, it goes to the default router. However, the router must be configured before it can forward such packets. For security reasons, this is not enabled in a default installation. To enable it, add the line `net.ipv4.ip_forward = 1` in the file `/etc/sysctl.conf`. Alternatively do this via YaST, for example by calling **`yast routing ip-forwarding on`**.

The target host of the connection can see your router, but knows nothing about the host in your internal network where the packets originated. This is why the technique is called masquerading. Because of the address translation, the router is the first destination of any reply packets. The router must identify these incoming packets and translate their target addresses, so packets can be forwarded to the correct host in the local network.

With the routing of inbound traffic depending on the masquerading table, there is no way to open a connection to an internal host from the outside. For such a connection, there would be no entry in the table. In addition, any connection already established has a status entry assigned to it in the table, so the entry cannot be used by another connection.

As a consequence of all this, you might experience some problems with several application protocols, such as ICQ, cucme, IRC (DCC, CTCP), and FTP (in PORT mode). Web browsers, the standard FTP program, and many other programs use the PASV mode. This passive mode is much less problematic as far as packet filtering and masquerading are concerned.

16.3 Firewalling Basics

Firewall is probably the term most widely used to describe a mechanism that controls the data flow between networks. Strictly speaking, the mechanism described in this section is called a *packet filter*. A packet filter regulates the data flow according to certain criteria, such as protocols, ports, and IP addresses. This allows you to block packets that, according to their addresses, are not supposed to reach your network. To allow public access to your Web server, for example, explicitly open the corresponding port. However, a packet filter does not scan the contents of packets with legitimate addresses, such as those directed to your Web server. For example, if incoming packets were intended to compromise a CGI program on your Web server, the packet filter would still let them through.

A more effective but more complex mechanism is the combination of several types of systems, such as a packet filter interacting with an application gateway or proxy. In this case, the packet filter rejects any packets destined for disabled ports. Only packets directed to the application gateway are accepted. This gateway or proxy pretends to be the actual client of the server. In a sense, such a proxy could be considered a masquerading host on the protocol level used by the application. One example for such a proxy is Squid, an HTTP and FTP proxy server. To use Squid, the browser must be configured to communicate via the proxy. Any HTTP pages or FTP files requested are served from the proxy cache and objects not found in the cache are fetched from the Internet by the proxy.

The following section focuses on the packet filter that comes with openSUSE Leap. For further information about packet filtering and firewalling, read the Firewall HOWTO (http://www.tldp.org/HOWTO/Firewall-HOWTO.html).

16.4 `firewalld`

`firewalld` is a daemon that maintains the system's **`iptables`** rules and offers a D-Bus interface for operating on them. It comes with a command line utility **`firewall-cmd`** and a graphical user interface **`firewall-config`** for interacting with it. Since `firewalld` is running in the background and provides a well defined interface it allows other applications to request changes to the iptables rules, for example to set up virtual machine networking.

`firewalld` implements different security zones. A number of predefined zones like `internal` and `public` exist. The administrator can define additional custom zones if desired. Each zone contains its own set of iptables rules. Each network interface is a member of exactly one zone. Individual connections can also be assigned to a zone based on the source addresses.

Each zone represents a certain level of trust. For example the `public` zone is not trusted, because other computers in this network are not under your control (suitable for Internet or wireless hotspot connections). On the other hand the `internal` zone is used for networks that *are* under your control, like a home or company network. By utilizing zones this way, a host can offer different kinds of services to trusted networks and untrusted networks in a defined way.

For more information about the predefined zones and their meaning in `firewalld`, refer to its manual at http://www.firewalld.org/documentation/zone/predefined-zones.html.

Note: No Zone Assigned Behavior

The initial state for network interfaces is to be assigned to no zone at all. In this case the network interface will be implicitly handled in the default zone, which can be determined by calling **`firewall-cmd --get-default-zone`**. If not configured otherwise, the default zone is the `public` zone.

The `firewalld` packet filtering model allows any outgoing connections to pass. Outgoing connections are connections that are actively established by the local host. Incoming connections that are established by remote hosts are blocked if the respective service is not allowed in the zone in question. Therefore, each of the interfaces with incoming traffic must be placed in a suitable zone to allow for the desired services to be accessible. For each of the zones, define the services or protocols you need.

An important concept of `firewalld` is the distinction between two separate configurations: the *runtime* and the *permanent* configuration. The runtime configuration represents the currently active rules, while the permanent configuration represents the saved rules that will be applied when restarting `firewalld`. This allows to add temporary rules that will be discarded after

restarting `firewalld`, or to experiment with new rules while being able to revert back to the original state. When you are changing the configuration, you need to be aware of which configuration you're editing. How this is done is discussed in *Section 16.4.1.2, "Runtime Versus Permanent Configuration"*.

If you want to perform the `firewalld` configuration using the graphical user interface **`firewall-config`** then refer to its documentation (http://www.firewalld.org/documentation/utilities/firewall-config.html). In the following section we will be looking at how to perform typical `firewalld` configuration tasks using **`firewall-cmd`** on the command line.

16.4.1 Configuring the Firewall on the Command Line

16.4.1.1 Firewall Startup

`firewalld` will be installed and enabled by default. It is a regular `systemd` service that can be configured via **`systemctl`** or the YaST Services Manager.

Important: Automatic Firewall Configuration

After the installation, YaST automatically starts `firewalld` and leaves all interfaces in the default `public` zone. If a server application is configured and activated on the system, YaST can adjust the firewall rules via the options *Open Ports on Selected Interface in Firewall* or *Open Ports on Firewall* in the server configuration modules. Some server module dialogs include a *Firewall Details* button for activating additional services and ports.

16.4.1.2 Runtime Versus Permanent Configuration

By default all **`firewall-cmd`** commands operate on the runtime configuration. You can apply most operations to the permanent configuration *only* by adding the `--permanent` parameter. When doing so the change will only affect the permanent configuration and will not be effective immediately in the runtime configuration. There is currently no way to add a rule to both runtime and permanent configurations in a single invocation. To achieve this you can apply all necessary changes to the runtime configuration and when all is working as expected issue the following command:

```
root # firewall-cmd --runtime-to-permanent
```

This will write all current runtime rules into the permanent configuration. Any temporary modifications you or other programs may have made to the firewall in other contexts are made permanent this way. If you're unsure about this, you can also take the opposite approach to be on the safe side: Add new rules to the permanent configuration and reload `firewalld` to make them active.

Note

Some configuration items, like the default zone, are shared by both the runtime and permanent configurations. Changing them will reflect in both configurations at once.

To revert the runtime configuration to the permanent configuration and thereby discard any temporary changes, two possibilities exist, either via the `firewalld` command line interface or via `systemd`:

```
root # firewall-cmd --reload
```

```
root # systemctl reload firewalld
```

For brevity the examples in the following sections will always operate on the runtime configuration, if applicable. Adjust them accordingly if you want to make them permanent.

16.4.1.3 Assignment of Interfaces to Zones

You can list all network interfaces currently assigned to a zone like this:

```
root # firewall-cmd --zone=public --list-interfaces
eth0
```

Similarly you can query which zone a specific interface is assigned to:

```
root # firewall-cmd --get-zone-of-interface=eth0
public
```

The following command lines assign an interface to a zone. The variant using `--add-interface` will only work if `eth0` is not already assigned to another zone. The variant using `--change-interface` will always work, removing `eth0` from its current zone if necessary:

```
root # firewall-cmd --zone=internal --add-interface=eth0
```

```
root # firewall-cmd --zone=internal --change-interface=eth0
```

Any operations without an explicit `--zone` argument will implicitly operate on the default zone. This pair of commands can be used for getting and setting the default zone assignment:

```
root # firewall-cmd --get-default-zone
dmz
root # firewall-cmd --set-default-zone=public
```

Important

Any network interfaces not explicitly assigned to a zone will be automatically part of the default zone. Changing the default zone will reassign all those network interfaces immediately for the permanent and runtime configurations. You should never use a trusted zone like `internal` as the default zone, to avoid unexpected exposure to threats. For example hotplugged network interfaces like USB ethernet interfaces would automatically become part of the trusted zone in such cases.

Also note that interfaces that are not explicitly part of any zone will not appear in the zone interface list. There is currently no command to list unassigned interfaces. Due to this it is best to avoid unassigned network interfaces during regular operation.

16.4.1.4 Making Network Services Accessible

`firewalld` has a concept of *services*. A service consists of definitions of ports and protocols. These definitions logically belong together in the context of a given network service like a Web or mail server protocol. The following commands can be used to get information about predefined services and their details:

```
root # firewall-cmd --get-services
[...] dhcp dhcpv6 dhcpv6-client dns docker-registry [...]
root # firewall-cmd --info-service dhcp
dhcp
  ports: 67/udp
  protocols:
  source-ports:
  modules:
  destination:
```

These service definitions can be used for easily making the associated network functionality accessible in a zone. This command line will open the http Web server port in the internal zone, for example:

```
root # firewall-cmd --add-service=http --zone=internal
```

The removal of a service from a zone is performed using the counterpart command `--remove-service`. You can also define custom services using the `--new-service` subcommand. Refer to http://www.firewalld.org/documentation/howto/add-a-service.html for more details on how to do this.

If you just want to open a single port by number, you can use the following approach. This will open TCP port 8000 in the internal zone:

```
root # firewall-cmd --add-port=8000/tcp --zone=internal
```

For removal use the counterpart command `--remove-port`.

Tip: Temporarily Opening a Service or Port

`firewalld` supports a `--timeout` parameter that allows to open a service or port for a limited time duration. This can be helpful for quick testing and makes sure that closing the service or port will not be forgotten. To allow the `imap` service in the `internal` zone for 5 minutes, you would call

```
root # firewall-cmd --add-service=imap --zone=internal --timeout=5m
```

16.4.1.5 Lockdown Mode

`firewalld` offers a *lockdown mode* that prevents changes to the firewall rules while it is active. Since applications can automatically change the firewall rules via the D-Bus interface, and depending on the PolicyKit rules regular users may be able to do the same, it can be helpful to prevent changes in some situations. You can find more information about this at https://fedoraproject.org/wiki/Features/FirewalldLockdown.

It is important to understand that the lockdown mode feature provides no real security, but merely protection against accidental or benign attempts to change the firewall. The way the lockdown mode is currently implemented in `firewalld` provides no security against malicious intent. as is pointed out at http://seclists.org/oss-sec/2017/q3/139.

16.4.1.6 Adding Custom `iptables` Rules

`firewalld` claims exclusive control over the host's `netfilter` rules. You should never modify firewall rules using other tools like `iptables`. Doing so could confuse `firewalld` and break security or functionality.

If you need to add custom firewall rules that aren't covered by `firewalld` features then there are two ways to do so. To directly pass raw `iptables` syntax you can use the `--direct` option. It expects the table, chain, and priority as initial arguments and the rest of the command line is passed as is to `iptables`. The following example adds a connection tracking rule for the forwarding filter table:

```
root # firewall-cmd  --direct --add-rule ipv4 filter FORWARD 0 -i eth0 -o eth1 \
    -p tcp --dport 80 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
```

Additionally, `firewalld` implements so called *rich rules*, an extended syntax for specifying `iptables` rules in an easier way. You can find the syntax specification at http://www.firewalld.org/documentation/man-pages/firewalld.richlanguage.html. The following example drops all IPv4 packets originating from a certain source address:

```
root # firewall-cmd --zone=public --add-rich-rule='rule family="ipv4" \
    source address="192.168.2.4" drop'
```

16.4.1.7 Routing, Forwarding, and Masquerading

`firewalld` is not designed to run as a fully fledged router. The basic functionality for typical home router setups is available. For a corporate production router you should not use `firewalld`, however, but use dedicated router and firewall devices instead. The following provides just a few pointers on what to look for to utilize routing in `firewalld`:

- First of all IP forwarding needs to be enabled as outlined in *Section 16.2, "Masquerading Basics"*.
- To enable IPv4 masquerading, for example in the `internal` zone, issue the following command.

  ```
  root # firewall-cmd --zone=internal --add-masquerade
  ```

- `firewalld` can also enable port forwarding. The following command will forward local TCP connections on port 80 to another host:

  ```
  root # firewall-cmd --zone=public \
  ```

```
--add-forward-port=port=80:proto=tcp:toport=80:toaddr=192.168.1.10
```

16.4.2 Accessing Services Listening on Dynamic Ports

Some network services do not listen on predefined port numbers. Instead they operate based on the `portmapper` or `rpcbind` protocol. We will use the term `rpcbind` from here on. When one of these services starts, it chooses a random local port and talks to `rpcbind` to make the port number known. `rpcbind` itself is listening on a well known port. Remote systems can then query `rpcbind` about the network services it knows about and on which ports they are listening. Not many programs use this approach anymore today. Popular examples are Network Information Services (NIS; **`ypserv`** and **`ypbind`**) and the Network File System (NFS) version 3.

Note: About NFSv4

The newer NFSv4 only requires the single well known TCP port 2049. For protocol version 4.0 the kernel parameter `fs.nfs.nfs_callback_tcpport` may need to be set to a static port (see *Example 16.1, "Callback Port Configuration for the `nfs` Kernel Module in `/etc/modprobe.d/60-nfs.conf`"*). Starting with protocol version 4.1 this setting has also become unnecessary.

The dynamic nature of the `rpcbind` protocol makes it difficult to make the affected services behind the firewall accessible. `firewalld` does not support these services by itself. For manual configuration, see *Section 16.4.2.1, "Configuring Static Ports"*. Alternatively, openSUSE Leap provides a helper script. For details, see *Section 16.4.2.2, "Using `firewall-rpcbind-helper` for Configuring Static Ports"*.

16.4.2.1 Configuring Static Ports

One possibility is to configure all involved network services to use fixed port numbers. Once this is done, the fixed ports can be opened in `firewalld` and everything should work. The actual port numbers used are at your discretion but should not clash with any well known port numbers assigned to other services. See *Table 16.1, "Important Sysconfig Variables for Static Port Configuration"* for a list of the available configuration items for NIS and NFSv3 services. Note that depending on your actual NIS or NFS configuration, not all of these ports may be required for your setup.

TABLE 16.1: IMPORTANT SYSCONFIG VARIABLES FOR STATIC PORT CONFIGURATION

File Path	Variable Name	Example Value
`/etc/sysconfig/nfs`	MOUNTD_PORT	21001
	STATD_PORT	21002
	LOCKD_TCPPORT	21003
	LOCKD_UDPPORT	21003
	RQUOTAD_PORT	21004
`/etc/sysconfig/ypbind`	YPBIND_OPTIONS	-p 24500
`/etc/sysconfig/ypserv`	YPXFRD_ARGS	-p 24501
	YPSERV_ARGS	-p 24502
	YPPASSWDD_ARGS	-port 24503

You will need to restart any related services that are affected by these static port configurations for the changes to take effect. You can see the currently assigned rpcbind ports by using the command **`rpcinfo -p`**. On success only the statically configured ports should show up there.

Apart from the port configuration for network services running in userspace there are also ports that are used by the Linux kernel directly when it comes to NFS. One of these ports is `nfs_callback_tcpport`. It is only required for NFS protocol versions older than 4.1. There is a sysctl named `fs.nfs.nfs_callback_tcpport` to configure this port. This sysctl node only appears dynamically when NFS mounts are active. Therefore it is best to configure the port via kernel module parameters. This can be achieved by creating a file as shown in *Example 16.1, "Callback Port Configuration for the `nfs` Kernel Module in `/etc/modprobe.d/60-nfs.conf`"*.

EXAMPLE 16.1: CALLBACK PORT CONFIGURATION FOR THE `nfs` KERNEL MODULE IN `/etc/modprobe.d/60-nfs.conf`

```
options nfs callback_tcpport=21005
```

To make this change effective it is easiest to reboot the machine. Otherwise all NFS services need to be stopped and the `nfs` kernel module needs to be reloaded. To verify the active NFS callback port, check the output of **`cat /sys/module/nfs/parameters/callback_tcpport`**.

For easy handling of the now statically configured RPC ports, it is useful to create a new `firewalld` service definition. This service definition will group all related ports and, for example, makes it easy to make them accessible in a specific zone. In *Example 16.2, "Commands to Define a new `firewalld` RPC Service for NFS"* this is done for the NFS ports as they have been configured in the accompanying examples.

EXAMPLE 16.2: COMMANDS TO DEFINE A NEW `firewalld` RPC SERVICE FOR NFS

```
root # firewall-cmd --permanent --new-service=nfs-rpc
root # firewall-cmd --permanent --service=nfs-rpc --set-description="NFS related,
 statically configured RPC ports"
# add UDP and TCP ports for the given sequence
root # for port in 21001 21002 21003 21004; do
    firewall-cmd --permanent --service=nfs-rpc --add-port ${port}/udp --add-port ${port}/
tcp
done
# the callback port is TCP only
root # firewall-cmd --permanent --service=nfs-rpc --add-port 21005/tcp

# show the complete definition of the new custom service
root # firewall-cmd --info-service=nfs-rpc --permanent -v
nfs-rpc
  summary:
  description: NFS and related, statically configured RPC ports
  ports: 4711/tcp 21001/udp 21001/tcp 21002/udp 21002/tcp 21003/udp 21003/tcp 21004/udp
 21004/tcp
  protocols:
  source-ports:
  modules:
  destination:

# reload firewalld to make the new service definition available
root # firewall-cmd --reload

# the new service definition can now be used to open the ports for example in the
 internal zone
root # firewall-cmd --add-service=nfs-rpc --zone=internal
```

16.4.2.2 Using `firewall-rpcbind-helper` for Configuring Static Ports

The steps to configure static ports as shown in the previous section can be simplified by using the SUSE helper tool **`firewall-rpc-helper.py`**. Install it with **`zypper in firewalld-rpcbind-helper`**.

The tool allows interactive configuration of the service patterns discussed in the previous section. It can also display current port assignments and can be used for scripting. For details, see `firewall-rpc-helper.py --help`.

16.5 For More Information

The most up-to-date information and other documentation about the `firewalld` package is found in `/usr/share/doc/packages/firewalld`. The home page of the netfilter and iptables project, http://www.netfilter.org, provides a large collection of documents about iptables in general in many languages.

17 Configuring a VPN Server

Today, Internet connections are cheap and available almost everywhere. However, not all connections are secure. Using a Virtual Private Network (VPN), you can create a secure network within an insecure network such as the Internet or Wi-Fi. It can be implemented in different ways and serves several purposes. In this chapter, we focus on the OpenVPN (http://www.openvpn.net) implementation to link branch offices via secure wide area networks (WANs).

17.1 Conceptual Overview

This section defines some terms regarding VPN and gives a brief overview of some scenarios.

17.1.1 Terminology

Endpoint

The two “ends” of a tunnel, the source or destination client.

Tap Device

A tap device simulates an Ethernet device (layer 2 packets in the OSI model, such as IP packets). A tap device is used for creating a network bridge. It works with Ethernet frames.

Tun Device

A tun device simulates a point-to-point network (layer 3 packets in the OSI model, such as Ethernet frames). A tun device is used with routing and works with IP frames.

Tunnel

Linking two locations through a primarily public network. From a more technical viewpoint, it is a connection between the client's device and the server's device. Usually a tunnel is encrypted, but it does need to be by definition.

17.1.2 VPN Scenarios

Whenever you set up a VPN connection, your IP packets are transferred over a secured *tunnel*. A tunnel can use either a *tun* or *tap* device. They are virtual network kernel drivers which implement the transmission of Ethernet frames or IP frames/packets.

Any user space program, such as OpenVPN, can attach itself to a tun or tap device to receive packets sent by your operating system. The program is also able to write packets to the device.

There are many solutions to set up and build a VPN connection. This section focuses on the OpenVPN package. Compared to other VPN software, OpenVPN can be operated in two modes:

Routed VPN

Routing is an easy solution to set up. It is more efficient and scales better than a bridged VPN. Furthermore, it allows the user to tune MTU (Maximum Transfer Unit) to raise efficiency. However, in a heterogeneous environment, if you do not have a Samba server on the gateway, NetBIOS broadcasts do not work. If you need IPv6, the drivers for the tun devices on both ends must support this protocol explicitly. This scenario is depicted in *Figure 17.1, "Routed VPN"*.

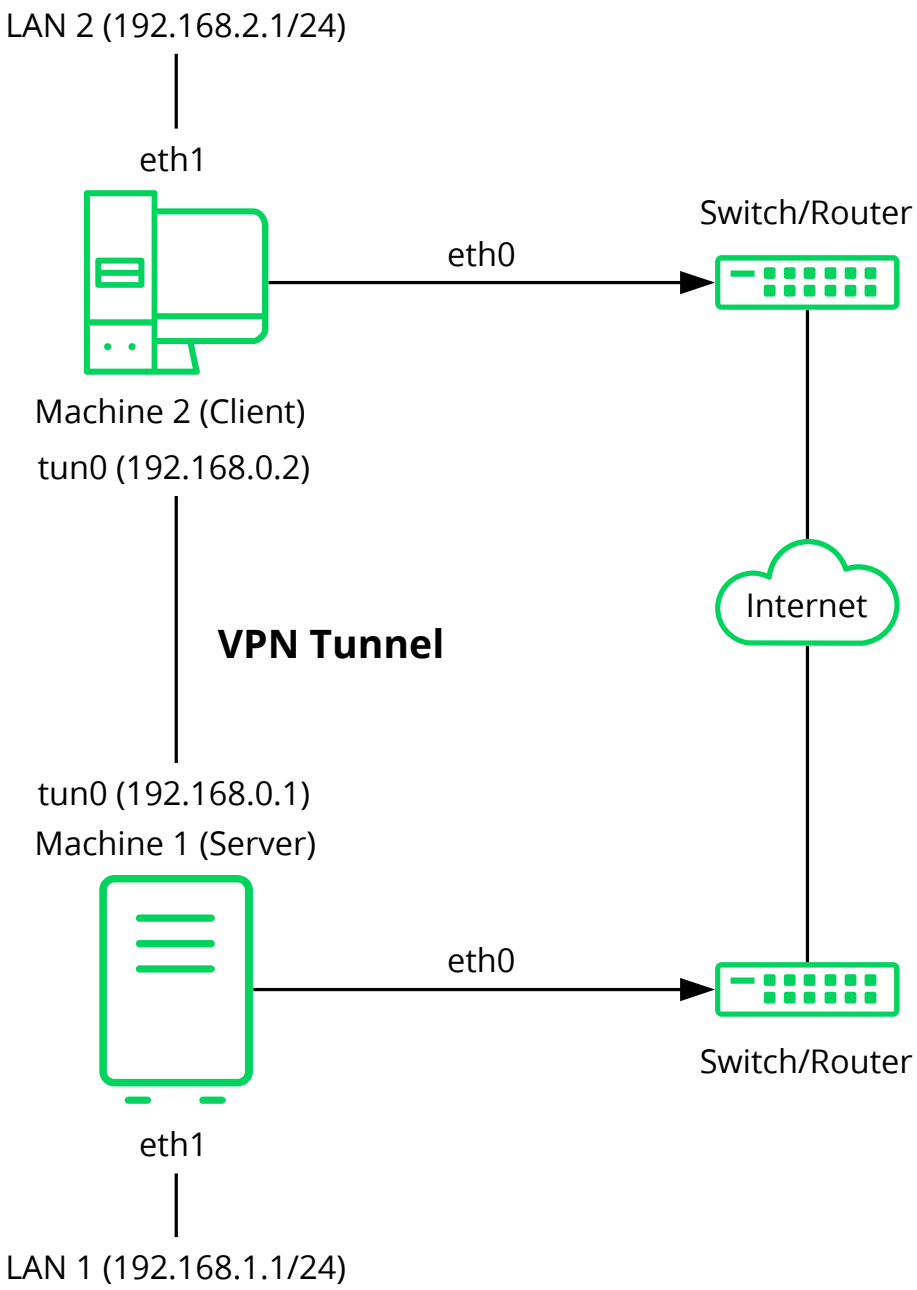


FIGURE 17.1: ROUTED VPN

Bridged VPN

Bridging is a more complex solution. It is recommended when you need to browse Windows file shares across the VPN without setting up a Samba or WINS server. Bridged VPN is also needed to use non-IP protocols (such as IPX) or applications relying on network broadcasts. However, it is less efficient than routed VPN. Another disadvantage is that it does not scale well. This scenario is depicted in the following figures.

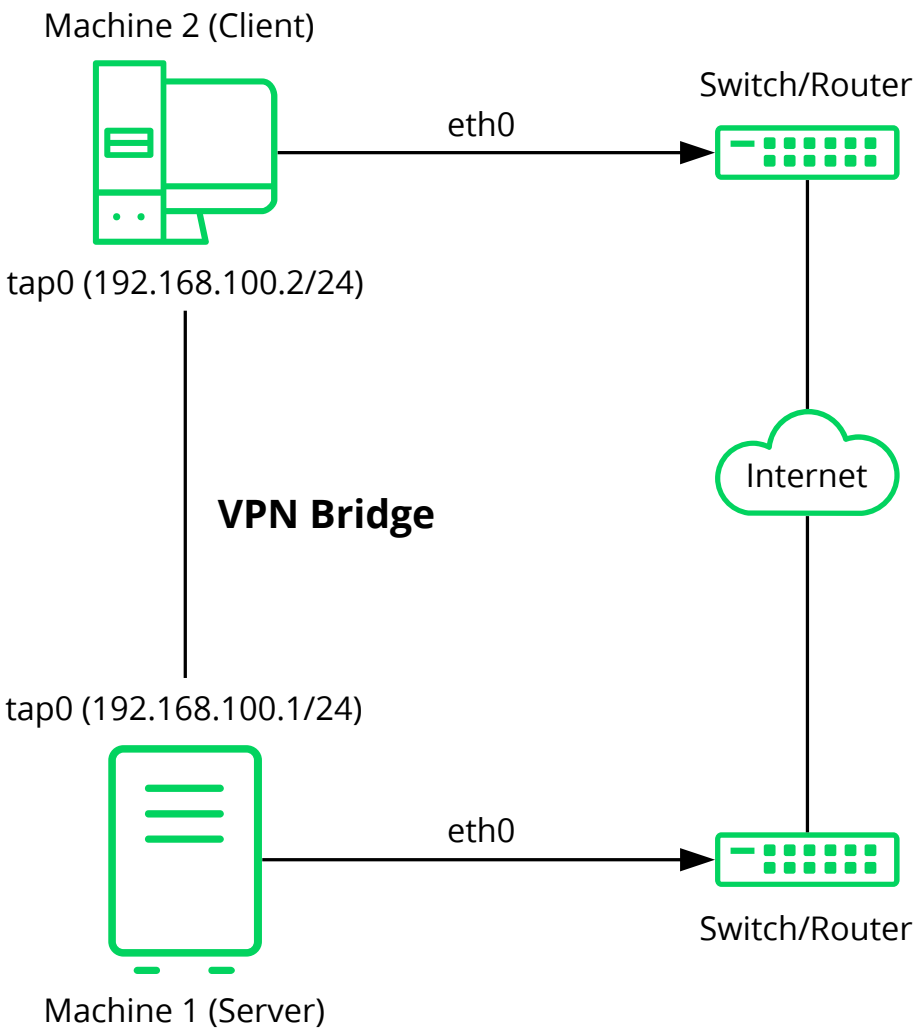


FIGURE 17.2: BRIDGED VPN - SCENARIO 1

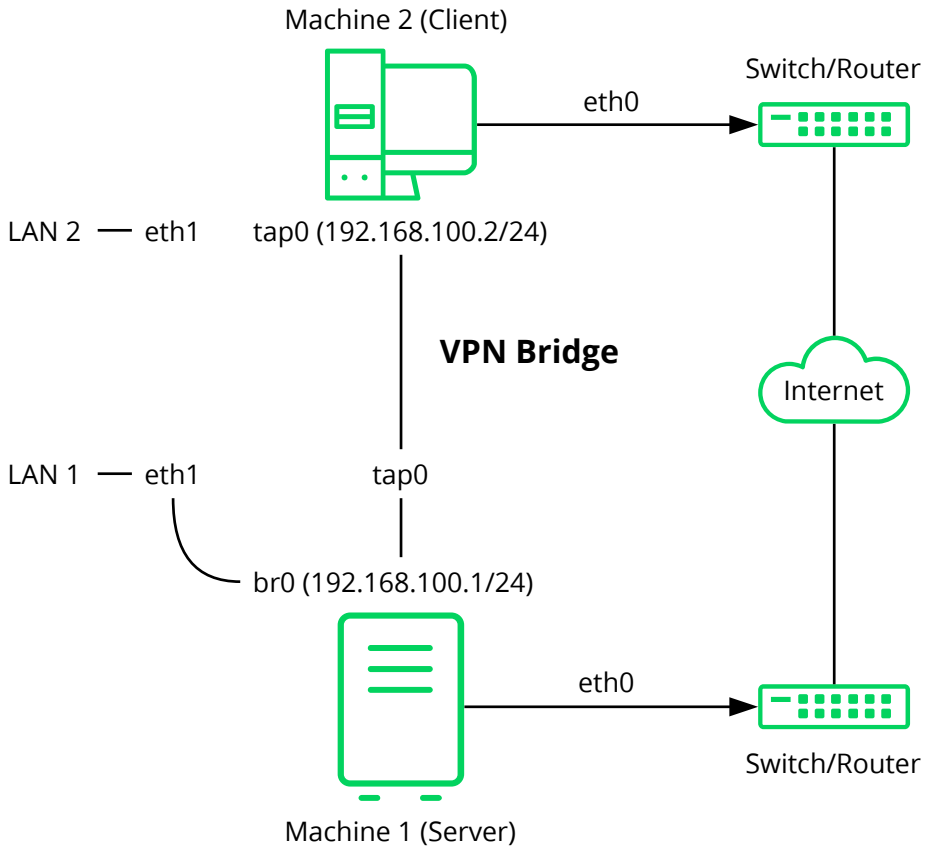


FIGURE 17.3: BRIDGED VPN - SCENARIO 2

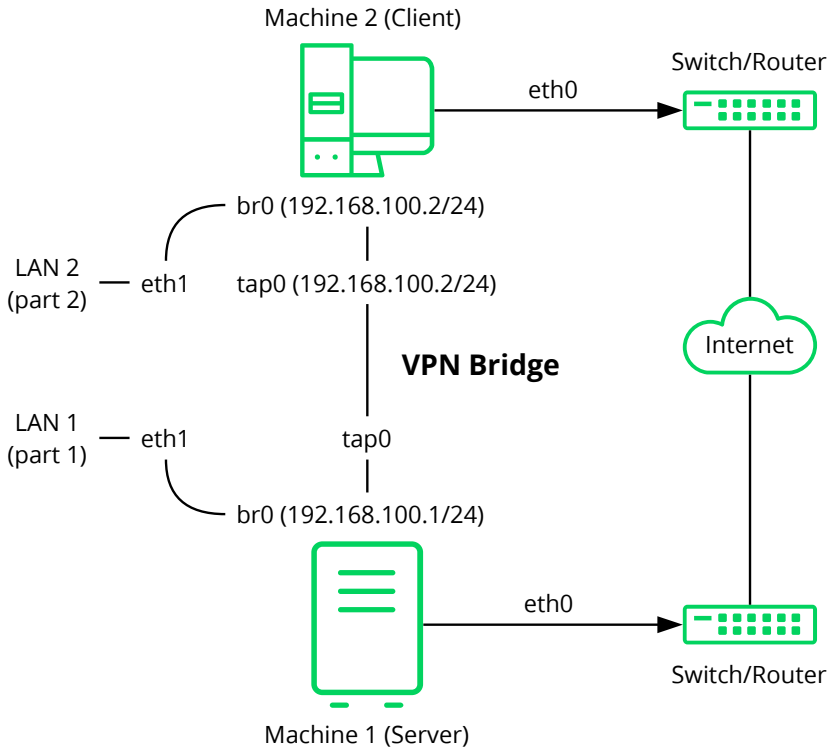


FIGURE 17.4: BRIDGED VPN - SCENARIO 3

The major difference between bridging and routing is that a routed VPN cannot IP-broadcast while a bridged VPN can.

17.2 Setting Up a Simple Test Scenario

In the following example, we will create a point-to-point VPN tunnel. The example demonstrates how to create a VPN tunnel between one client and a server. It is assumed that your VPN server will use private IP addresses like *IP_OF_SERVER* and your client will use the IP address *IP_OF_CLIENT*. Make sure you select addresses which do not conflict with other IP addresses.

Warning: Use Only for Testing

This following scenario is provided as an example meant for familiarizing yourself with VPN technology. *Do not* use this as a real world scenario, as it can compromise the security and safety of your IT infrastructure!

Tip: Names for Configuration File

To simplify working with OpenVPN configuration files, we recommend the following:

- Place your OpenVPN configuration files in the directory `/etc/openvpn`.
- Name your configuration files *`MY_CONFIGURATION`*`.conf`.
- If there are multiple files that belong to the same configuration, place them in a subdirectory like `/etc/openvpn/`*`MY_CONFIGURATION`*.

17.2.1 Configuring the VPN Server

To configure a VPN server, proceed as follows:

PROCEDURE 17.1: VPN SERVER CONFIGURATION

1. Install the package `openvpn` on the machine that will later become your VPN server.
2. Open a shell, become `root` and create the VPN secret key:

```
root # openvpn --genkey --secret /etc/openvpn/secret.key
```

3. Copy the secret key to your client:

```
root # scp /etc/openvpn/secret.key root@IP_OF_CLIENT:/etc/openvpn/
```

4. Create the file `/etc/openvpn/server.conf` with the following content:

```
dev tun
ifconfig IP_OF_SERVER IP_OF_CLIENT
secret secret.key
```

5. Set up a tun device configuration by creating a file called `/etc/sysconfig/network/ifcfg-tun0` with the following content:

```
STARTMODE='manual'
BOOTPROTO='static'
TUNNEL='tun'
TUNNEL_SET_OWNER='nobody'
TUNNEL_SET_GROUP='nobody'
LINK_REQUIRED=no
PRE_UP_SCRIPT='systemd:openvpn@server'
PRE_DOWN_SCRIPT='systemd:openvpn@service'
```

   The notation `openvpn@server` points to the OpenVPN server configuration file located at `/etc/openvpn/server.conf`. For more information, see `/usr/share/doc/packages/openvpn/README.SUSE`.

6. If you use a firewall, start YaST and open UDP port 1194 (*Security and Users* › *Firewall* › *Allowed Services*).

7. Start the OpenVPN server service by setting the tun device to `up`:

```
tux > sudo wicked ifup tun0
```

   You should see the confirmation:

```
tun0            up
```

17.2.2 Configuring the VPN Clients

To configure the VPN client, do the following:

PROCEDURE 17.2: VPN CLIENT CONFIGURATION

1. Install the package `openvpn` on your client VPN machine.

2. Create `/etc/openvpn/client.conf` with the following content:

```
remote DOMAIN_OR_PUBLIC_IP_OF_SERVER
dev tun
ifconfig IP_OF_CLIENT IP_OF_SERVER
secret secret.key
```

   Replace the placeholder *`IP_OF_CLIENT`* in the first line with either the domain name, or the public IP address of your server.

3. Set up a tun device configuration by creating a file called `/etc/sysconfig/network/ifcfg-tun0` with the following content:

```
STARTMODE='manual'
BOOTPROTO='static'
TUNNEL='tun'
TUNNEL_SET_OWNER='nobody'
TUNNEL_SET_GROUP='nobody'
LINK_REQUIRED=no
PRE_UP_SCRIPT='systemd:openvpn@client'
PRE_DOWN_SCRIPT='systemd:openvpn@client'
```

4. If you use a firewall, start YaST and open UDP port 1194 as described in *Step 6* of *Procedure 17.1, "VPN Server Configuration"*.

5. Start the OpenVPN server service by setting the tun device to `up`:

```
tux > sudo wicked ifup tun0
```

You should see the confirmation:

```
tun0            up
```

17.2.3 Testing the VPN Example Scenario

After OpenVPN has successfully started, test the availability of the tun device with the following command:

```
ip addr show tun0
```

To verify the VPN connection, use **`ping`** on both client and server side to see if they can reach each other. Ping the server from the client:

```
ping -I tun0 IP_OF_SERVER
```

Ping the client from the server:

```
ping -I tun0 IP_OF_CLIENT
```

17.3 Setting Up Your VPN Server Using a Certificate Authority

The example in *Section 17.2* is useful for testing, but not for daily work. This section explains how to build a VPN server that allows more than one connection at the same time. This is done with a public key infrastructure (PKI). A PKI consists of a pair of public and private keys for the server and each client, and a master certificate authority (CA), which is used to sign every server and client certificate.

This setup involves the following basic steps:

1. *Section 17.3.1, "Creating Certificates"*
2. *Section 17.3.2, "Configuring the VPN Server"*
3. *Section 17.3.3, "Configuring the VPN Clients"*

17.3.1 Creating Certificates

Before a VPN connection can be established, the client must authenticate the server certificate. Conversely, the server must also authenticate the client certificate. This is called *mutual authentication*. To create such certificates, use the YaST CA module. See *Chapter 18, Managing X.509 Certification* for more details.

To create a VPN root, server, and client CA, proceed as follows:

PROCEDURE 17.3: CREATING A VPN SERVER CERTIFICATE

1. Prepare a common VPN Certificate Authority (CA):
   a. Start the YaST CA module.
   b. Click *Create Root CA*.
   c. Enter a *CA Name* and a *Common Name*, for example `VPN-Server-CA`.
   d. Fill out the other boxes like e-mail addresses, organization, etc. and proceed with *Next*.
   e. Enter your password twice and proceed with *Next*.
   f. Review the summary. YaST displays the current settings for confirmation. Click *Create*. The root CA is created and displayed in the overview.

2. Create a VPN server certificate:

   a. Select the root CA you created in *Step 1* and click *Enter CA*.

   b. When prompted, enter the *CA Password*.

   c. Click the *Certificate* tab and click *Add* › *Add Server Certificate*.

   d. Specify a *Common Name*, for example, `openvpn.example.com` and proceed with *Next*.

   e. Specify your password and confirm it. Then click *Advanced options*.
   Switch to the *Advanced Settings* › *Key Usage* list and check one of the following sets:

      - `digitalSignature` and `keyEncipherment`, or,
      - `digitalSignature` and `keyAgreement`

   Switch to the *Advanced Settings* › *extendedKeyUsage* and type `serverAuth` for a server certificate.

   Important: Avoiding Man-in-the-Middle Attacks

   If you are using the method `remote-cert-tls server` or `remote-cert-tls client` to verify certificates, limit the number of times a key can be used. This mitigates man-in-the-middle attacks.

   For more information, see http://openvpn.net/index.php/open-source/documentation/howto.html#mitm.

   Finish with *Ok* and proceed with *Next*.

   f. Review the summary. YaST displays the current settings for confirmation. Click *Create*. When the VPN server certificate is created, it is displayed in the *Certificates* tab.

3. Create VPN client certificates:

   a. Make sure you are on the *Certificates* tab.

   b. Click *Add* › *Add Client Certificate*.

c. Enter a *Common Name*, for example, `client1.example.com`.

d. Enter the e-mail addresses for your client, for example, `user1@client1.example.com`, and click *Add*. Proceed with *Next*.

e. Enter your password twice and click *Advanced options*.
Switch to *Advanced Settings* › *Key Usage* list and check one of the following flags:

- `digitalSignature` or,
- `keyAgreement` or,
- `digitalSignature` and `keyAgreement`.

Switch to the *Advanced Settings* › *extendedKeyUsage* and type `clientAuth` for a server certificate.

f. Review the summary. YaST displays the current settings for confirmation. Click *Create*. The VPN client certificate is created and is displayed in the *Certificates* tab.

g. If you need certificates for more clients, repeat *Step 3*.

After you have successfully finished *Procedure 17.3, "Creating a VPN Server Certificate"* you have a VPN root CA, a VPN server CA, and one or more VPN client CAs. To finish the task, proceed with the following procedure:

1. Choose the *Certificates* tab.
2. Export the VPN server certificate in two formats: PEM and unencrypted key in PEM.

   a. Select your VPN server certificate (`openvpn.example.com` in our example) and choose *Export* › *Export to File*.

   b. Select *Only the Certificate in PEM Format*, enter your VPN server certificate password and save the file to `/etc/openvpn/server_crt.pem`.

   c. Repeat *Step 2.a* and *Step 2.b*, but choose the format *Only the Key Unencrypted in PEM Format*. Save the file to `/etc/openvpn/server_key.pem`.

3. Export the VPN client certificates and choose an export format, PEM or PKCS12 (preferred). For each client:

   a. Select your VPN client certificate (`client1.example.com` in our example) and choose *Export* › *Export to File*.

   b. Select *Like PKCS12 and Include the CA Chain*, enter your VPN client certificate key password and provide a PKCS12 password. Enter a *File Name*, click *Browse* and save the file to `/etc/openvpn/client1.p12`.

4. Copy the files to your client (in our example, `client1.example.com`).

5. Export the VPN CA (in our example `VPN-Server-CA`):

   a. Switch to the *Description* tab.

   b. Select *Advanced* › *Export to File*.

   c. Mark *Only the Certificate in PEM Format* and save the file to `/etc/openvpn/vpn_ca.pem`.

If desired, the client PKCS12 file can be converted into the PEM format using this command:

```
openssl pkcs12 -in client1.p12 -out client1.pem
```

Enter your client password to create the `client1.pem` file. The PEM file contains the client certificate, client key, and the CA certificate. You can split this combined file using a text editor and create three separate files. The file names can be used for the `ca`, `cert`, and `key` options in the OpenVPN configuration file (see *Example 17.1, "VPN Server Configuration File"*).

17.3.2 Configuring the VPN Server

As the basis of your configuration file, copy `/usr/share/doc/packages/openvpn/sample-config-files/server.conf` to `/etc/openvpn/`. Then customize it to your needs.

EXAMPLE 17.1: VPN SERVER CONFIGURATION FILE

```
# /etc/openvpn/server.conf
port 1194 ❶
proto udp ❷
dev tun0 ❸
```

```
# Security ④

ca    vpn_ca.pem
cert  server_crt.pem
key   server_key.pem

# ns-cert-type server
remote-cert-tls client ⑤
dh   server/dh2048.pem ⑥

server 192.168.1.0 255.255.255.0 ⑦
ifconfig-pool-persist /var/run/openvpn/ipp.txt ⑧

# Privileges ⑨
user nobody
group nobody

# Other configuration ⑩
keepalive 10 120
comp-lzo
persist-key
persist-tun
# status      /var/log/openvpn-status.tun0.log ⑪
# log-append  /var/log/openvpn-server.log ⑫
verb 4
```

① The TCP/UDP port on which OpenVPN listens. You need to open the port in the firewall, see *Chapter 16, Masquerading and Firewalls*. The standard port for VPN is 1194, so you can usually leave that as it is.

② The protocol, either UDP or TCP.

③ The tun or tap device. For the difference between these, see *Section 17.1.1, "Terminology"*.

④ The following lines contain the relative or absolute path to the root server CA certificate (`ca`), the root CA key (`cert`), and the private server key (`key`). These were generated in *Section 17.3.1, "Creating Certificates"*.

⑤ Require that peer certificates have been signed with an explicit key usage and extended key usage based on RFC3280 TLS rules. There is a description of how to make a server use this explicit key in *Procedure 17.3, "Creating a VPN Server Certificate"*.

⑥ The Diffie-Hellman parameters. Create the required file with the following command:

```
openssl dhparam -out /etc/openvpn/dh2048.pem 2048
```

⑦ Supplies a VPN subnet. The server can be reached by `192.168.1.1`.

8 Records a mapping of clients and its virtual IP address in the given file. Useful when the server goes down and (after the restart) the clients get their previously assigned IP address.

9 For security reasons, run the OpenVPN daemon with reduced privileges. To do so, specify that it should use the group and user `nobody`.

10 Several other configuration options—see the comment in the example configuration file: `/usr/share/doc/packages/openvpn/sample-config-files`.

11 Enable this option to write short status updates with statistical data ("operational status dump") to the named file. By default, this is not enabled.
All output is written to the system journal which can be displayed with **`journalctl`**. If you have more than one configuration file (for example, one for home and another for work), it is recommended to include the device name into the file name. This avoids overwriting output files accidentally. In this case, it is `tun0`, taken from the `dev` directive—see 3.

12 By default, log messages go to syslog. Overwrite this behavior by removing the hash character. In that case, all messages go to `/var/log/openvpn-server.log`. Do not forget to configure a logrotate service. See **`man 8 logrotate`** for further details.

After having completed this configuration, you can see log messages of your OpenVPN server under `/var/log/openvpn.log`. After having started it for the first time, it should finish with:

```
... Initialization Sequence Completed
```

If you do not see this message, check the log carefully for any hints of what is wrong in your configuration file.

17.3.3 Configuring the VPN Clients

As the basis of your configuration file, copy `/usr/share/doc/packages/openvpn/sample-config-files/client.conf` to `/etc/openvpn/`. Then customize it to your needs.

EXAMPLE 17.2: VPN CLIENT CONFIGURATION FILE

```
# /etc/openvpn/client.conf
client 1
dev tun 2
proto udp 3
remote IP_OR_HOST_NAME 1194 4
resolv-retry infinite
nobind

remote-cert-tls server 5
```

```
# Privileges ❻
user nobody
group nobody

# Try to preserve some state across restarts.
persist-key
persist-tun

# Security ❼
pkcs12 client1.p12

comp-lzo ❽
```

❶ Specifies that this machine is a client.

❷ The network device. Both clients and server must use the same device.

❸ The protocol. Use the same settings as on the server.

❺ This is security option for clients which ensures that the host they connect to is a designated server.

❹ Replace the placeholder *IP_OR_HOST_NAME* with the respective host name or IP address of your VPN server. After the host name, the port of the server is given. You can have multiple lines of `remote` entries pointing to different VPN servers. This is useful for load balancing between different VPN servers.

❻ For security reasons, run the OpenVPN daemon with reduced privileges. To do so, specify that it should use the group and user `nobody`.

❼ Contains the client files. For security reasons, use a separate pair of files for each client.

❽ Turn on compression. Only use this parameter if compression is enabled on the server as well.

17.4 Setting Up a VPN Server or Client Using YaST

You can also use YaST to set up a VPN server. However, the YaST module does not support OpenVPN. Instead, it provides support for the IPsec protocol (as implemented in the software StrongSwan). Like OpenVPN, IPsec is a widely supported VPN scheme.

PROCEDURE 17.4: SETTING UP AN IPSEC SERVER

1. To start the YaST VPN module, select *Applications* › *VPN Gateways and Clients*.
2. Under *Global Configuration*, activate *Enable VPN Daemon*.

3. To create a new VPN, click *New VPN*, then enter a name for the connection.
4. Under *Type*, select *Gateway (Server)*.
5. Then choose the scenario:

   - The scenarios *Secure communication with a pre-shared key* and *Secure communication with a certificate* are best suited to Linux client setups.
   - The scenario *Provide access to Android, iOS, Mac OS X clients* sets up a configuration that is natively supported by modern versions of Android, iOS, and macOS. It is based on a pre-shared key setup with an additional user name and password authentication.
   - The scenario *Provide access to Windows 7, Windows 8 clients* is a configuration that is natively supported by Windows and BlackBerry devices. It is based on a certificate setup with an additional user name and password authentication.

   For this example, choose *Secure communication with a pre-shared key*.
6. To specify the key, click *Edit Credentials*. Activate *Show key*, then type the secret key. Confirm with *OK*.
7. Choose whether and how to limit access within your VPN under *Provide VPN clients access to*. To enable only certain IP ranges, specify these in CIDR format, separated by commas in *Limited CIDRs*. For more information about the CIDR format, see https://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing.
8. Under *Clients' address pool*, specify the format of IP addresses your VPN should provide to its clients.
9. To finish, click *OK*. The YaST VPN module will now automatically add and enable firewall rules to allow clients to connect to the new VPN.
   To view the connection status, in the following confirmation window, click *Yes*. You will then see the output of `systemctl status` for your VPN, which allows you to check if the VPN is running and configured correctly.

17.5 For More Information

For more information on setting up a VPN connection using NetworkManager, see *Book "Reference", Chapter 28 "Using NetworkManager", Section 28.3.4 "NetworkManager and VPN"*.

For more information about VPN in general, see:

- http://www.openvpn.net: the OpenVPN home page
- `man openvpn`
- `/usr/share/doc/packages/openvpn/sample-config-files/`: example configuration files for different scenarios.
- `/usr/src/linux/Documentation/networking/tuntap.txt`, to install the `kernel-source` package.

18 Managing X.509 Certification

An increasing number of authentication mechanisms are based on cryptographic procedures. Digital certificates that assign cryptographic keys to their owners play an important role in this context. These certificates are used for communication and can also be found, for example, on company ID cards. The generation and administration of certificates is mostly handled by official institutions that offer this as a commercial service. In some cases, however, it may make sense to carry out these tasks yourself. For example, if a company does not want to pass personal data to third parties.

YaST provides two modules for certification, which offer basic management functions for digital X.509 certificates. The following sections explain the basics of digital certification and how to use YaST to create and administer certificates of this type.

18.1 The Principles of Digital Certification

Digital certification uses cryptographic processes to encrypt and protect data from access by unauthorized people. The user data is encrypted using a second data record, or *key*. The key is applied to the user data in a mathematical process, producing an altered data record in which the original content can no longer be identified. Asymmetrical encryption is now in general use (*public key method*). Keys always occur in pairs:

Private Key

The private key must be kept safely by the key owner. Accidental publication of the private key compromises the key pair and renders it useless.

Public Key

The key owner circulates the public key for use by third parties.

18.1.1 Key Authenticity

Because the public key process is in widespread use, there are many public keys in circulation. Successful use of this system requires that every user be sure that a public key actually belongs to the assumed owner. The assignment of users to public keys is confirmed by trustworthy organizations with public key certificates. Such certificates contain the name of the key owner, the corresponding public key, and the electronic signature of the person issuing the certificate.

Trustworthy organizations that issue and sign public key certificates are usually part of a certification infrastructure. This is responsible for the other aspects of certificate management, such as publication, withdrawal, and renewal of certificates. An infrastructure of this kind is generally called a *public key infrastructure* or *PKI*. One familiar PKI is the *OpenPGP* standard in which users publish their certificates themselves without central authorization points. These certificates become trustworthy when signed by other parties in the "web of trust."

The *X.509 Public Key Infrastructure* (PKIX) is an alternative model defined by the *IETF* (Internet Engineering Task Force) that serves as a model for almost all publicly-used PKIs today. In this model, authentication is made by *certificate authorities* (CA) in a hierarchical tree structure. The root of the tree is the root CA, which certifies all sub-CAs. The lowest level of sub-CAs issue user certificates. The user certificates are trustworthy by certification that can be traced to the root CA.

The security of such a PKI depends on the trustworthiness of the CA certificates. To make certification practices clear to PKI customers, the PKI operator defines a *certification practice statement* (CPS) that defines the procedures for certificate management. This should ensure that the PKI only issues trustworthy certificates.

18.1.2 X.509 Certificates

An X.509 certificate is a data structure with several fixed fields and, optionally, additional extensions. The fixed fields mainly contain the name of the key owner, the public key, and the data relating to the issuing CA (name and signature). For security reasons, a certificate should only have a limited period of validity, so a field is also provided for this date. The CA guarantees the validity of the certificate in the specified period. The CPS usually requires the PKI (the issuing CA) to create and distribute a new certificate before expiration.

The extensions can contain any additional information. An application is only required to be able to evaluate an extension if it is identified as *critical*. If an application does not recognize a critical extension, it must reject the certificate. Some extensions are only useful for a specific application, such as signature or encryption.

Table 18.1 shows the fields of a basic X.509 certificate in version 3.

TABLE 18.1: X.509V3 CERTIFICATE

Field	Content
Version	The version of the certificate, for example, v3
Serial Number	Unique certificate ID (an integer)
Signature	The ID of the algorithm used to sign the certificate
Issuer	Unique name (DN) of the issuing authority (CA)
Validity	Period of validity
Subject	Unique name (DN) of the owner
Subject Public Key Info	Public key of the owner and the ID of the algorithm
Issuer Unique ID	Unique ID of the issuing CA (optional)
Subject Unique ID	Unique ID of the owner (optional)
Extensions	Optional additional information, such as “KeyUsage” or “BasicConstraints”

18.1.3 Blocking X.509 Certificates

If a certificate becomes untrustworthy before it has expired, it must be blocked immediately. This can become necessary if, for example, the private key has accidentally been made public. Blocking certificates is especially important if the private key belongs to a CA rather than a

user certificate. In this case, all user certificates issued by the relevant CA must be blocked immediately. If a certificate is blocked, the PKI (the responsible CA) must make this information available to all those involved using a *certificate revocation list* (CRL).

These lists are supplied by the CA to public CRL distribution points (CDPs) at regular intervals. The CDP can optionally be named as an extension in the certificate, so a checker can fetch a current CRL for validation purposes. One way to do this is the *online certificate status protocol* (OCSP). The authenticity of the CRLs is ensured with the signature of the issuing CA. *Table 18.2* shows the basic parts of a X.509 CRL.

TABLE 18.2: X.509 CERTIFICATE REVOCATION LIST (CRL)

Field	Content
Version	The version of the CRL, such as v2
Signature	The ID of the algorithm used to sign the CRL
Issuer	Unique name (DN) of the publisher of the CRL (usually the issuing CA)
This Update	Time of publication (date, time) of this CRL
Next Update	Time of publication (date, time) of the next CRL
List of revoked certificates	Every entry contains the serial number of the certificate, the time of revocation, and optional extensions (CRL entry extensions)
Extensions	Optional CRL extensions

18.1.4 Repository for Certificates and CRLs

The certificates and CRLs for a CA must be made publicly accessible using a *repository*. Because the signature protects the certificates and CRLs from being forged, the repository itself does not need to be secured in a special way. Instead, it tries to grant the simplest and fastest access possible. For this reason, certificates are often provided on an LDAP or HTTP server. Find explanations about LDAP in *Chapter 5, LDAP—A Directory Service*. *Book "*Reference*", Chapter 24 "The Apache HTTP Server"* contains information about the HTTP server.

18.1.5 Proprietary PKI

YaST contains modules for the basic management of X.509 certificates. This mainly involves the creation of CAs, sub-CAs, and their certificates. The services of a PKI go far beyond simply creating and distributing certificates and CRLs. The operation of a PKI requires a well-conceived administrative infrastructure allowing continuous update of certificates and CRLs. This infrastructure is provided by commercial PKI products and can also be partly automated. YaST provides tools for creating and distributing CAs and certificates, but cannot currently offer this background infrastructure. To set up a small PKI, you can use the available YaST modules. However, you should use commercial products to set up an “official” or commercial PKI.

18.2 YaST Modules for CA Management

YaST provides two modules for basic CA management. The primary management tasks with these modules are explained here.

18.2.1 Creating a Root CA

The first step when setting up a PKI is to create a root CA. Do the following:

1. Start YaST and go to *Security and Users* › *CA Management*.
2. Click *Create Root CA*.

3. Enter the basic data for the CA in the first dialog, shown in *Figure 18.1*. The text boxes have the following meanings:

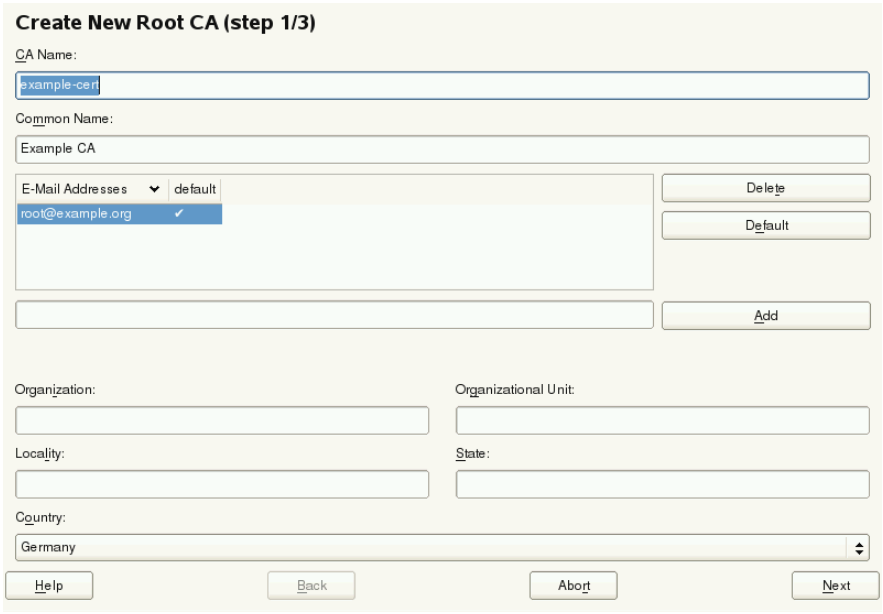


FIGURE 18.1: YAST CA MODULE—BASIC DATA FOR A ROOT CA

CA Name

Enter the technical name of the CA. Directory names, among other things, are derived from this name, which is why only the characters listed in the help can be used. The technical name is also displayed in the overview when the module is started.

Common Name

Enter the name for use in referring to the CA.

E-Mail Addresses

Several e-mail addresses can be entered that can be seen by the CA user. This can be helpful for inquiries.

Country

Select the country where the CA is operated.

Organization, Organizational Unit, Locality, State

Optional values

Proceed with *Next*.

4. Enter a password in the second dialog. This password is always required when using the CA—when creating a sub-CA or generating certificates. The text boxes have the following meaning:

   Key Length

   Key Length contains a meaningful default and does not generally need to be changed unless an application cannot deal with this key length. The higher the number the more secure your password is.

   Valid Period (days)

   The *Valid Period* in the case of a CA defaults to 3650 days (roughly ten years). This long period makes sense because the replacement of a deleted CA involves an enormous administrative effort.

   Clicking *Advanced Options* opens a dialog for setting different attributes from the X.509 extensions (*Figure 18.4, "YaST CA Module—Extended Settings"*). These values have rational default settings and should only be changed if you are really sure of what you are doing. Proceed with *Next*.

5. Review the summary. YaST displays the current settings for confirmation. Click *Create*. The root CA is created then appears in the overview.

Tip

In general, it is best not to allow user certificates to be issued by the root CA. It is better to create at least one sub-CA and create the user certificates from there. This has the advantage that the root CA can be kept isolated and secure, for example, on an isolated computer on secure premises. This makes it very difficult to attack the root CA.

18.2.2 Changing Password

If you need to change your password for your CA, proceed as follows:

1. Start YaST and open the CA module.
2. Select the required root CA and click *Enter CA*.
3. Enter the password if you entered a CA the first time. YaST displays the CA key information in the *Description* tab (see *Figure 18.2*).

4. Click *Advanced* and select *Change CA Password*. A dialog opens.
5. Enter the old and the new password.
6. Finish with *OK*

18.2.3 Creating or Revoking a Sub-CA

A sub-CA is created in the same way as a root CA.

Note

The validity period for a sub-CA must be fully within the validity period of the “parent” CA. A sub-CA is always created after the “parent” CA, therefore, the default value leads to an error message. To avoid this, enter a permissible value for the period of validity.

Do the following:

1. Start YaST and open the CA module.
2. Select the required root CA and click *Enter CA*.
3. Enter the password if you are entering a CA for the first time. YaST displays the CA key information in the tab *Description* (see *Figure 18.2*).

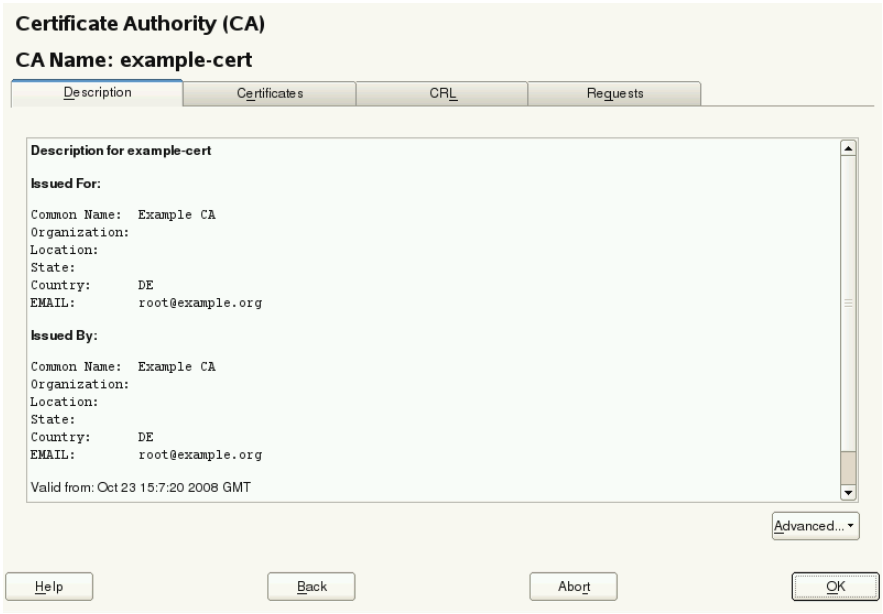


FIGURE 18.2: YAST CA MODULE—USING A CA

4. Click *Advanced* and select *Create SubCA*. This opens the same dialog as for creating a root CA.

5. Proceed as described in *Section 18.2.1, "Creating a Root CA"*.
   It is possible to use one password for all your CAs. Enable *Use CA Password as Certificate Password* to give your sub-CAs the same password as your root CA. This helps to reduce the amount of passwords for your CAs.

   Note: Check your Valid Period

   Take into account that the valid period must be lower than the valid period in the root CA.

6. Select the *Certificates* tab. Reset compromised or otherwise unwanted sub-CAs here, using *Revoke*. Revocation alone is not enough to deactivate a sub-CA. You must also publish revoked sub-CAs in a CRL. The creation of CRLs is described in *Section 18.2.6, "Creating Certificate Revocation Lists (CRLs)"*.

7. Finish with *OK*.

18.2.4 Creating or Revoking User Certificates

Creating client and server certificates is very similar to creating CAs in *Section 18.2.1, "Creating a Root CA"*. The same principles apply here. In certificates intended for e-mail signature, the e-mail address of the sender (the private key owner) should be contained in the certificate to enable the e-mail program to assign the correct certificate.

For certificate assignment during encryption, it is necessary for the e-mail address of the recipient (the public key owner) to be included in the certificate. In the case of server and client certificates, the host name of the server must be entered in the *Common Name* field. The default validity period for certificates is 365 days.

To create client and server certificates, do the following:

1. Start YaST and open the CA module.

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

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

4. Click *Certificates* (see *Figure 18.3*).

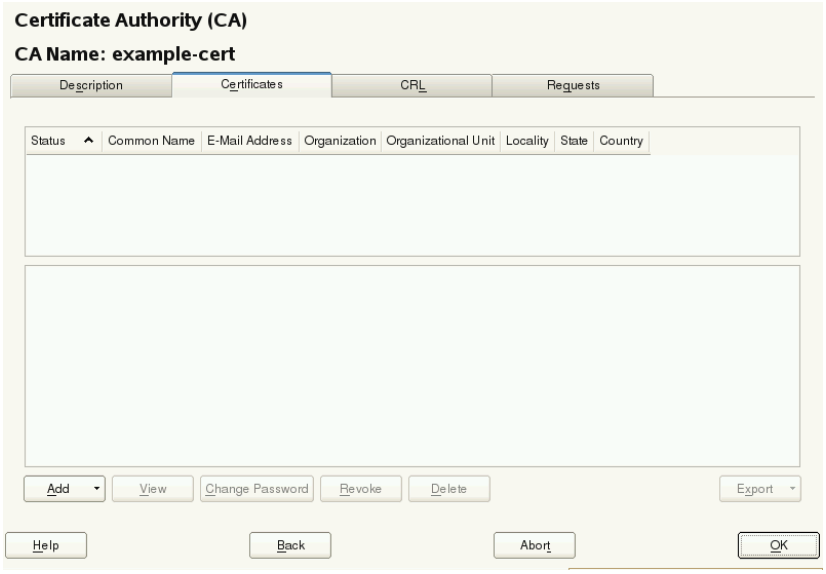


FIGURE 18.3: CERTIFICATES OF A CA

5. Click *Add* › *Add Server Certificate* and create a server certificate.

6. Click *Add* › *Add Client Certificate* and create a client certificate. Do not forget to enter an e-mail address.

7. Finish with *OK*

To revoke compromised or otherwise unwanted certificates, do the following:

1. Start YaST and open the CA module.

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

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

4. Click *Certificates* (see *Section 18.2.3, "Creating or Revoking a Sub-CA"*).

5. Select the certificate to revoke and click *Revoke*.

6. Choose a reason to revoke this certificate.

7. Finish with *OK*.

Note

Revocation alone is not enough to deactivate a certificate. Also publish revoked certificates in a CRL. *Section 18.2.6, "Creating Certificate Revocation Lists (CRLs)"* explains how to create CRLs. Revoked certificates can be completely removed after publication in a CRL with *Delete*.

18.2.5 Changing Default Values

The previous sections explained how to create sub-CAs, client certificates, and server certificates. Special settings are used in the extensions of the X.509 certificate. These settings have been given rational defaults for every certificate type and do not normally need to be changed. However, it may be that you have special requirements for these extensions. In this case, it may make sense to adjust the defaults. Otherwise, start from scratch every time you create a certificate.

1. Start YaST and open the CA module.
2. Enter the required root CA, as described in *Section 18.2.3, "Creating or Revoking a Sub-CA"*.
3. Click *Advanced* › *Edit Default*.
4. Choose type of certificate to change and proceed with *Next*.
5. The dialog for changing the defaults as shown in *Figure 18.4, "YaST CA Module—Extended Settings"* opens.

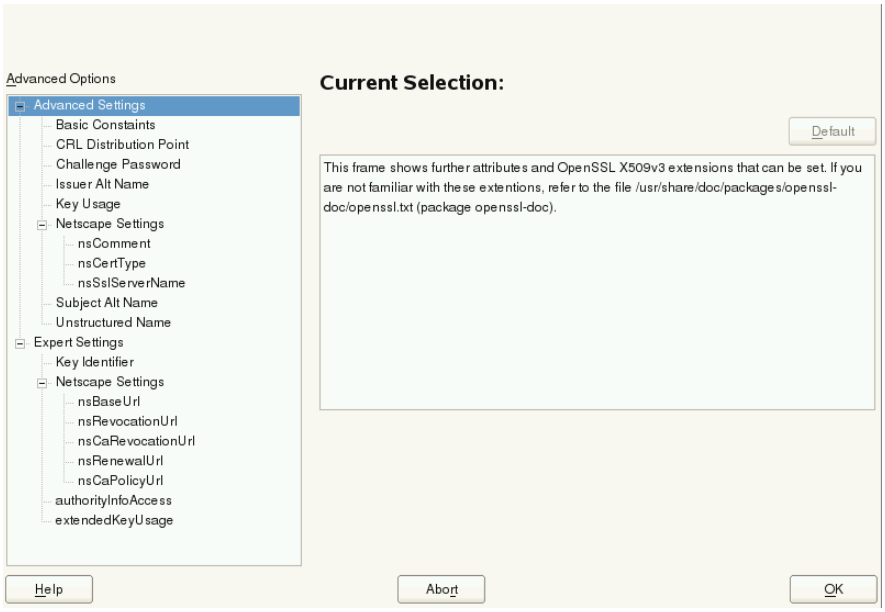


FIGURE 18.4: YAST CA MODULE—EXTENDED SETTINGS

6. Change the associated value on the right side and set or delete the critical setting with *critical*.

7. Click *Next* to see a short summary.

8. Finish your changes with *Save*.

Note

All changes to the defaults only affect objects created after this point. Already-existing CAs and certificates remain unchanged.

18.2.6 Creating Certificate Revocation Lists (CRLs)

If compromised or otherwise unwanted certificates need to be excluded from further use, they must first be revoked. The procedure for this is explained in *Section 18.2.3, "Creating or Revoking a Sub-CA"* (for sub-CAs) and *Section 18.2.4, "Creating or Revoking User Certificates"* (for user certificates). After this, a CRL must be created and published with this information.

The system maintains only one CRL for each CA. To create or update this CRL, do the following:

1. Start YaST and open the CA module.

2. Enter the required CA, as described in *Section 18.2.3, "Creating or Revoking a Sub-CA"*.
3. Click *CRL*. The dialog that opens displays a summary of the last CRL of this CA.
4. Create a new CRL with *Generate CRL* if you have revoked new sub-CAs or certificates since its creation.
5. Specify the period of validity for the new CRL (default: 30 days).
6. Click *OK* to create and display the CRL. Afterward, you must publish this CRL.

Note

Applications that evaluate CRLs reject every certificate if the CRL is not available or has expired. As a PKI provider, it is your duty always to create and publish a new CRL before the current CRL expires (period of validity). YaST does not provide a function for automating this procedure.

18.2.7 Exporting CA Objects to LDAP

The executing computer should be configured with the YaST LDAP client for LDAP export. This provides LDAP server information at runtime that can be used when completing dialog fields. Otherwise (although export may be possible), all LDAP data must be entered manually. You must always enter several passwords (see *Table 18.3, "Passwords during LDAP Export"*).

TABLE 18.3: PASSWORDS DURING LDAP EXPORT

Password	Meaning
LDAP Password	Authorizes the user to make entries in the LDAP tree.
Certificate Password	Authorizes the user to export the certificate.
New Certificate Password	The PKCS12 format is used during LDAP export. This format forces the assignment of a new password for the exported certificate.

Certificates, CAs, and CRLs can be exported to LDAP.

Exporting a CA to LDAP

To export a CA, enter the CA as described in *Section 18.2.3, "Creating or Revoking a Sub-CA"*. Select *Extended* › *Export to LDAP* in the subsequent dialog, which opens the dialog for entering LDAP data. If your system has been configured with the YaST LDAP client, the fields are already partly completed. Otherwise, enter all the data manually. Entries are made in LDAP in a separate tree with the attribute "caCertificate".

Exporting a Certificate to LDAP

Enter the CA containing the certificate to export then select *Certificates*. Select the required certificate from the certificate list in the upper part of the dialog and select *Export* › *Export to LDAP*. The LDAP data is entered here in the same way as for CAs. The certificate is saved with the corresponding user object in the LDAP tree with the attributes "userCertificate" (PEM format) and "userPKCS12" (PKCS12 format).

Exporting a CRL to LDAP

Enter the CA containing the CRL to export and select *CRL*. If desired, create a new CRL and click *Export*. The dialog that opens displays the export parameters. You can export the CRL for this CA either once or in periodical time intervals. Activate the export by selecting *Export to LDAP* and enter the respective LDAP data. To do this at regular intervals, select the *Repeated Recreation and Export* radio button and change the interval, if appropriate.

18.2.8 Exporting CA Objects as a File

If you have set up a repository on the computer for administering CAs, you can use this option to create the CA objects directly as a file at the correct location. Different output formats are available, such as PEM, DER, and PKCS12. In the case of PEM, it is also possible to choose whether a certificate should be exported with or without key and whether the key should be encrypted. In the case of PKCS12, it is also possible to export the certification path.

Export a file in the same way for certificates, CAs as with LDAP, described in *Section 18.2.7, "Exporting CA Objects to LDAP"*, except you should select *Export as File* instead of *Export to LDAP*. This then takes you to a dialog for selecting the required output format and entering the password and file name. The certificate is stored at the required location after clicking *OK*.

For CRLs click *Export*, select *Export to file*, choose the export format (PEM or DER) and enter the path. Proceed with *OK* to save it to the respective location.

Tip

You can select any storage location in the file system. This option can also be used to save CA objects on a transport medium, such as a flash disk. The `/media` directory generally holds any type of drive except the hard disk of your system.

18.2.9 Importing Common Server Certificates

If you have exported a server certificate with YaST to your media on an isolated CA management computer, you can import this certificate on a server as a *common server certificate*. Do this during installation or at a later point with YaST.

Note

You need one of the PKCS12 formats to import your certificate successfully.

The general server certificate is stored in `/etc/ssl/servercerts` and can be used there by any CA-supported service. When this certificate expires, it can easily be replaced using the same mechanisms. To get things functioning with the replaced certificate, restart the participating services.

Tip

If you select *Import* here, you can select the source in the file system. This option can also be used to import certificates from removable media, such as a flash disk.

To import a common server certificate, do the following:

1. Start YaST and open *Common Server Certificate* under *Security and Users*
2. View the data for the current certificate in the description field after YaST has been started.
3. Select *Import* and the certificate file.
4. Enter the password and click *Next*. The certificate is imported then displayed in the description field.
5. Close YaST with *Finish*.

IV Confining Privileges with AppArmor

19 Introducing AppArmor

Many security vulnerabilities result from bugs in *trusted* programs. A trusted program runs with privileges that attackers want to possess. The program fails to keep that trust if there is a bug in the program that allows the attacker to acquire said privilege.

AppArmor® is an application security solution designed specifically to apply privilege confinement to suspect programs. AppArmor allows the administrator to specify the domain of activities the program can perform by developing a security *profile*. A security profile is a listing of files that the program may access and the operations the program may perform. AppArmor secures applications by enforcing good application behavior without relying on attack signatures, so it can prevent attacks even if previously unknown vulnerabilities are being exploited.

19.1 AppArmor Components

AppArmor consists of:

- A library of AppArmor profiles for common Linux* applications, describing what files the program needs to access.
- A library of AppArmor profile foundation classes (profile building blocks) needed for common application activities, such as DNS lookup and user authentication.
- A tool suite for developing and enhancing AppArmor profiles, so that you can change the existing profiles to suit your needs and create new profiles for your own local and custom applications.
- Several specially modified applications that are AppArmor enabled to provide enhanced security in the form of unique subprocess confinement (including Apache).
- The AppArmor-related kernel code and associated control scripts to enforce AppArmor policies on your openSUSE® Leap system.

19.2 Background Information on AppArmor Profiling

For more information about the science and security of AppArmor, refer to the following papers:

SubDomain: Parsimonious Server Security **by Crispin Cowan, Steve Beattie, Greg Kroah-Hartman, Calton Pu, Perry Wagle, and Virgil Gligor**

Describes the initial design and implementation of AppArmor. Published in the proceedings of the USENIX LISA Conference, December 2000, New Orleans, LA. This paper is now out of date, describing syntax and features that are different from the current AppArmor product. This paper should be used only for background, and not for technical documentation.

Defcon Capture the Flag: Defending Vulnerable Code from Intense Attack **by Crispin Cowan, Seth Arnold, Steve Beattie, Chris Wright, and John Viega**

A good guide to strategic and tactical use of AppArmor to solve severe security problems in a very short period of time. Published in the Proceedings of the DARPA Information Survivability Conference and Expo (DISCEX III), April 2003, Washington, DC.

AppArmor for Geeks **by Seth Arnold**

This document tries to convey a better understanding of the technical details of AppArmor. It is available at http://en.opensuse.org/SDB:AppArmor_geeks.

20 Getting Started

Prepare a successful deployment of AppArmor on your system by carefully considering the following items:

1. Determine the applications to profile. Read more on this in *Section 20.3, "Choosing Applications to Profile"*.
2. Build the needed profiles as roughly outlined in *Section 20.4, "Building and Modifying Profiles"*. Check the results and adjust the profiles when necessary.
3. Update your profiles whenever your environment changes or you need to react to security events logged by the reporting tool of AppArmor. Refer to *Section 20.5, "Updating Your Profiles"*.

20.1 Installing AppArmor

AppArmor is installed and running on any installation of openSUSE® Leap by default, regardless of what patterns are installed. The packages listed below are needed for a fully-functional instance of AppArmor:

- `apparmor-docs`
- `apparmor-parser`
- `apparmor-profiles`
- `apparmor-utils`
- `audit`
- `libapparmor1`
- `perl-libapparmor`
- `yast2-apparmor`

Tip

If AppArmor is not installed on your system, install the pattern `apparmor` for a complete AppArmor installation. Either use the YaST Software Management module for installation, or use Zypper on the command line:

```
tux > sudo zypper in -t pattern apparmor
```

20.2 Enabling and Disabling AppArmor

AppArmor is configured to run by default on any fresh installation of openSUSE Leap. There are two ways of toggling the status of AppArmor:

Using YaST Services Manager

Disable or enable AppArmor by removing or adding its boot script to the sequence of scripts executed on system boot. Status changes are applied on reboot.

Using AppArmor Configuration Window

Toggle the status of AppArmor in a running system by switching it off or on using the YaST AppArmor Control Panel. Changes made here are applied instantaneously. The Control Panel triggers a stop or start event for AppArmor and removes or adds its boot script in the system's boot sequence.

To disable AppArmor permanently (by removing it from the sequence of scripts executed on system boot) proceed as follows:

1. Start YaST.
2. Select *System* › *Services Manager*.
3. Mark `apparmor` by clicking its row in the list of services, then click *Enable/Disable* in the lower part of the window. Check that *Enabled* changed to *Disabled* in the `apparmor` row.
4. Confirm with *OK*.

AppArmor will not be initialized on reboot, and stays inactive until you re-enable it. Re-enabling a service using the YaST *Services Manager* tool is similar to disabling it.

Toggle the status of AppArmor in a running system by using the AppArmor Configuration window. These changes take effect when you apply them and survive a reboot of the system. To toggle the status of AppArmor, proceed as follows:

1. Start YaST, select *AppArmor Configuration*, and click *Settings* in the main window.
2. Enable AppArmor by checking *Enable AppArmor* or disable AppArmor by deselecting it.
3. Click *Done* in the *AppArmor Configuration* window.

20.3 Choosing Applications to Profile

You only need to protect the programs that are exposed to attacks in your particular setup, so only use profiles for those applications you actually run. Use the following list to determine the most likely candidates:

Network Agents
Web Applications
Cron Jobs

To find out which processes are currently running with open network ports and might need a profile to confine them, run **`aa-unconfined`** as `root`.

EXAMPLE 20.1: OUTPUT OF **`aa-unconfined`**

```
19848 /usr/sbin/cupsd not confined
19887 /usr/sbin/sshd not confined
19947 /usr/lib/postfix/master not confined
1328 /usr/sbin/smbd confined by '/usr/sbin/smbd (enforce)'
```

Each of the processes in the above example labeled `not confined` might need a custom profile to confine it. Those labeled `confined by` are already protected by AppArmor.

Tip: For More Information

For more information about choosing the right applications to profile, refer to *Section 21.2, "Determining Programs to Immunize"*.

20.4 Building and Modifying Profiles

AppArmor on openSUSE Leap ships with a preconfigured set of profiles for the most important applications. In addition, you can use AppArmor to create your own profiles for any application you want.

There are two ways of managing profiles. One is to use the graphical front-end provided by the YaST AppArmor modules and the other is to use the command line tools provided by the AppArmor suite itself. The main difference is that YaST supports only basic functionality for AppArmor profiles, while the command line tools let you update/tune the profiles in a more fine-grained way.

For each application, perform the following steps to create a profile:

1. As `root`, let AppArmor create a rough outline of the application's profile by running **`aa-genprof`** *`PROGRAM_NAME`*.
   or
   Outline the basic profile by running *YaST › Security and Users › AppArmor Configuration › Manually Add Profile* and specifying the complete path to the application you want to profile.
   A new basic profile is outlined and put into learning mode, which means that it logs any activity of the program you are executing, but does not yet restrict it.

2. Run the full range of the application's actions to let AppArmor get a very specific picture of its activities.

3. Let AppArmor analyze the log files generated in *Step 2* by typing S in aa-genprof.
   AppArmor scans the logs it recorded during the application's run and asks you to set the access rights for each event that was logged. Either set them for each file or use globbing.

4. Depending on the complexity of your application, it might be necessary to repeat *Step 2* and *Step 3*. Confine the application, exercise it under the confined conditions, and process any new log events. To properly confine the full range of an application's capabilities, you might be required to repeat this procedure often.

5. When you finish **`aa-genprof`**, your profile is set to enforce mode. The profile is applied and AppArmor restricts the application according to it.
   If you started **`aa-genprof`** on an application that had an existing profile that was in complain mode, this profile remains in learning mode upon exit of this learning cycle. For more information about changing the mode of a profile, refer to *Section 25.7.3.2, "aa-complain—Entering Complain or Learning Mode"* and *Section 25.7.3.6, "aa-enforce—Entering Enforce Mode"*.

Test your profile settings by performing every task you need with the application you confined. Normally, the confined program runs smoothly and you do not notice AppArmor activities. However, if you notice certain misbehavior with your application, check the system logs and see if AppArmor is too tightly confining your application. Depending on the log mechanism used on your system, there are several places to look for AppArmor log entries:

`/var/log/audit/audit.log`
The command **`journalctl | grep -i apparmor`**

The command **dmesg -T**

To adjust the profile, analyze the log messages relating to this application again as described in *Section 25.7.3.9, "aa-logprof—Scanning the System Log"*. Determine the access rights or restrictions when prompted.

Tip: For More Information

For more information about profile building and modification, refer to *Chapter 22, Profile Components and Syntax*, *Chapter 24, Building and Managing Profiles with YaST*, and *Chapter 25, Building Profiles from the Command Line*.

20.5 Updating Your Profiles

Software and system configurations change over time. As a result, your profile setup for AppArmor might need some fine-tuning from time to time. AppArmor checks your system log for policy violations or other AppArmor events and lets you adjust your profile set accordingly. Any application behavior that is outside of any profile definition can be addressed by **aa-logprof**. For more information, see *Section 25.7.3.9, "aa-logprof—Scanning the System Log"*.

21 Immunizing Programs

Effective hardening of a computer system requires minimizing the number of programs that mediate privilege, then securing the programs as much as possible. With AppArmor, you only need to profile the programs that are exposed to attack in your environment, which drastically reduces the amount of work required to harden your computer. AppArmor profiles enforce policies to make sure that programs do what they are supposed to do, but nothing else.

AppArmor provides immunization technologies that protect applications from the inherent vulnerabilities they possess. After installing AppArmor, setting up AppArmor profiles, and rebooting the computer, your system becomes immunized because it begins to enforce the AppArmor security policies. Protecting programs with AppArmor is called *immunizing*.

Administrators need only concern themselves with the applications that are vulnerable to attacks, and generate profiles for these. Hardening a system thus comes down to building and maintaining the AppArmor profile set and monitoring any policy violations or exceptions logged by AppArmor's reporting facility.

Users should not notice AppArmor. It runs “behind the scenes” and does not require any user interaction. Performance is not noticeably affected by AppArmor. If some activity of the application is not covered by an AppArmor profile or if some activity of the application is prevented by AppArmor, the administrator needs to adjust the profile of this application.

AppArmor sets up a collection of default application profiles to protect standard Linux services. To protect other applications, use the AppArmor tools to create profiles for the applications that you want protected. This chapter introduces the philosophy of immunizing programs. Proceed to *Chapter 22, Profile Components and Syntax*, *Chapter 24, Building and Managing Profiles with YaST*, or *Chapter 25, Building Profiles from the Command Line* if you are ready to build and manage AppArmor profiles.

AppArmor provides streamlined access control for network services by specifying which files each program is allowed to read, write, and execute, and which type of network it is allowed to access. This ensures that each program does what it is supposed to do, and nothing else. AppArmor quarantines programs to protect the rest of the system from being damaged by a compromised process.

AppArmor is a host intrusion prevention or mandatory access control scheme. Previously, access control schemes were centered around users because they were built for large timeshare systems. Alternatively, modern network servers largely do not permit users to log in, but instead provide

a variety of network services for users (such as Web, mail, file, and print servers). AppArmor controls the access given to network services and other programs to prevent weaknesses from being exploited.

Tip: Background Information for AppArmor

To get a more in-depth overview of AppArmor and the overall concept behind it, refer to *Section 19.2, "Background Information on AppArmor Profiling"*.

21.1 Introducing the AppArmor Framework

This section provides a very basic understanding of what is happening "behind the scenes" (and under the hood of the YaST interface) when you run AppArmor.

An AppArmor profile is a plain text file containing path entries and access permissions. See *Section 22.1, "Breaking an AppArmor Profile into Its Parts"* for a detailed reference profile. The directives contained in this text file are then enforced by the AppArmor routines to quarantine the process or program.

The following tools interact in the building and enforcement of AppArmor profiles and policies:

`aa-status`

`aa-status` reports various aspects of the current state of the running AppArmor confinement.

`aa-unconfined`

`aa-unconfined` detects any application running on your system that listens for network connections and is not protected by an AppArmor profile. Refer to *Section 25.7.3.12, "aa-unconfined—Identifying Unprotected Processes"* for detailed information about this tool.

`aa-autodep`

`aa-autodep` creates a basic framework of a profile that needs to be fleshed out before it is put to use in production. The resulting profile is loaded and put into complain mode, reporting any behavior of the application that is not (yet) covered by AppArmor rules. Refer to *Section 25.7.3.1, "aa-autodep—Creating Approximate Profiles"* for detailed information about this tool.

`aa-genprof`

aa-genprof generates a basic profile and asks you to refine this profile by executing the application and generating log events that need to be taken care of by AppArmor policies. You are guided through a series of questions to deal with the log events that have been triggered during the application's execution. After the profile has been generated, it is loaded and put into enforce mode. Refer to *Section 25.7.3.8, "aa-genprof—Generating Profiles"* for detailed information about this tool.

aa-logprof

aa-logprof interactively scans and reviews the log entries generated by an application that is confined by an AppArmor profile in both complain and enforced modes. It assists you in generating new entries in the profile concerned. Refer to *Section 25.7.3.9, "aa-logprof —Scanning the System Log"* for detailed information about this tool.

aa-easyprof

aa-easyprof provides an easy-to-use interface for AppArmor profile generation. **aa-easyprof** supports the use of templates and policy groups to quickly profile an application. Note that while this tool can help with policy generation, its utility is dependent on the quality of the templates, policy groups and abstractions used. **aa-easyprof** may create a profile that is less restricted than creating the profile with **aa-genprof** and **aa-logprof**.

aa-complain

aa-complain toggles the mode of an AppArmor profile from enforce to complain. Violations to rules set in a profile are logged, but the profile is not enforced. Refer to *Section 25.7.3.2, "aa-complain—Entering Complain or Learning Mode"* for detailed information about this tool.

aa-enforce

aa-enforce toggles the mode of an AppArmor profile from complain to enforce. Violations to rules set in a profile are logged and not permitted—the profile is enforced. Refer to *Section 25.7.3.6, "aa-enforce—Entering Enforce Mode"* for detailed information about this tool.

aa-disable

aa-disable disables the enforcement mode for one or more AppArmor profiles. This command will unload the profile from the kernel and prevent it from being loaded on AppArmor start-up. The **aa-enforce** and **aa-complain** utilities may be used to change this behavior.

aa-exec

aa-exec launches a program confined by the specified AppArmor profile and/or namespace. If both a profile and namespace are specified, the command will be confined by the profile in the new policy namespace. If only a namespace is specified, the profile name of the current confinement will be used. If neither a profile or namespace is specified, the command will be run using standard profile attachment—as if run without **aa-exec**.

aa-notify

aa-notify is a handy utility that displays AppArmor notifications in your desktop environment. You can also configure it to display a summary of notifications for the specified number of recent days. For more information, see *Section 25.7.3.13, "aa-notify"*.

21.2 Determining Programs to Immunize

Now that you have familiarized yourself with AppArmor, start selecting the applications for which to build profiles. Programs that need profiling are those that mediate privilege. The following programs have access to resources that the person using the program does not have, so they grant the privilege to the user when used:

`cron` Jobs

Programs that are run periodically by `cron`. Such programs read input from a variety of sources and can run with special privileges, sometimes with as much as `root` privilege. For example, `cron` can run **/usr/sbin/logrotate** daily to rotate, compress, or even mail system logs. For instructions for finding these types of programs, refer to *Section 21.3, "Immunizing `cron` Jobs"*.

Web Applications

Programs that can be invoked through a Web browser, including CGI Perl scripts, PHP pages, and more complex Web applications. For instructions for finding these types of programs, refer to *Section 21.4.1, "Immunizing Web Applications"*.

Network Agents

Programs (servers and clients) that have open network ports. User clients, such as mail clients and Web browsers mediate privilege. These programs run with the privilege to write to the user's home directory and they process input from potentially hostile remote sources, such as hostile Web sites and e-mailed malicious code. For instructions for finding these types of programs, refer to *Section 21.4.2, "Immunizing Network Agents"*.

Conversely, unprivileged programs do not need to be profiled. For example, a shell script might invoke the **cp** program to copy a file. Because **cp** does not by default have its own profile or subprofile, it inherits the profile of the parent shell script. Thus **cp** can copy any files that the parent shell script's profile can read and write.

21.3 Immunizing cron Jobs

To find programs that are run by `cron`, inspect your local `cron` configuration. Unfortunately, `cron` configuration is rather complex, so there are numerous files to inspect. Periodic `cron` jobs are run from these files:

```
/etc/crontab
/etc/cron.d/*
/etc/cron.daily/*
/etc/cron.hourly/*
/etc/cron.monthly/*
/etc/cron.weekly/*
```

The **crontab** command lists/edits the current user's crontab. To manipulate `root`'s `cron` jobs, first become `root`, and then edit the tasks with **crontab -e** or list them with **crontab -l**.

21.4 Immunizing Network Applications

An automated method for finding network server daemons that should be profiled is to use the **aa-unconfined** tool.

The **aa-unconfined** tool uses the command **netstat -nlp** to inspect open ports from inside your computer, detect the programs associated with those ports, and inspect the set of AppArmor profiles that you have loaded. **aa-unconfined** then reports these programs along with the AppArmor profile associated with each program, or reports “none” (if the program is not confined).

Note

If you create a new profile, you must restart the program that has been profiled to have it be effectively confined by AppArmor.

Below is a sample **aa-unconfined** output:

```
3702 (1) /usr/sbin/sshd (2) confined
   by '/usr/sbin/sshd (3) (enforce)'
4040 /usr/sbin/smbd confined by '/usr/sbin/smbd (enforce)'
4373 /usr/lib/postfix/master confined by '/usr/lib/postfix/master (enforce)'
4505 /usr/sbin/httpd2-prefork confined by '/usr/sbin/httpd2-prefork (enforce)'
646 /usr/lib/wicked/bin/wickedd-dhcp4 not confined
647 /usr/lib/wicked/bin/wickedd-dhcp6 not confined
5592 /usr/bin/ssh not confined
7146 /usr/sbin/cupsd confined by '/usr/sbin/cupsd (complain)'
```

1. The first portion is a number. This number is the process ID number (PID) of the listening program.
2. The second portion is a string that represents the absolute path of the listening program
3. The final portion indicates the profile confining the program, if any.

Note

aa-unconfined requires `root` privileges and should not be run from a shell that is confined by an AppArmor profile.

aa-unconfined does not distinguish between one network interface and another, so it reports all unconfined processes, even those that might be listening to an internal LAN interface.

Finding user network client applications is dependent on your user preferences. The **aa-unconfined** tool detects and reports network ports opened by client applications, but only those client applications that are running at the time the **aa-unconfined** analysis is performed. This is a problem because network services tend to be running all the time, while network client applications tend only to be running when the user is interested in them.

Applying AppArmor profiles to user network client applications is also dependent on user preferences. Therefore, we leave the profiling of user network client applications as an exercise for the user.

To aggressively confine desktop applications, the **aa-unconfined** command supports a `--paranoid` option, which reports all processes running and the corresponding AppArmor profiles that might or might not be associated with each process. The user can then decide whether each of these programs needs an AppArmor profile.

If you have new or modified profiles, you can submit them to the `apparmor@lists.ubuntu.com` mailing list along with a use case for the application behavior that you exercised. The AppArmor team reviews and may submit the work into openSUSE Leap. We cannot guarantee that every profile will be included, but we make a sincere effort to include as much as possible.

21.4.1 Immunizing Web Applications

To find Web applications, investigate your Web server configuration. The Apache Web server is highly configurable and Web applications can be stored in many directories, depending on your local configuration. openSUSE Leap, by default, stores Web applications in `/srv/www/cgi-bin/`. To the maximum extent possible, each Web application should have an AppArmor profile.

Once you find these programs, you can use the **`aa-genprof`** and **`aa-logprof`** tools to create or update their AppArmor profiles.

Because CGI programs are executed by the Apache Web server, the profile for Apache itself, `usr.sbin.httpd2-prefork` for Apache2 on openSUSE Leap, must be modified to add execute permissions to each of these programs. For example, adding the line `/srv/www/cgi-bin/my_hit_counter.pl rPx` grants Apache permission to execute the Perl script `my_hit_counter.pl` and requires that there be a dedicated profile for `my_hit_counter.pl`. If `my_hit_counter.pl` does not have a dedicated profile associated with it, the rule should say `/srv/www/cgi-bin/my_hit_counter.pl rix` to cause `my_hit_counter.pl` to inherit the `usr.sbin.httpd2-prefork` profile.

Some users might find it inconvenient to specify execute permission for every CGI script that Apache might invoke. Instead, the administrator can grant controlled access to collections of CGI scripts. For example, adding the line `/srv/www/cgi-bin/*.{pl,py,pyc} rix` allows Apache to execute all files in `/srv/www/cgi-bin/` ending in `.pl` (Perl scripts) and `.py` or `.pyc` (Python scripts). As above, the `ix` part of the rule causes Python scripts to inherit the Apache profile, which is appropriate if you do not want to write individual profiles for each CGI script.

Note

If you want the subprocess confinement module (`apache2-mod-apparmor`) functionality when Web applications handle Apache modules (`mod_perl` and `mod_php`), use the ChangeHat features when you add a profile in YaST or at the command line. To take advantage of the subprocess confinement, refer to *Section 26.2, “Managing ChangeHat-Aware Applications”*.

Profiling Web applications that use `mod_perl` and `mod_php` requires slightly different handling. In this case, the "program" is a script interpreted directly by the module within the Apache process, so no exec happens. Instead, the AppArmor version of Apache calls **`change_hat()`** using a subprofile (a "hat") corresponding to the name of the URI requested.

Note

The name presented for the script to execute might not be the URI, depending on how Apache has been configured for where to look for module scripts. If you have configured your Apache to place scripts in a different place, the different names appear in the log file when AppArmor complains about access violations. See *Chapter 28, Managing Profiled Applications*.

For `mod_perl` and `mod_php` scripts, this is the name of the Perl script or the PHP page requested. For example, adding this subprofile allows the `localtime.php` page to execute and access to the local system time and locale files:

```
/usr/bin/httpd2-prefork {
  # ...
  ^/cgi-bin/localtime.php {
    /etc/localtime                  r,
    /srv/www/cgi-bin/localtime.php  r,
    /usr/lib/locale/**              r,
  }
}
```

If no subprofile has been defined, the AppArmor version of Apache applies the `DEFAULT_URI` hat. This subprofile is sufficient to display a Web page. The `DEFAULT_URI` hat that AppArmor provides by default is the following:

```
^DEFAULT_URI {
    /usr/sbin/suexec2                 mixr,
    /var/log/apache2/**               rwl,
    @{HOME}/public_html               r,
    @{HOME}/public_html/**            r,
    /srv/www/htdocs                   r,
    /srv/www/htdocs/**                r,
    /srv/www/icons/*.{gif,jpg,png}    r,
    /srv/www/vhosts                   r,
    /srv/www/vhosts/**                r,
    /usr/share/apache2/**             r,
    /var/lib/php/sess_*               rwl
```

```
}
```

To use a single AppArmor profile for all Web pages and CGI scripts served by Apache, a good approach is to edit the `DEFAULT_URI` subprofile. For more information on confining Web applications with Apache, see *Chapter 26, Profiling Your Web Applications Using ChangeHat*.

21.4.2 Immunizing Network Agents

To find network server daemons and network clients (such as **`fetchmail`** or Firefox) that need to be profiled, you should inspect the open ports on your machine. Also consider the programs that are answering on those ports, and provide profiles for as many of those programs as possible. If you provide profiles for all programs with open network ports, an attacker cannot get to the file system on your machine without passing through an AppArmor profile policy.

Scan your server for open network ports manually from outside the machine using a scanner (such as nmap), or from inside the machine using the **`netstat --inet -n -p`** command as `root`. Then, inspect the machine to determine which programs are answering on the discovered open ports.

Tip

Refer to the man page of the **`netstat`** command for a detailed reference of all possible options.

22 Profile Components and Syntax

Building AppArmor profiles to confine an application is very straightforward and intuitive. AppArmor ships with several tools that assist in profile creation. It does not require you to do any programming or script handling. The only task that is required of the administrator is to determine a policy of strictest access and execute permissions for each application that needs to be hardened.

Updates or modifications to the application profiles are only required if the software configuration or the desired range of activities changes. AppArmor offers intuitive tools to handle profile updates and modifications.

You are ready to build AppArmor profiles after you select the programs to profile. To do so, it is important to understand the components and syntax of profiles. AppArmor profiles contain several building blocks that help build simple and reusable profile code:

Include Files

Include statements are used to pull in parts of other AppArmor profiles to simplify the structure of new profiles.

Abstractions

Abstractions are include statements grouped by common application tasks.

Program Chunks

Program chunks are include statements that contain chunks of profiles that are specific to program suites.

Capability Entries

Capability entries are profile entries for any of the POSIX.1e http://en.wikipedia.org/wiki/POSIX#POSIX.1 Linux capabilities allowing a fine-grained control over what a confined process is allowed to do through system calls that require privileges.

Network Access Control Entries

Network Access Control Entries mediate network access based on the address type and family.

Local Variable Definitions

Local variables define shortcuts for paths.

File Access Control Entries

File Access Control Entries specify the set of files an application can access.

rlimit Entries

rlimit entries set and control an application's resource limits.

For help determining the programs to profile, refer to *Section 21.2, "Determining Programs to Immunize"*. To start building AppArmor profiles with YaST, proceed to *Chapter 24, Building and Managing Profiles with YaST*. To build profiles using the AppArmor command line interface, proceed to *Chapter 25, Building Profiles from the Command Line*.

For more details about creating AppArmor profiles, see **`man 5 apparmor`**.

22.1 Breaking an AppArmor Profile into Its Parts

The easiest way of explaining what a profile consists of and how to create one is to show the details of a sample profile, in this case for a hypothetical application called **`/usr/bin/foo`**:

```
#include <tunables/global>❶

# a comment naming the application to confine
/usr/bin/foo❷ {❸
   #include <abstractions/base>❹

   capability setgid❺,
   network inet tcp❻,

   link /etc/sysconfig/foo -> /etc/foo.conf,❼
   /bin/mount            ux,
   /dev/{,u}❽random      r,
   /etc/ld.so.cache      r,
   /etc/foo/*            r,
   /lib/ld-*.so*         mr,
   /lib/lib*.so*         mr,
   /proc/[0-9]**         r,
   /usr/lib/**           mr,
   /tmp/                 r,❾
   /tmp/foo.pid          wr,
   /tmp/foo.*            lrw,
   /@{HOME}❿/.foo_file   rw,
   /@{HOME}/.foo_lock    kw,
   owner⓫ /shared/foo/** rw,
   /usr/bin/foobar       Cx,⓬
   /bin/**               Px -> bin_generic,⓭

   # a comment about foo's local (children) profile for /usr/bin/foobar.
```

```
  profile /usr/bin/foobar 14 {
     /bin/bash          rmix,
     /bin/cat           rmix,
     /bin/more          rmix,
     /var/log/foobar*   rwl,
     /etc/foobar        r,
   }

  # foo's hat, bar.
   ^bar 15 {
    /lib/ld-*.so*         mr,
    /usr/bin/bar          px,
    /var/spool/*          rwl,
   }
}
```

1. This loads a file containing variable definitions.
2. The normalized path to the program that is confined.
3. The curly braces (`{}`) serve as a container for include statements, subprofiles, path entries, capability entries, and network entries.
4. This directive pulls in components of AppArmor profiles to simplify profiles.
5. Capability entry statements enable each of the 29 POSIX.1e draft capabilities.
6. A directive determining the kind of network access allowed to the application. For details, refer to *Section 22.5, "Network Access Control"*.
7. A link pair rule specifying the source and the target of a link. See *Section 22.7.6, "Link Pair"* for more information.
8. The curly braces (`{}`) here allow for each of the listed possibilities, one of which is the empty string.
9. A path entry specifying what areas of the file system the program can access. The first part of a path entry specifies the absolute path of a file (including regular expression globbing) and the second part indicates permissible access modes (for example `r` for read, `w` for write, and `x` for execute). A whitespace of any kind (spaces or tabs) can precede the path name, but must separate the path name and the mode specifier. Spaces between the access mode and the trailing comma are optional. Find a comprehensive overview of the available access modes in *Section 22.7, "File Permission Access Modes"*.
10. This variable expands to a value that can be changed without changing the entire profile.
11. An owner conditional rule, granting read and write permission on files owned by the user. Refer to *Section 22.7.8, "Owner Conditional Rules"* for more information.

12 This entry defines a transition to the local profile `/usr/bin/foobar`. Find a comprehensive overview of the available execute modes in *Section 22.12, "Execute Modes"*.

13 A named profile transition to the profile bin_generic located in the global scope. See *Section 22.12.7, "Named Profile Transitions"* for details.

14 The local profile `/usr/bin/foobar` is defined in this section.

15 This section references a "hat" subprofile of the application. For more details on AppArmor's ChangeHat feature, refer to *Chapter 26, Profiling Your Web Applications Using ChangeHat*.

When a profile is created for a program, the program can access only the files, modes, and POSIX capabilities specified in the profile. These restrictions are in addition to the native Linux access controls.

Example: To gain the capability `CAP_CHOWN`, the program must have both access to `CAP_CHOWN` under conventional Linux access controls (typically, be a `root`-owned process) and have the capability `chown` in its profile. Similarly, to be able to write to the file `/foo/bar` the program must have both the correct user ID and mode bits set in the files attributes and have `/foo/bar w` in its profile.

Attempts to violate AppArmor rules are recorded in `/var/log/audit/audit.log` if the `audit` package is installed, or in `/var/log/messages`, or only in `journalctl` if no traditional syslog is installed. Often AppArmor rules prevent an attack from working because necessary files are not accessible and, in all cases, AppArmor confinement restricts the damage that the attacker can do to the set of files permitted by AppArmor.

22.2 Profile Types

AppArmor knows four different types of profiles: standard profiles, unattached profiles, local profiles and hats. Standard and unattached profiles are stand-alone profiles, each stored in a file under `/etc/apparmor.d/`. Local profiles and hats are children profiles embedded inside of a parent profile used to provide tighter or alternate confinement for a subtask of an application.

22.2.1 Standard Profiles

The default AppArmor profile is attached to a program by its name, so a profile name must match the path to the application it is to confine.

```
/usr/bin/foo {
```

```
...
}
```

This profile will be automatically used whenever an unconfined process executes `/usr/bin/foo`.

22.2.2 Unattached Profiles

Unattached profiles do not reside in the file system namespace and therefore are not automatically attached to an application. The name of an unattached profile is preceded by the keyword `profile`. You can freely choose a profile name, except for the following limitations: the name must not begin with a `:` or `.` character. If it contains a whitespace, it must be quoted. If the name begins with a `/`, the profile is considered to be a standard profile, so the following two profiles are identical:

```
profile /usr/bin/foo {
...
}
/usr/bin/foo {
...
}
```

Unattached profiles are never used automatically, nor can they be transitioned to through a `Px` rule. They need to be attached to a program by either using a named profile transition (see *Section 22.12.7, "Named Profile Transitions"*) or with the `change_profile` rule (see *Section 22.2.5, "Change rules"*).

Unattached profiles are useful for specialized profiles for system utilities that generally should not be confined by a system-wide profile (for example, `/bin/bash`). They can also be used to set up roles or to confine a user.

22.2.3 Local Profiles

Local profiles provide a convenient way to provide specialized confinement for utility programs launched by a confined application. They are specified like standard profiles, except that they are embedded in a parent profile and begin with the `profile` keyword:

```
/parent/profile {
   ...
```

```
   profile /local/profile {
      ...
   }
}
```

To transition to a local profile, either use a `cx` rule (see *Section 22.12.2, "Discrete Local Profile Execute Mode (Cx)"*) or a named profile transition (see *Section 22.12.7, "Named Profile Transitions"*).

22.2.4 Hats

AppArmor "hats" are a local profiles with some additional restrictions and an implicit rule allowing for `change_hat` to be used to transition to them. Refer to *Chapter 26, Profiling Your Web Applications Using ChangeHat* for a detailed description.

22.2.5 Change rules

AppArmor provides `change_hat` and `change_profile` rules that control domain transitioning. `change_hat` are specified by defining hats in a profile, while `change_profile` rules refer to another profile and start with the keyword `change_profile`:

```
change_profile -> /usr/bin/foobar,
```

Both `change_hat` and `change_profile` provide for an application directed profile transition, without having to launch a separate application. `change_profile` provides a generic one way transition between any of the loaded profiles. `change_hat` provides for a returnable parent child transition where an application can switch from the parent profile to the hat profile and if it provides the correct secret key return to the parent profile at a later time.

`change_profile` is best used in situations where an application goes through a trusted setup phase and then can lower its privilege level. Any resources mapped or opened during the start-up phase may still be accessible after the profile change, but the new profile will restrict the opening of new resources, and will even limit some resources opened before the switch. Specifically, memory resources will still be available while capability and file resources (as long as they are not memory mapped) can be limited.

`change_hat` is best used in situations where an application runs a virtual machine or an interpreter that does not provide direct access to the applications resources (for example Apache's `mod_php`). Since `change_hat` stores the return secret key in the application's memory the phase of reduced privilege should not have direct access to memory. It is also important that

file access is properly separated, since the hat can restrict accesses to a file handle but does not close it. If an application does buffering and provides access to the open files with buffering, the accesses to these files might not be seen by the kernel and hence not restricted by the new profile.

Warning: Safety of Domain Transitions

The `change_hat` and `change_profile` domain transitions are less secure than a domain transition done through an exec because they do not affect a process's memory mappings, nor do they close resources that have already been opened.

22.3 Include Statements

Include statements are directives that pull in components of other AppArmor profiles to simplify profiles. Include files retrieve access permissions for programs. By using an include, you can give the program access to directory paths or files that are also required by other programs. Using includes can reduce the size of a profile.

Include statements normally begin with a hash (`#`) sign. This is confusing because the same hash sign is used for comments inside profile files. Because of this, `#include` is treated as an include only if there is no preceding # (`##include` is a comment) and there is no whitespace between `#` and `include` (`# include` is a comment).

You can also use `include` without the leading `#`.

```
include "/etc/apparmor.d/abstractions/foo"
```

is the same as using

```
#include "/etc/apparmor.d/abstractions/foo"
```

Note: No Trailing ','

Note that because includes follow the C pre-processor syntax, they do not have a trailing ',' like most AppArmor rules.

By slight changes in syntax, you can modify the behavior of `include`. If you use `""` around the including path, you instruct the parser to do an absolute or relative path lookup.

```
include "/etc/apparmor.d/abstractions/foo"   # absolute path
```

```
include "abstractions/foo"   # relative path to the directory of current file
```

Note that when using relative path includes, when the file is included, it is considered the new current file for its includes. For example, suppose you are in the `/etc/apparmor.d/bar` file, then

```
include "abstractions/foo"
```

includes the file `/etc/apparmor.d/abstractions/foo`. If then there is

```
include "example"
```

inside the `/etc/apparmor.d/abstractions/foo` file, it includes `/etc/apparmor.d/abstractions/example`.

The use of `<>` specifies to try the include path (specified by `-I`, defaults to the `/etc/apparmor.d` directory) in an ordered way. So assuming the include path is

```
-I /etc/apparmor.d/ -I /usr/share/apparmor/
```

then the include statement

```
include <abstractions/foo>
```

will try `/etc/apparmor.d/abstractions/foo`, and if that file does not exist, the next try is `/usr/share/apparmor/abstractions/foo`.

Tip

The default include path can be overridden manually by passing `-I` to the **`apparmor_parser`**, or by setting the include paths in `/etc/apparmor/parser.conf`:

```
Include /usr/share/apparmor/
Include /etc/apparmor.d/
```

Multiple entries are allowed, and they are taken in the same order as when they are when using `-I` or `--Include` from the **`apparmor_parser`** command line.

If an include ends with '/', this is considered a directory include, and all files within the directory are included.

To assist you in profiling your applications, AppArmor provides three classes of includes: abstractions, program chunks and tunables.

22.3.1 Abstractions

Abstractions are includes that are grouped by common application tasks. These tasks include access to authentication mechanisms, access to name service routines, common graphics requirements, and system accounting. Files listed in these abstractions are specific to the named task. Programs that require one of these files usually also require other files listed in the abstraction file (depending on the local configuration and the specific requirements of the program). Find abstractions in `/etc/apparmor.d/abstractions`.

22.3.2 Program Chunks

The program-chunks directory (`/etc/apparmor.d/program-chunks`) contains some chunks of profiles that are specific to program suites and not generally useful outside of the suite, thus are never suggested for use in profiles by the profile wizards (**`aa-logprof`** and **`aa-genprof`**). Currently, program chunks are only available for the postfix program suite.

22.3.3 Tunables

The tunables directory (`/etc/apparmor.d/tunables`) contains global variable definitions. When used in a profile, these variables expand to a value that can be changed without changing the entire profile. Add all the tunables definitions that should be available to every profile to `/etc/apparmor.d/tunables/global`.

22.4 Capability Entries (POSIX.1e)

Capability rules are simply the word `capability` followed by the name of the POSIX.1e capability as defined in the `capabilities(7)` man page. You can list multiple capabilities in a single rule, or grant all implemented capabilities with the bare keyword `capability`.

```
capability dac_override sys_admin,   # multiple capabilities
capability,                          # grant all capabilities
```

22.5 Network Access Control

AppArmor allows mediation of network access based on the address type and family. The following illustrates the network access rule syntax:

```
network [[<domain>❶][<type❷>][<protocol❸>]]
```

❶ Supported domains: `inet`, `ax25`, `ipx`, `appletalk`, `netrom`, `bridge`, `x25`, `inet6`, `rose`, `netbeui`, `security`, `key`, `packet`, `ash`, `econet`, `atmsvc`, `sna`, `pppox`, `wanpipe`, `bluetooth`, `unix`, `atmpvc`,`netlink`, `llc`, `can`, `tipc`, `iucv`, `rxrpc`, `isdn`, `phonet`, `ieee802154`, `caif`, `alg`, `nfc`, `vsock`

❷ Supported types: `stream`, `dgram`, `seqpacket`, `rdm`, `raw`, `packet`

❸ Supported protocols: `tcp`, `udp`, `icmp`

The AppArmor tools support only family and type specification. The AppArmor module emits only `network` *`DOMAIN TYPE`* in "ACCESS DENIED" messages. And only these are output by the profile generation tools, both YaST and command line.

The following examples illustrate possible network-related rules to be used in AppArmor profiles. Note that the syntax of the last two are not currently supported by the AppArmor tools.

```
network❶,
network inet❷,
network inet6❸,
network inet stream❹,
network inet tcp❺,
network tcp❻,
```

❶ Allow all networking. No restrictions applied with regard to domain, type, or protocol.

❷ Allow general use of IPv4 networking.

❸ Allow general use of IPv6 networking.

❹ Allow the use of IPv4 TCP networking.

❺ Allow the use of IPv4 TCP networking, paraphrasing the rule above.

❻ Allow the use of both IPv4 and IPv6 TCP networking.

22.6 Profile Names, Flags, Paths, and Globbing

A profile is usually attached to a program by specifying a full path to the program's executable. For example in the case of a standard profile (see *Section 22.2.1, "Standard Profiles"*), the profile is defined by

```
/usr/bin/foo { ... }
```

The following sections describe several useful techniques that can be applied when naming a profile or putting a profile in the context of other existing ones, or specifying file paths.

AppArmor explicitly distinguishes directory path names from file path names. Use a trailing `/` for any directory path that needs to be explicitly distinguished:

`/some/random/example/* r`
: Allow read access to files in the `/some/random/example` directory.

`/some/random/example/ r`
: Allow read access to the directory only.

`/some/**/ r`
: Give read access to any directories below `/some` (but not /some/ itself).

`/some/random/example/** r`
: Give read access to files and directories under `/some/random/example` (but not /some/random/example/ itself).

`/some/random/example/**[^/] r`
: Give read access to files under `/some/random/example`. Explicitly exclude directories (`[^/]`).

Globbing (or regular expression matching) is when you modify the directory path using wild cards to include a group of files or subdirectories. File resources can be specified with a globbing syntax similar to that used by popular shells, such as csh, Bash, and zsh.

`*`	Substitutes for any number of any characters, except `/`. Example: An arbitrary number of file path elements.

`**`	Substitutes for any number of characters, including `/`. Example: An arbitrary number of path elements, including entire directories.
`?`	Substitutes for any single character, except `/`.
`[abc]`	Substitutes for the single character `a`, `b`, or `c`. Example: a rule that matches `/home[01]/*/.plan` allows a program to access `.plan` files for users in both `/home0` and `/home1`.
`[a-c]`	Substitutes for the single character `a`, `b`, or `c`.
`{ab,cd}`	Expands to one rule to match `ab` and one rule to match `cd`. Example: a rule that matches `/{usr,www}/pages/**` grants access to Web pages in both `/usr/pages` and `/www/pages`.
`[^a]`	Substitutes for any character except `a`.

22.6.1 Profile Flags

Profile flags control the behavior of the related profile. You can add profile flags to the profile definition by editing it manually, see the following syntax:

```
/path/to/profiled/binary flags=(list_of_flags) {
  [...]
}
```

You can use multiple flags separated by a comma ',' or space ' '. There are three basic types of profile flags: mode, relative, and attach flags.

Mode flag is `complain` (illegal accesses are allowed and logged). If it is omitted, the profile is in `enforce` mode (enforces the policy).

Tip

A more flexible way of setting the whole profile into complain mode is to create a symbolic link from the profile file inside the `/etc/apparmor.d/force-complain/` directory.

```
ln -s /etc/apparmor.d/bin.ping /etc/apparmor.d/force-complain/bin.ping
```

Relative flags are `chroot_relative` (states that the profile is relative to the chroot instead of namespace) or `namespace_relative` (the default, with the path being relative to outside the chroot). They are mutually exclusive.

Attach flags consist of two pairs of mutually exclusive flags: `attach_disconnected` or `no_attach_disconnected` (determine if path names resolved to be outside of the namespace are attached to the root, which means they have the '/' character at the beginning), and `chroot_attach` or `chroot_no_attach` (control path name generation when in a chroot environment while a file is accessed that is external to the chroot but within the namespace).

22.6.2 Using Variables in Profiles

AppArmor allows to use variables holding paths in profiles. Use global variables to make your profiles portable and local variables to create shortcuts for paths.

A typical example of when global variables come in handy are network scenarios in which user home directories are mounted in different locations. Instead of rewriting paths to home directories in all affected profiles, you only need to change the value of a variable. Global variables are defined under `/etc/apparmor.d/tunables` and need to be made available via an include statement. Find the variable definitions for this use case (`@{HOME}` and `@{HOMEDIRS}`) in the `/etc/apparmor.d/tunables/home` file.

Local variables are defined at the head of a profile. This is useful to provide the base of for a chrooted path, for example:

```
@{CHROOT_BASE}=/tmp/foo
/sbin/rsyslogd {
...
# chrooted applications
@{CHROOT_BASE}/var/lib/*/dev/log w,
@{CHROOT_BASE}/var/log/** w,
...
}
```

In the following example, while @{HOMEDIRS} lists where all the user home directories are stored, @{HOME} is a space-separated list of home directories. Later on, @{HOMEDIRS} is expanded by two new specific places where user home directories are stored.

```
@{HOMEDIRS}=/home/
@{HOME}=@{HOMEDIRS}/*/ /root/
[...]
@{HOMEDIRS}+=/srv/nfs/home/ /mnt/home/
```

Note

With the current AppArmor tools, variables can only be used when manually editing and maintaining a profile.

22.6.3 Pattern Matching

Profile names can contain globbing expressions allowing the profile to match against multiple binaries.

The following example is valid for systems where the **`foo`** binary resides either in `/usr/bin` or `/bin`.

```
/{usr/,}bin/foo { ... }
```

In the following example, when matching against the executable `/bin/foo`, the `/bin/foo` profile is an exact match so it is chosen. For the executable `/bin/fat`, the profile `/bin/foo` does not match, and because the `/bin/f*` profile is more specific (less general) than `/bin/**`, the `/bin/f*` profile is chosen.

```
/bin/foo { ... }

/bin/f*  { ... }

/bin/**  { ... }
```

For more information on profile name globbing examples, see the man page of AppArmor, **`man 5 apparmor.d,`**, section `Globbing`.

22.6.4 Namespaces

Namespaces are used to provide different profiles sets. Say one for the system, another for a chroot environment or container. Namespaces are hierarchical—a namespace can see its children but a child cannot see its parent. Namespace names start with a colon `:` followed by an alphanumeric string, a trailing colon `:` and an optional double slash `//`, such as

```
:childNameSpace://
```

Profiles loaded to a child namespace will be prefixed with their namespace name (viewed from a parent's perspective):

```
:childNameSpace://apache
```

Namespaces can be entered via the `change_profile` API, or named profile transitions:

```
/path/to/executable px -> :childNameSpace://apache
```

22.6.5 Profile Naming and Attachment Specification

Profiles can have a name, and an attachment specification. This allows for profiles with a logical name that can be more meaningful to users/administrators than a profile name that contains pattern matching (see *Section 22.6.3, "Pattern Matching"*). For example, the default profile

```
/** { ... }
```

can be named

```
profile default /** { ... }
```

Also, a profile with pattern matching can be named. For example:

```
/usr/lib/firefox-3.*/firefox-*bin { ... }
```

can be named

```
profile firefox /usr/lib/firefox-3.*/firefox-*bin { ... }
```

22.6.6 Alias Rules

Alias rules provide an alternative way to manipulate profile path mappings to site specific layouts. They are an alternative form of path rewriting to using variables, and are done post variable resolution. The alias rule says to treat rules that have the same source prefix as if the rules are at target prefix.

```
alias /home/ -> /usr/home/
```

All the rules that have a prefix match to `/home/` will provide access to `/usr/home/`. For example

```
/home/username/** r,
```

allows as well access to

```
/usr/home/username/** r,
```

Aliases provide a quick way of remapping rules without the need to rewrite them. They keep the source path still accessible—in our example, the alias rule keeps the paths under `/home/` still accessible.

With the `alias` rule, you can point to multiple targets at the same time.

```
alias /home/ -> /usr/home/
alias /home/ -> /mnt/home/
```

Note

With the current AppArmor tools, alias rules can only be used when manually editing and maintaining a profile.

Tip

Insert global alias definitions in the file `/etc/apparmor.d/tunables/alias`.

22.7 File Permission Access Modes

File permission access modes consist of combinations of the following modes:

`r`	Read mode

`w`	Write mode (mutually exclusive to `a`)
`a`	Append mode (mutually exclusive to `w`)
`k`	File locking mode
`l`	Link mode
`link FILE -> TARGET`	Link pair rule (cannot be combined with other access modes)

22.7.1 Read Mode (r)

Allows the program to have read access to the resource. Read access is required for shell scripts and other interpreted content and determines if an executing process can core dump.

22.7.2 Write Mode (w)

Allows the program to have write access to the resource. Files must have this permission if they are to be unlinked (removed).

22.7.3 Append Mode (a)

Allows a program to write to the end of a file. In contrast to the `w` mode, the append mode does not include the ability to overwrite data, to rename, or to remove a file. The append permission is typically used with applications who need to be able to write to log files, but which should not be able to manipulate any existing data in the log files. As the append permission is a subset of the permissions associated with the write mode, the `w` and `a` permission flags cannot be used together and are mutually exclusive.

22.7.4 File Locking Mode (k)

The application can take file locks. Former versions of AppArmor allowed files to be locked if an application had access to them. By using a separate file locking mode, AppArmor makes sure locking is restricted only to those files which need file locking and tightens security as locking can be used in several denial of service attack scenarios.

22.7.5 Link Mode (l)

The link mode mediates access to hard links. When a link is created, the target file must have the same access permissions as the link created (but the destination does not need link access).

22.7.6 Link Pair

The link mode grants permission to link to arbitrary files, provided the link has a subset of the permissions granted by the target (subset permission test).

```
/srv/www/htdocs/index.html rl,
```

By specifying origin and destination, the link pair rule provides greater control over how hard links are created. Link pair rules by default do not enforce the link subset permission test that the standard rules link permission requires.

```
link /srv/www/htdocs/index.html -> /var/www/index.html
```

To force the rule to require the test, the `subset` keyword is used. The following rules are equivalent:

```
/var/www/index.html l,
link subset /var/www/index.html -> /**,
```

Note

Currently link pair rules are not supported by YaST and the command line tools. Manually edit your profiles to use them. Updating such profiles using the tools is safe, because the link pair entries will not be touched.

22.7.7 Optional `allow` and `file` Rules

The `allow` prefix is optional, and it is idiomatically implied if not specified and the `deny` (see *Section 22.7.9, "Deny Rules"*) keyword is not used.

```
allow file /example r,
allow /example r,
allow network,
```

You can also use the optional `file` keyword. If you omit it and there are no other rule types that start with a keyword, such as `network` or `mount`, it is automatically implied.

```
file /example/rule r,
```

is equivalent to

```
/example/rule r,
```

The following rule grants access to all files:

```
file,
```

which is equal to

```
/** rwmlk,
```

File rules can use leading or trailing permissions. The permissions should not be specified as a trailing permission, but rather used at the start of the rule. This is important in that it makes file rules behave like any other rule types.

```
/path rw,            # old style
rw /path,            # leading permission
file rw /path,       # with explicit 'file' keyword
allow file rw /path, # optional 'allow' keyword added
```

22.7.8 Owner Conditional Rules

The file rules can be extended so that they can be conditional upon the user being the owner of the file (the fsuid needs to match the file's uid). For this purpose the `owner` keyword is put in front of the rule. Owner conditional rules accumulate like regular file rules do.

```
owner /home/*/** rw
```

When using file ownership conditions with link rules the ownership test is done against the target file so the user must own the file to be able to link to it.

Note: Precedence of Regular File Rules

Owner conditional rules are considered a subset of regular file rules. If a regular file rule overlaps with an owner conditional file rule, the rules are merged. Consider the following example.

```
/foo r,
owner /foo rw,  # or w,
```

The rules are merged—it results in `r` for everybody, and `w` for the owner only.

Tip

To address everybody *but* the owner of the file, use the keyword `other`.

```
owner /foo rw,
other /foo r,
```

22.7.9 Deny Rules

Deny rules can be used to annotate or quiet known rejects. The profile generating tools will not ask about a known reject treated with a deny rule. Such a reject will also not show up in the audit logs when denied, keeping the log files lean. If this is not desired, put the keyword `audit` in front of the deny entry.

It is also possible to use deny rules in combination with allow rules. This allows you to specify a broad allow rule, and then subtract a few known files that should not be allowed. Deny rules can also be combined with owner rules, to deny files owned by the user. The following example allows read/write access to everything in a users directory except write access to the `.ssh/` files:

```
deny /home/*/.ssh/** w,
owner /home/*/** rw,
```

The extensive use of deny rules is generally not encouraged, because it makes it much harder to understand what a profile does. However a judicious use of deny rules can simplify profiles. Therefore the tools only generate profiles denying specific files and will not use globbing in deny rules. Manually edit your profiles to add deny rules using globbing. Updating such profiles using the tools is safe, because the deny entries will not be touched.

22.8 Mount Rules

AppArmor can limit mount and unmount operations, including file system types and mount flags. The rule syntax is based on the **`mount`** command syntax and starts with one of the keywords `mount`, `remount`, or `umount`. Conditions are optional and unspecified conditionals are assumed to match all entries. For example, not specifying a file system means that all file systems are matched.

Conditionals can be specified by specifying conditionals with `options=` or `options in`.

`options=` specifies conditionals that have to be met exactly. The rule

```
mount options=ro /dev/foo -E /mnt/,
```

matches

```
root # mount -o ro /dev/foo /mnt
```

but does not match

```
root # mount -o ro,atime /dev/foo /mnt
root # mount -o rw /dev/foo /mnt
```

`options in` requires that at least one of the listed mount options is used. The rule

```
mount options in (ro,atime) /dev/foo -> /mnt/,
```

matches

```
root # mount -o ro /dev/foo /mnt
root # mount -o ro,atime /dev/foo /mnt
root # mount -o atime /dev/foo /mnt
```

but does not match

```
root # mount -o ro,sync /dev/foo /mnt
root # mount -o ro,atime,sync /dev/foo /mnt
root # mount -o rw /dev/foo /mnt
root # mount -o rw,noatime /dev/foo /mnt
root # mount /dev/foo /mnt
```

With multiple conditionals, the rule grants permission for each set of options. The rule

```
mount options=ro options=atime
```

matches

```
root # mount -o ro /dev/foo /mnt
root # mount -o atime /dev/foo /mnt
```

but does not match

```
root # mount -o ro,atime /dev/foo /mnt
```

Separate mount rules are distinct and the options do not accumulate. The rules

```
mount options=ro,
mount options=atime,
```

are not equivalent to

```
mount options=(ro,atime),
mount options in (ro,atime),
```

The following rule allows mounting `/dev/foo` on `/mnt/` read only and using inode access times or allows mounting `/dev/foo` on `/mnt/` with some combination of 'nodev' and 'user'.

```
mount options=(ro, atime) options in (nodev, user) /dev/foo -> /mnt/,
```

allows

```
root # mount -o ro,atime /dev/foo /mnt
root # mount -o nodev /dev/foo /mnt
root # mount -o user /dev/foo /mnt
root # mount -o nodev,user /dev/foo /mnt
```

22.9 Pivot Root Rules

AppArmor can limit changing the root file system. The syntax is

```
pivot_root [oldroot=OLD_ROOT] NEW_ROOT
```

The paths specified in 'pivot_root' rules must end with '/' since they are directories.

```
# Allow any pivot
pivot_root,

# Allow pivoting to any new root directory and putting the old root
# directory at /mnt/root/old/
pivot_root oldroot=/mnt/root/old/,

# Allow pivoting the root directory to /mnt/root/
pivot_root /mnt/root/,

# Allow pivoting to /mnt/root/ and putting the old root directory at
# /mnt/root/old/
pivot_root oldroot=/mnt/root/old/ /mnt/root/,

# Allow pivoting to /mnt/root/, putting the old root directory at
# /mnt/root/old/ and transition to the /mnt/root/sbin/init profile
```

```
pivot_root oldroot=/mnt/root/old/ /mnt/root/ -> /mnt/root/sbin/init,
```

22.10 PTrace Rules

AppArmor supports limiting ptrace system calls. ptrace rules are accumulated so that the granted ptrace permissions are the union of all the listed ptrace rule permissions. If a rule does not specify an access list, permissions are implicitly granted.

The `trace` and `tracedby` permissions control ptrace(2); `read` and `readby` control proc(5) file system access, kcmp(2), futexes (get_robust_list(2)) and perf trace events.

For a ptrace operation to be allowed, the tracing and traced processes need the correct permissions. This means that the tracing process needs the `trace` permission and the traced process needs the `tracedby` permission.

Example AppArmor PTrace rules:

```
# Allow all PTrace access
ptrace,

# Explicitly allow all PTrace access,
ptrace (read, readby, trace, tracedby),

# Explicitly deny use of ptrace(2)
deny ptrace (trace),

# Allow unconfined processes (eg, a debugger) to ptrace us
ptrace (readby, tracedby) peer=unconfined,

# Allow ptrace of a process running under the /usr/bin/foo profile
ptrace (trace) peer=/usr/bin/foo,
```

22.11 Signal Rules

AppArmor supports limiting inter-process signals. AppArmor signal rules are accumulated so that the granted signal permissions are the union of all the listed signal rule permissions. AppArmor signal permissions are implied when a rule does not explicitly state an access list.

The sending and receiving process must both have the correct permissions.

Example signal rules:

```
# Allow all signal access
```

```
signal,

# Explicitly deny sending the HUP and INT signals
deny signal (send) set=(hup, int),

# Allow unconfined processes to send us signals
signal (receive) peer=unconfined,

# Allow sending of signals to a process running under the /usr/bin/foo
# profile
signal (send) peer=/usr/bin/foo,

# Allow checking for PID existence
signal (receive, send) set=("exists"),

# Allow us to signal ourselves using the built-in @{profile_name} variable
signal peer=@{profile_name},

# Allow two real-time signals
signal set=(rtmin+0 rtmin+32),
```

22.12 Execute Modes

Execute modes, also named profile transitions, consist of the following modes:

`Px`	Discrete profile execute mode
`Cx`	Discrete local profile execute mode
`Ux`	Unconfined execute mode
`ix`	Inherit execute mode
`m`	Allow `PROT_EXEC` with **`mmap(2)`** calls

22.12.1 Discrete Profile Execute Mode (Px)

This mode requires that a discrete security profile is defined for a resource executed at an AppArmor domain transition. If there is no profile defined, the access is denied.

Incompatible with `Ux`, `ux`, `px`, and `ix`.

22.12.2 Discrete Local Profile Execute Mode (Cx)

As `Px`, but instead of searching the global profile set, `Cx` only searches the local profiles of the current profile. This profile transition provides a way for an application to have alternate profiles for helper applications.

Note: Limitations of the Discrete Local Profile Execute Mode (Cx)

Currently, Cx transitions are limited to top level profiles and cannot be used in hats and children profiles. This restriction will be removed in the future.

Incompatible with `Ux`, `ux`, `Px`, `px`, `cx`, and `ix`.

22.12.3 Unconfined Execute Mode (Ux)

Allows the program to execute the resource without any AppArmor profile applied to the executed resource. This mode is useful when a confined program needs to be able to perform a privileged operation, such as rebooting the machine. By placing the privileged section in another executable and granting unconfined execution rights, it is possible to bypass the mandatory constraints imposed on all confined processes. Allowing a root process to go unconfined means it can change AppArmor policy itself. For more information about what is constrained, see the `apparmor(7)` man page.

This mode is incompatible with `ux`, `px`, `Px`, and `ix`.

22.12.4 Unsafe Exec Modes

Use the lowercase versions of exec modes—`px`, `cx`, `ux`—only in very special cases. They do not scrub the environment of variables such as `LD_PRELOAD`. As a result, the calling domain may have an undue amount of influence over the called resource. Use these modes only if the child absolutely *must* be run unconfined and `LD_PRELOAD` must be used. Any profile using such modes provides negligible security. Use at your own risk.

22.12.5 Inherit Execute Mode (ix)

`ix` prevents the normal AppArmor domain transition on **`execve(2)`** when the profiled program executes the named program. Instead, the executed resource inherits the current profile.

This mode is useful when a confined program needs to call another confined program without gaining the permissions of the target's profile or losing the permissions of the current profile. There is no version to scrub the environment because `ix` executions do not change privileges.

Incompatible with `cx`, `ux`, and `px`. Implies `m`.

22.12.6 Allow Executable Mapping (m)

This mode allows a file to be mapped into memory using **`mmap(2)`**'s `PROT_EXEC` flag. This flag marks the pages executable. It is used on some architectures to provide non executable data pages, which can complicate exploit attempts. AppArmor uses this mode to limit which files a well-behaved program (or all programs on architectures that enforce non executable memory access controls) may use as libraries, to limit the effect of invalid `-L` flags given to **`ld(1)`** and `LD_PRELOAD`, `LD_LIBRARY_PATH`, given to **`ld.so(8)`**.

22.12.7 Named Profile Transitions

By default, the `px` and `cx` (and their clean exec variants, too) transition to a profile whose name matches the executable name. With named profile transitions, you can specify a profile to be transitioned to. This is useful if multiple binaries need to share a single profile, or if they need to use a different profile than their name would specify. Named profile transitions can be used with `cx`, `Cx`, `px` and `Px`. Currently there is a limit of twelve named profile transitions per profile.

Named profile transitions use `->` to indicate the name of the profile that needs to be transitioned to:

```
/usr/bin/foo
{
  /bin/** px -> shared_profile,
  ...
  /usr/*bash cx -> local_profile,
  ...
  profile local_profile
  {
    ...
  }
```

```
}
```

Note: Difference Between Normal and Named Transitions

When used with globbing, normal transitions provide a “one to many” relationship—`/bin/** px` will transition to `/bin/ping`, `/bin/cat`, etc, depending on the program being run.

Named transitions provide a “many to one” relationship—all programs that match the rule regardless of their name will transition to the specified profile.

Named profile transitions show up in the log as having the mode `Nx`. The name of the profile to be changed to is listed in the `name2` field.

22.12.8 Fallback Modes for Profile Transitions

The `px` and `cx` transitions specify a hard dependency—if the specified profile does not exist, the exec will fail. With the inheritance fallback, the execution will succeed but inherit the current profile. To specify inheritance fallback, `ix` is combined with `cx`, `Cx`, `px` and `Px` into the modes `cix`, `Cix`, `pix` and `Pix`.

```
/path Cix -> profile_name,
```

or

```
Cix /path -> profile_name,
```

where `-> profile_name` is optional.

The same applies if you add the unconfined `ux` mode, where the resulting modes are `cux`, `CUx`, `pux` and `PUx`. These modes allow falling back to “unconfined” when the specified profile is not found.

```
/path PUx -> profile_name,
```

or

```
PUx /path -> profile_name,
```

where `-> profile_name` is optional.

The fallback modes can be used with named profile transitions, too.

22.12.9 Variable Settings in Execution Modes

When choosing one of the Px, Cx or Ux execution modes, take into account that the following environment variables are removed from the environment before the child process inherits it. As a consequence, applications or processes relying on any of these variables do not work anymore if the profile applied to them carries Px, Cx or Ux flags:

- `GCONV_PATH`
- `GETCONF_DIR`
- `HOSTALIASES`
- `LD_AUDIT`
- `LD_DEBUG`
- `LD_DEBUG_OUTPUT`
- `LD_DYNAMIC_WEAK`
- `LD_LIBRARY_PATH`
- `LD_ORIGIN_PATH`
- `LD_PRELOAD`
- `LD_PROFILE`
- `LD_SHOW_AUXV`
- `LD_USE_LOAD_BIAS`
- `LOCALDOMAIN`
- `LOCPATH`
- `MALLOC_TRACE`
- `NLSPATH`
- `RESOLV_HOST_CONF`
- `RES_OPTIONS`
- `TMPDIR`
- `TZDIR`

22.12.10 `safe` and `unsafe` Keywords

You can use the `safe` and `unsafe` keywords for rules instead of using the case modifier of execution modes. For example

```
/example_rule Px,
```

is the same as any of the following

```
safe /example_rule px,
safe /example_rule Px,
safe px /example_rule,
safe Px /example_rule,
```

and the rule

```
/example_rule px,
```

is the same as any of

```
unsafe /example_rule px,
unsafe /example_rule Px,
unsafe px /example_rule,
unsafe Px /example_rule,
```

The `safe`/`unsafe` keywords are mutually exclusive and can be used in a file rule after the `owner` keyword, so the order of rule keywords is

```
[audit] [deny] [owner] [safe|unsafe] file_rule
```

22.13 Resource Limit Control

AppArmor can set and control an application's resource limits (rlimits, also known as ulimits). By default, AppArmor does not control application's rlimits, and it will only control those limits specified in the confining profile. For more information about resource limits, refer to the `setrlimit(2)`, `ulimit(1)`, or `ulimit(3)` man pages.

AppArmor leverages the system's rlimits and as such does not provide an additional auditing that would normally occur. It also cannot raise rlimits set by the system, AppArmor rlimits can only reduce an application's current resource limits.

The values will be inherited by the children of a process and will remain even if a new profile is transitioned to or the application becomes unconfined. So when an application transitions to a new profile, that profile can further reduce the application's rlimits.

AppArmor's rlimit rules will also provide mediation of setting an application's hard limits, should it try to raise them. The application cannot raise its hard limits any further than specified in the profile. The mediation of raising hard limits is not inherited as the set value is, so that when the application transitions to a new profile it is free to raise its limits as specified in the profile.

AppArmor's rlimit control does not affect an application's soft limits beyond ensuring that they are less than or equal to the application's hard limits.

AppArmor's hard limit rules have the general form of:

```
set rlimit RESOURCE <= value,
```

where *RESOURCE* and *VALUE* are to be replaced with the following values:

`cpu`
: CPU time limit in seconds.

`fsize`, `data`, `stack`, `core`, `rss`, `as`, `memlock`, `msgqueue`
: a number in bytes, or a number with a suffix where the suffix can be K/KB (kilobytes), M/MB (megabytes), G/GB (gigabytes), for example

  ```
  rlimit data <= 100M,
  ```

`fsize`, `nofile`, `locks`, `sigpending`, `nproc`*, `rtprio`
: a number greater or equal to 0

`nice`
: a value between -20 and 19

*The nproc rlimit is handled different than all the other rlimits. Instead of indicating the standard process rlimit it controls the maximum number of processes that can be running under the profile at any time. When the limit is exceeded the creation of new processes under the profile will fail until the number of currently running processes is reduced.

Note

Currently the tools cannot be used to add rlimit rules to profiles. The only way to add rlimit controls to a profile is to manually edit the profile with a text editor. The tools will still work with profiles containing rlimit rules and will not remove them, so it is safe to use the tools to update profiles containing them.

22.14 Auditing Rules

AppArmor provides the ability to audit given rules so that when they are matched an audit message will appear in the audit log. To enable audit messages for a given rule, the `audit` keyword is put in front of the rule:

```
audit /etc/foo/*        rw,
```

If it is desirable to audit only a given permission the rule can be split into two rules. The following example will result in audit messages when files are opened for writing, but not when they are opened for reading:

```
audit /etc/foo/*  w,
/etc/foo/*        r,
```

Note

Audit messages are not generated for every read or write of a file but only when a file is opened for reading or writing.

Audit control can be combined with `owner`/`other` conditional file rules to provide auditing when users access files they own/do not own:

```
audit owner /home/*/.ssh/**       rw,
audit other /home/*/.ssh/**       r,
```

23 AppArmor Profile Repositories

AppArmor ships with a set of profiles enabled by default. These are created by the AppArmor developers, and are stored in `/etc/apparmor.d`. In addition to these profiles, openSUSE Leap ships profiles for individual applications together with the relevant application. These profiles are not enabled by default, and reside under another directory than the standard AppArmor profiles, `/etc/apparmor/profiles/extras`.

The AppArmor tools (YaST, **`aa-genprof`** and **`aa-logprof`**) support the use of a local repository. Whenever you start to create a new profile from scratch, and there already is an inactive profile in your local repository, you are asked whether you want to use the existing inactive one from `/etc/apparmor/profiles/extras` and whether you want to base your efforts on it. If you decide to use this profile, it gets copied over to the directory of profiles enabled by default (`/etc/apparmor.d`) and loaded whenever AppArmor is started. Any further adjustments will be done to the active profile under `/etc/apparmor.d`.

24 Building and Managing Profiles with YaST

YaST provides a basic way to build profiles and manage AppArmor® profiles. It provides two interfaces: a graphical one and a text-based one. The text-based interface consumes less resources and bandwidth, making it a better choice for remote administration, or for times when a local graphical environment is inconvenient. Although the interfaces have differing appearances, they offer the same functionality in similar ways. Another alternative is to use AppArmor commands, which can control AppArmor from a terminal window or through remote connections. The command line tools are described in *Chapter 25, Building Profiles from the Command Line*.

Start YaST from the main menu and enter your `root` password when prompted for it. Alternatively, start YaST by opening a terminal window, logging in as `root`, and entering **`yast2`** for the graphical mode or **`yast`** for the text-based mode.

In the *Security and Users* section, there is an *AppArmor Configuration* icon. Click it to launch the AppArmor YaST module.

24.1 Manually Adding a Profile

AppArmor enables you to create an AppArmor profile by manually adding entries into the profile. Select the application for which to create a profile, then add entries.

1. Start YaST, select *AppArmor Configuration*, and click *Manually Add Profile* in the main window.
2. Browse your system to find the application for which to create a profile.
3. When you find the application, select it and click *Open*. A basic, empty profile appears in the *AppArmor Profile Dialog* window.
4. In *AppArmor Profile Dialog*, add, edit, or delete AppArmor profile entries by clicking the corresponding buttons and referring to *Section 24.2.1, “Adding an Entry”*, *Section 24.2.2, “Editing an Entry”*, or *Section 24.2.3, “Deleting an Entry”*.
5. When finished, click *Done*.

24.2 Editing Profiles

Tip

YaST offers basic manipulation for AppArmor profiles, such as creating or editing. However, the most straightforward way to edit an AppArmor profile is to use a text editor such as **vi**:

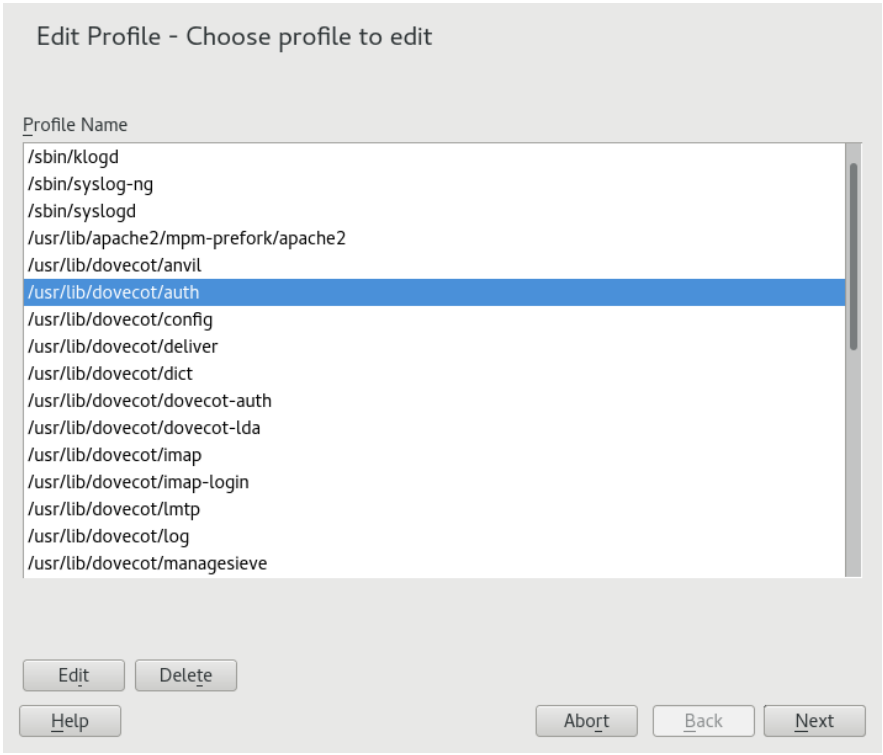
```
tux > sudo vi /etc/apparmor.d/usr.sbin.httpd2-prefork
```

Tip

The **vi** editor also includes syntax (error) highlighting and syntax error highlighting, which visually warns you when the syntax of the edited AppArmor profile is wrong.

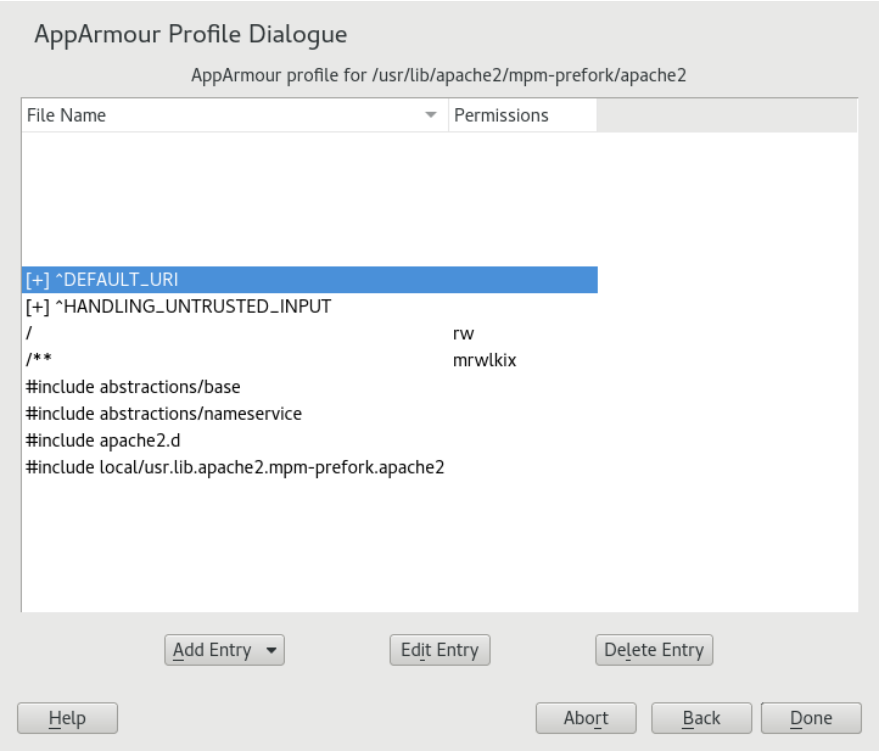
AppArmor enables you to edit AppArmor profiles manually by adding, editing, or deleting entries. To edit a profile, proceed as follows:

1. Start YaST, select *AppArmor Configuration*, and click *Manage Existing Profiles* in the main window.



2. From the list of profiled applications, select the profile to edit.

3. Click *Edit*. The *AppArmor Profile Dialog* window displays the profile.



4. In the *AppArmor Profile Dialog* window, add, edit, or delete AppArmor profile entries by clicking the corresponding buttons and referring to *Section 24.2.1, "Adding an Entry"*, *Section 24.2.2, "Editing an Entry"*, or *Section 24.2.3, "Deleting an Entry"*.

5. When you are finished, click *Done*.

6. In the pop-up that appears, click *Yes* to confirm your changes to the profile and reload the AppArmor profile set.

Tip: Syntax Checking in AppArmor

AppArmor contains a syntax check that notifies you of any syntax errors in profiles you are trying to process with the YaST AppArmor tools. If an error occurs, edit the profile manually as `root` and reload the profile set with **`systemctl reload apparmor`**.

24.2.1 Adding an Entry

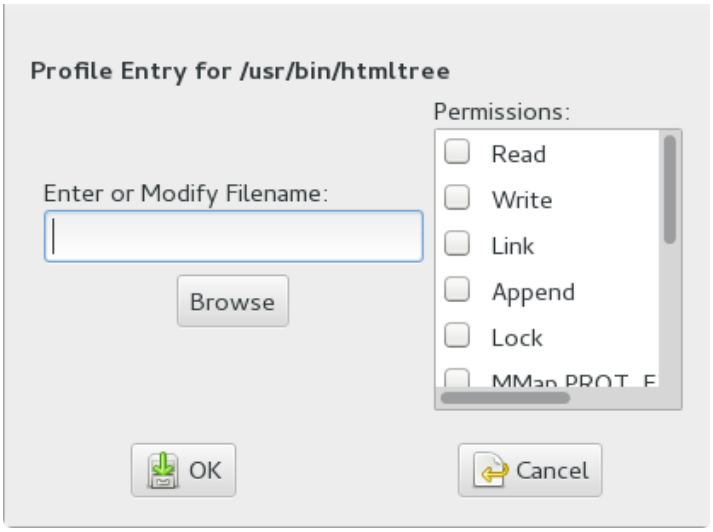
The *Add Entry* button in the *AppArmor Profile Window* lists types of entries you can add to the AppArmor profile.

From the list, select one of the following:

File

In the pop-up window, specify the absolute path of a file, including the type of access permitted. When finished, click *OK*.

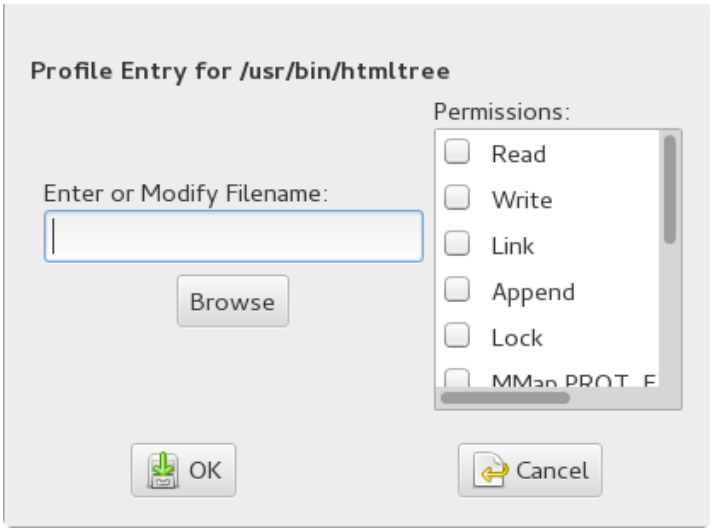
You can use globbing if necessary. For globbing information, refer to *Section 22.6, "Profile Names, Flags, Paths, and Globbing"*. For file access permission information, refer to *Section 22.7, "File Permission Access Modes"*.



Directory

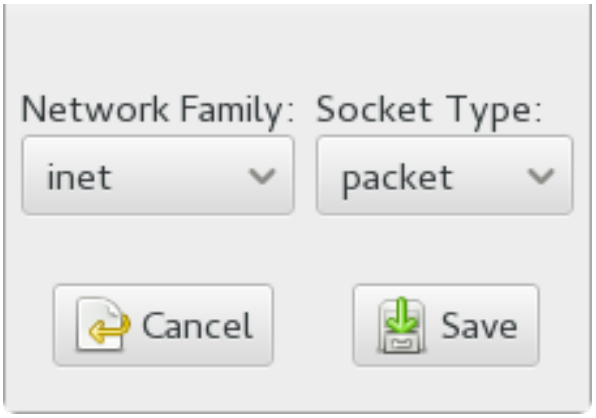
In the pop-up window, specify the absolute path of a directory, including the type of access permitted. You can use globbing if necessary. When finished, click *OK*.

For globbing information, refer to *Section 22.6, "Profile Names, Flags, Paths, and Globbing"*. For file access permission information, refer to *Section 22.7, "File Permission Access Modes"*.



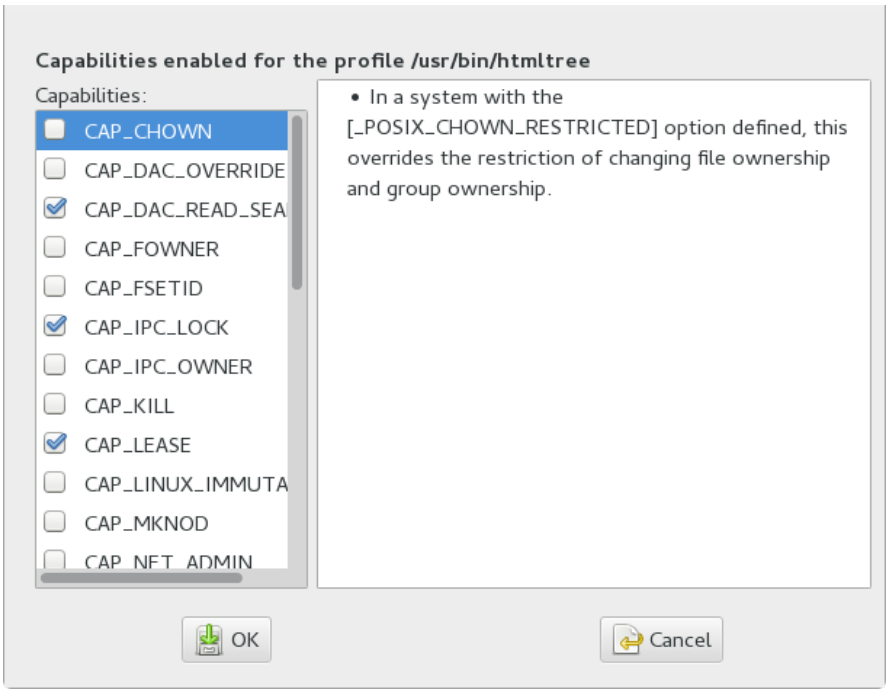
Network Rule

In the pop-up window, select the appropriate network family and the socket type. For more information, refer to *Section 22.5, "Network Access Control"*.



Capability

In the pop-up window, select the appropriate capabilities. These are statements that enable each of the 32 POSIX.1e capabilities. Refer to *Section 22.4, "Capability Entries (POSIX.1e)"* for more information about capabilities. When finished making your selections, click *OK*.



Include File

In the pop-up window, browse to the files to use as includes. Includes are directives that pull in components of other AppArmor profiles to simplify profiles. For more information, refer to *Section 22.3, "Include Statements"*.

Hat

In the pop-up window, specify the name of the subprofile (*hat*) to add to your current profile and click *Create Hat*. For more information, refer to *Chapter 26, Profiling Your Web Applications Using ChangeHat*.

24.2.2 Editing an Entry

When you select *Edit Entry*, a pop-up window opens. From here, edit the selected entry.

In the pop-up window, edit the entry you need to modify. You can use globbing if necessary. When finished, click *OK*.

For globbing information, refer to *Section 22.6, "Profile Names, Flags, Paths, and Globbing"*. For access permission information, refer to *Section 22.7, "File Permission Access Modes"*.

24.2.3 Deleting an Entry

To delete an entry in a given profile, select *Delete Entry*. AppArmor removes the selected profile entry.

24.3 Deleting a Profile

AppArmor enables you to delete an AppArmor profile manually. Simply select the application for which to delete a profile then delete it as follows:

1. Start YaST, select *AppArmor Configuration*, and click *Manage Existing Profiles* in the main window.
2. Select the profile to delete.
3. Click *Delete*.
4. In the pop-up that opens, click *Yes* to delete the profile and reload the AppArmor profile set.

24.4 Managing AppArmor

You can change the status of AppArmor by enabling or disabling it. Enabling AppArmor protects your system from potential program exploitation. Disabling AppArmor, even if your profiles have been set up, removes protection from your system. To change the status of AppArmor, start YaST, select *AppArmor Configuration*, and click *Settings* in the main window.

To change the status of AppArmor, continue as described in *Section 24.4.1, "Changing AppArmor Status"*. To change the mode of individual profiles, continue as described in *Section 24.4.2, "Changing the Mode of Individual Profiles"*.

24.4.1 Changing AppArmor Status

When you change the status of AppArmor, set it to enabled or disabled. When AppArmor is enabled, it is installed, running, and enforcing the AppArmor security policies.

1. Start YaST, select *AppArmor Configuration*, and click *Settings* in the main window.
2. Enable AppArmor by checking *Enable AppArmor* or disable AppArmor by deselecting it.
3. Click *Done* in the *AppArmor Configuration* window.

Tip

You always need to restart running programs to apply the profiles to them.

24.4.2 Changing the Mode of Individual Profiles

AppArmor can apply profiles in two different modes. In *complain* mode, violations of AppArmor profile rules, such as the profiled program accessing files not permitted by the profile, are detected. The violations are permitted, but also logged. This mode is convenient for developing profiles and is used by the AppArmor tools for generating profiles. Loading a profile in *enforce* mode enforces the policy defined in the profile, and reports policy violation attempts to `rsyslogd` (or `auditd` or `journalctl`, depending on system configuration).

The *Profile Mode Configuration* dialog allows you to view and edit the mode of currently loaded AppArmor profiles. This feature is useful for determining the status of your system during profile development. During systemic profiling (see *Section 25.7.2, "Systemic Profiling"*), you can use this tool to adjust and monitor the scope of the profiles for which you are learning behavior.

To edit an application's profile mode, proceed as follows:

1. Start YaST, select *AppArmor Configuration*, and click *Settings* in the main window.
2. In the *Configure Profile Modes* section, select *Configure*.
3. Select the profile for which to change the mode.
4. Select *Toggle Mode* to set this profile to *complain* mode or to *enforce* mode.
5. Apply your settings and leave YaST with *Done*.

To change the mode of all profiles, use *Set All to Enforce* or *Set All to Complain*.

Tip: Listing the Profiles Available

By default, only active profiles are listed (any profile that has a matching application installed on your system). To set up a profile before installing the respective application, click *Show All Profiles* and select the profile to configure from the list that appears.

25 Building Profiles from the Command Line

AppArmor® provides the user the ability to use a command line interface rather than a graphical interface to manage and configure the system security. Track the status of AppArmor and create, delete, or modify AppArmor profiles using the AppArmor command line tools.

Tip: Background Information

Before starting to manage your profiles using the AppArmor command line tools, check out the general introduction to AppArmor given in *Chapter 21, Immunizing Programs* and *Chapter 22, Profile Components and Syntax*.

25.1 Checking the AppArmor Status

AppArmor can be in any one of three states:

Unloaded

AppArmor is not activated in the kernel.

Running

AppArmor is activated in the kernel and is enforcing AppArmor program policies.

Stopped

AppArmor is activated in the kernel, but no policies are enforced.

Detect the state of AppArmor by inspecting `/sys/kernel/security/apparmor/profiles`. If `cat /sys/kernel/security/apparmor/profiles` reports a list of profiles, AppArmor is running. If it is empty and returns nothing, AppArmor is stopped. If the file does not exist, AppArmor is unloaded.

Manage AppArmor with `systemctl`. It lets you perform the following operations:

`sudo systemctl start apparmor`

Behavior depends on the state of AppArmor. If it is not activated, `start` activates and starts it, putting it in the running state. If it is stopped, `start` causes the re-scan of AppArmor profiles usually found in `/etc/apparmor.d` and puts AppArmor in the running state. If AppArmor is already running, `start` reports a warning and takes no action.

Note: Already Running Processes

Already running processes need to be restarted to apply the AppArmor profiles on them.

`sudo systemctl stop apparmor`

Stops AppArmor if it is running by removing all profiles from kernel memory, effectively disabling all access controls, and putting AppArmor into the stopped state. If the AppArmor is already stopped, `stop` tries to unload the profiles again, but nothing happens.

`sudo systemctl reload apparmor`

Causes the AppArmor module to re-scan the profiles in `/etc/apparmor.d` without unconfining running processes. Freshly created profiles are enforced and recently deleted ones are removed from the `/etc/apparmor.d` directory.

25.2 Building AppArmor Profiles

The AppArmor module profile definitions are stored in the `/etc/apparmor.d` directory as plain text files. For a detailed description of the syntax of these files, refer to *Chapter 22, Profile Components and Syntax*.

All files in the `/etc/apparmor.d` directory are interpreted as profiles and are loaded as such. Renaming files in that directory is not an effective way of preventing profiles from being loaded. You must remove profiles from this directory to prevent them from being read and evaluated effectively, or call **`aa-disable`** on the profile, which will create a symbolic link in `/etc/apparmor.d/disabled/`.

You can use a text editor, such as **`vi`**, to access and make changes to these profiles. The following sections contain detailed steps for building profiles:

Adding or Creating AppArmor Profiles

Refer to *Section 25.3, “Adding or Creating an AppArmor Profile”*

Editing AppArmor Profiles

Refer to *Section 25.4, “Editing an AppArmor Profile”*

Deleting AppArmor Profiles

Refer to *Section 25.6, “Deleting an AppArmor Profile”*

25.3 Adding or Creating an AppArmor Profile

To add or create an AppArmor profile for an application, you can use a systemic or standalone profiling method, depending on your needs. Learn more about these two approaches in *Section 25.7, "Two Methods of Profiling"*.

25.4 Editing an AppArmor Profile

The following steps describe the procedure for editing an AppArmor profile:

1. If you are not currently logged in as `root`, enter **`su`** in a terminal window.
2. Enter the `root` password when prompted.
3. Go to the profile directory with **`cd /etc/apparmor.d/`**.
4. Enter **`ls`** to view all profiles currently installed.
5. Open the profile to edit in a text editor, such as vim.
6. Make the necessary changes, then save the profile.
7. Restart AppArmor by entering **`systemctl reload apparmor`** in a terminal window.

25.5 Unloading Unknown AppArmor Profiles

Warning: Danger of Unloading Wanted Profiles

`aa-remove-unknown` will unload all profiles that are not stored in `/etc/apparmor.d`, for example automatically generated LXD profiles. This may compromise the security of the system. Use the `-n` parameter to list all profiles that will be unloaded.

To unload all AppArmor profiles that are no longer in `/etc/apparmor.d/`, run:

```
tux > sudo aa-remove-unknown
```

You can print a list of profiles that will be removed:

```
tux > sudo aa-remove-unknown -n
```

25.6 Deleting an AppArmor Profile

The following steps describe the procedure for deleting an AppArmor profile.

1. Remove the AppArmor definition from the kernel:

```
tux > sudo apparmor_parser -R /etc/apparmor.d/PROFILE
```

2. Remove the definition file:

```
tux > sudo rm /etc/apparmor.d/PROFILE
    tux > sudo rm /var/lib/apparmor/cache/PROFILE
```

25.7 Two Methods of Profiling

Given the syntax for AppArmor profiles in *Chapter 22, Profile Components and Syntax*, you could create profiles without using the tools. However, the effort involved would be substantial. To avoid such a situation, use the AppArmor tools to automate the creation and refinement of profiles.

There are two ways to approach AppArmor profile creation. Tools are available for both methods.

Stand-Alone Profiling

A method suitable for profiling small applications that have a finite runtime, such as user client applications like mail clients. For more information, refer to *Section 25.7.1, "Stand-Alone Profiling"*.

Systemic Profiling

A method suitable for profiling many programs at once and for profiling applications that may run for days, weeks, or continuously across reboots, such as network server applications like Web servers and mail servers. For more information, refer to *Section 25.7.2, "Systemic Profiling"*.

Automated profile development becomes more manageable with the AppArmor tools:

1. Decide which profiling method suits your needs.
2. Perform a static analysis. Run either **aa-genprof** or **aa-autodep**, depending on the profiling method chosen.

3. Enable dynamic learning. Activate learning mode for all profiled programs.

25.7.1 Stand-Alone Profiling

Stand-alone profile generation and improvement is managed by a program called **aa-genprof**. This method is easy because **aa-genprof** takes care of everything, but is limited because it requires **aa-genprof** to run for the entire duration of the test run of your program (you cannot reboot the machine while you are still developing your profile).

To use **aa-genprof** for the stand-alone method of profiling, refer to *Section 25.7.3.8, "aa-genprof —Generating Profiles"*.

25.7.2 Systemic Profiling

This method is called *systemic profiling* because it updates all of the profiles on the system at once, rather than focusing on the one or few targeted by **aa-genprof** or stand-alone profiling. With systemic profiling, profile construction and improvement are somewhat less automated, but more flexible. This method is suitable for profiling long-running applications whose behavior continues after rebooting, or many programs at once.

Build an AppArmor profile for a group of applications as follows:

1. Create profiles for the individual programs that make up your application.
   Although this approach is systemic, AppArmor only monitors those programs with profiles and their children. To get AppArmor to consider a program, you must at least have **aa-autodep** create an approximate profile for it. To create this approximate profile, refer to *Section 25.7.3.1, "aa-autodep—Creating Approximate Profiles"*.

2. Put relevant profiles into learning or complain mode.
   Activate learning or complain mode for all profiled programs by entering

   ```
   tux > sudo aa-complain /etc/apparmor.d/*
   ```

   in a terminal window while logged in as `root`. This functionality is also available through the YaST Profile Mode module, described in *Section 24.4.2, "Changing the Mode of Individual Profiles"*.

When in learning mode, access requests are not blocked, even if the profile dictates that they should be. This enables you to run through several tests (as shown in *Step 3*) and learn the access needs of the program so it runs properly. With this information, you can decide how secure to make the profile.

Refer to *Section 25.7.3.2, "aa-complain—Entering Complain or Learning Mode"* for more detailed instructions for using learning or complain mode.

3. Exercise your application.

   Run your application and exercise its functionality. How much to exercise the program is up to you, but you need the program to access each file representing its access needs. Because the execution is not being supervised by **aa-genprof**, this step can go on for days or weeks and can span complete system reboots.

4. Analyze the log.

   In systemic profiling, run **aa-logprof** directly instead of letting **aa-genprof** run it (as in stand-alone profiling). The general form of **aa-logprof** is:

   ```
   tux > sudo aa-logprof [ -d /path/to/profiles ] [ -f /path/to/logfile ]
   ```

   Refer to *Section 25.7.3.9, "aa-logprof—Scanning the System Log"* for more information about using **aa-logprof**.

5. Repeat *Step 3* and *Step 4*.

   This generates optimal profiles. An iterative approach captures smaller data sets that can be trained and reloaded into the policy engine. Subsequent iterations generate fewer messages and run faster.

6. Edit the profiles.

   You should review the profiles that have been generated. You can open and edit the profiles in `/etc/apparmor.d/` using a text editor.

7. Return to enforce mode.

   This is when the system goes back to enforcing the rules of the profiles, not only logging information. This can be done manually by removing the `flags=(complain)` text from the profiles or automatically by using the **aa-enforce** command, which works identically to the **aa-complain** command, except it sets the profiles to enforce mode. This functionality is also available through the YaST Profile Mode module, described in *Section 24.4.2, "Changing the Mode of Individual Profiles"*.

   To ensure that all profiles are taken out of complain mode and put into enforce mode, enter **aa-enforce /etc/apparmor.d/***.

8. Re-scan all profiles.

   To have AppArmor re-scan all of the profiles and change the enforcement mode in the kernel, enter **`systemctl reload apparmor`**.

25.7.3 Summary of Profiling Tools

All of the AppArmor profiling utilities are provided by the `apparmor-utils` RPM package and are stored in `/usr/sbin`. Each tool has a different purpose.

25.7.3.1 aa-autodep—Creating Approximate Profiles

This creates an approximate profile for the program or application selected. You can generate approximate profiles for binary executables and interpreted script programs. The resulting profile is called “approximate” because it does not necessarily contain all of the profile entries that the program needs to be properly confined by AppArmor. The minimum **`aa-autodep`** approximate profile has, at minimum, a base include directive, which contains basic profile entries needed by most programs. For certain types of programs, **`aa-autodep`** generates a more expanded profile. The profile is generated by recursively calling **`ldd(1)`** on the executables listed on the command line.

To generate an approximate profile, use the **`aa-autodep`** program. The program argument can be either the simple name of the program, which **`aa-autodep`** finds by searching your shell's path variable, or it can be a fully qualified path. The program itself can be of any type (ELF binary, shell script, Perl script, etc.). **`aa-autodep`** generates an approximate profile to improve through the dynamic profiling that follows.

The resulting approximate profile is written to the `/etc/apparmor.d` directory using the AppArmor profile naming convention of naming the profile after the absolute path of the program, replacing the forward slash (`/`) characters in the path with period (`.`) characters. The general syntax of **`aa-autodep`** is to enter the following in a terminal window:

```
tux > sudo aa-autodep [ -d /PATH/TO/PROFILES ] [PROGRAM1 PROGRAM2...]
```

If you do not enter the program name or names, you are prompted for them. *`/path/to/profiles`* overrides the default location of `/etc/apparmor.d`, should you keep profiles in a location other than the default.

To begin profiling, you must create profiles for each main executable service that is part of your application (anything that might start without being a child of another program that already has a profile). Finding all such programs depends on the application in question. Here are several strategies for finding such programs:

Directories

If all the programs to profile are in one directory and there are no other programs in that directory, the simple command **aa-autodep** */path/to/your/programs/** creates basic profiles for all programs in that directory.

pstree -p

You can run your application and use the standard Linux **pstree** command to find all processes running. Then manually hunt down the location of these programs and run the **aa-autodep** for each one. If the programs are in your path, **aa-autodep** finds them for you. If they are not in your path, the standard Linux command **find** might be helpful in finding your programs. Execute **find / -name '** *MY_APPLICATION*' -print to determine an application's path (*MY_APPLICATION* being an example application). You may use wild cards if appropriate.

25.7.3.2 aa-complain—Entering Complain or Learning Mode

The complain or learning mode tool (**aa-complain**) detects violations of AppArmor profile rules, such as the profiled program accessing files not permitted by the profile. The violations are permitted, but also logged. To improve the profile, turn complain mode on, run the program through a suite of tests to generate log events that characterize the program's access needs, then postprocess the log with the AppArmor tools to transform log events into improved profiles.

Manually activating complain mode (using the command line) adds a flag to the top of the profile so that `/bin/foo` becomes `/bin/foo flags=(complain)`. To use complain mode, open a terminal window and enter one of the following lines as `root`:

- If the example program (*PROGRAM1*) is in your path, use:

```
tux > sudo aa-complain [PROGRAM1 PROGRAM2 ...]
```

- If the program is not in your path, specify the entire path as follows:

```
tux > sudo aa-complain /sbin/PROGRAM1
```

- If the profiles are not in `/etc/apparmor.d`, use the following to override the default location:

```
tux > sudo aa-complain /path/to/profiles/PROGRAM1
```

- Specify the profile for `/sbin/program1` as follows:

```
tux > sudo aa-complain /etc/apparmor.d/sbin.PROGRAM1
```

Each of the above commands activates the complain mode for the profiles or programs listed. If the program name does not include its entire path, **aa-complain** searches `$PATH` for the program. For example, **aa-complain /usr/sbin/*** finds profiles associated with all of the programs in `/usr/sbin` and puts them into complain mode. **aa-complain /etc/apparmor.d/*** puts all of the profiles in `/etc/apparmor.d` into complain mode.

Tip: Toggling Profile Mode with YaST

YaST offers a graphical front-end for toggling complain and enforce mode. See *Section 24.4.2, "Changing the Mode of Individual Profiles"* for information.

25.7.3.3 aa-decode—Decoding Hex-encoded Strings in AppArmor Log Files

aa-decode will decode hex-encoded strings in the AppArmor log output. It can also process the audit log on standard input, convert any hex-encoded AppArmor log entries, and display them on standard output.

25.7.3.4 aa-disable—Disabling an AppArmor Security Profile

Use **aa-disable** to disable the enforcement mode for one or more AppArmor profiles. This command will unload the profile from the kernel, and prevent the profile from being loaded on AppArmor start-up. Use **aa-enforce** or **aa-complain** utilities to change this behavior.

25.7.3.5 aa-easyprof—Easy Profile Generation

aa-easyprof provides an easy-to-use interface for AppArmor profile generation. **aa-easyprof** supports the use of templates and profile groups to quickly profile an application. While **aa-easyprof** can help with profile generation, its utility is dependent on the quality of the templates, profile groups and abstractions used. Also, this tool may create a profile that is less restricted than when creating a profile manually or with **aa-genprof** and **aa-logprof**.

For more information, see the man page of **aa-easyprof** (8).

25.7.3.6 aa-enforce—Entering Enforce Mode

The enforce mode detects violations of AppArmor profile rules, such as the profiled program accessing files not permitted by the profile. The violations are logged and not permitted. The default is for enforce mode to be enabled. To log the violations only, but still permit them, use complain mode.

Manually activating enforce mode (using the command line) removes the complain flag from the top of the profile so that `/bin/foo flags=(complain)` becomes `/bin/foo`. To use enforce mode, open a terminal window and enter one of the following lines.

- If the example program (*PROGRAM1*) is in your path, use:

```
tux > sudo aa-enforce [PROGRAM1 PROGRAM2 ...]
```

- If the program is not in your path, specify the entire path, as follows:

```
tux > sudo aa-enforce /sbin/PROGRAM1
```

- If the profiles are not in */etc/apparmor.d*, use the following to override the default location:

```
tux > sudo aa-enforce -d /path/to/profiles/     program1
```

- Specify the profile for */sbin/program1* as follows:

```
tux > sudo aa-enforce /etc/apparmor.d/sbin.PROGRAM1
```

Each of the above commands activates the enforce mode for the profiles and programs listed.

If you do not enter the program or profile names, you are prompted to enter one. */path/to/profiles* overrides the default location of `/etc/apparmor.d`.

The argument can be either a list of programs or a list of profiles. If the program name does not include its entire path, **aa-enforce** searches `$PATH` for the program.

Tip: Toggling Profile Mode with YaST

YaST offers a graphical front-end for toggling complain and enforce mode. See *Section 24.4.2, "Changing the Mode of Individual Profiles"* for information.

25.7.3.7 aa-exec—Confining a Program with the Specified Profile

Use **aa-exec** to launch a program confined by a specified profile and/or profile namespace. If both a profile and namespace are specified, the program will be confined by the profile in the new namespace. If only a profile namespace is specified, the profile name of the current confinement will be used. If neither a profile nor namespace is specified, the command will be run using the standard profile attachment—as if you did not use the **aa-exec** command.

For more information on the command's options, see its manual page **man 8 aa-exec**.

25.7.3.8 aa-genprof—Generating Profiles

aa-genprof is AppArmor's profile generating utility. It runs **aa-autodep** on the specified program, creating an approximate profile (if a profile does not already exist for it), sets it to complain mode, reloads it into AppArmor, marks the log, and prompts the user to execute the program and exercise its functionality. Its syntax is as follows:

```
tux > sudo aa-genprof [ -d /path/to/profiles ]  PROGRAM
```

To create a profile for the Apache Web server program httpd2-prefork, do the following as `root`:

1. Enter **systemctl stop apache2**.
2. Next, enter **aa-genprof httpd2-prefork**.

Now **aa-genprof** does the following:

1. Resolves the full path of httpd2-prefork using your shell's path variables. You can also specify a full path. On openSUSE Leap, the default full path is `/usr/sbin/httpd2-prefork`.

2. Checks to see if there is an existing profile for httpd2-prefork. If there is one, it updates it. If not, it creates one using the **aa-autodep** as described in *Section 25.7.3, "Summary of Profiling Tools"*.

3. Puts the profile for this program into learning or complain mode so that profile violations are logged, but are permitted to proceed. A log event looks like this (see `/var/log/audit/audit.log`):

```
type=APPARMOR_ALLOWED msg=audit(1189682639.184:20816): \
apparmor="DENIED" operation="file_mmap" parent=2692 \
profile="/usr/sbin/httpd2-prefork//HANDLING_UNTRUSTED_INPUT" \
name="/var/log/apache2/access_log-20140116" pid=28730 comm="httpd2-prefork" \
requested_mask="::r" denied_mask="::r" fsuid=30 ouid=0
```

   If you are not running the audit daemon, the AppArmor events are logged directly to `systemd` journal (see *Book "Reference", Chapter 11 "**journalctl**: Query the* `systemd` *Journal"*):

```
Sep 13 13:20:30 K23 kernel: audit(1189682430.672:20810): \
apparmor="DENIED" operation="file_mmap" parent=2692 \
profile="/usr/sbin/httpd2-prefork//HANDLING_UNTRUSTED_INPUT" \
name="/var/log/apache2/access_log-20140116" pid=28730 comm="httpd2-prefork" \
requested_mask="::r" denied_mask="::r" fsuid=30 ouid=0
```

   They also can be viewed using the **dmesg** command:

```
audit(1189682430.672:20810): apparmor="DENIED" \
operation="file_mmap" parent=2692 \
profile="/usr/sbin/httpd2-prefork//HANDLING_UNTRUSTED_INPUT" \
name="/var/log/apache2/access_log-20140116" pid=28730 comm="httpd2-prefork" \
requested_mask="::r" denied_mask="::r" fsuid=30 ouid=0
```

4. Marks the log with a beginning marker of log events to consider. For example:

```
Sep 13 17:48:52 figwit root: GenProf: e2ff78636296f16d0b5301209a04430d
```

3. When prompted by the tool, run the application to profile in another terminal window and perform as many of the application functions as possible. Thus, the learning mode can log the files and directories to which the program requires access to function properly. For example, in a new terminal window, enter **`systemctl start apache2`**.

4. Select from the following options that are available in the **`aa-genprof`** terminal window after you have executed the program function:

   - S runs **`aa-genprof`** on the system log from where it was marked when **`aa-genprof`** was started and reloads the profile. If system events exist in the log, AppArmor parses the learning mode log files. This generates a series of questions that you must answer to guide **`aa-genprof`** in generating the security profile.
   - F exits the tool.

Note

If requests to add hats appear, proceed to *Chapter 26, Profiling Your Web Applications Using ChangeHat*.

5. Answer two types of questions:

   - A resource is requested by a profiled program that is not in the profile (see *Example 25.1, "Learning Mode Exception: Controlling Access to Specific Resources"*).
   - A program is executed by the profiled program and the security domain transition has not been defined (see *Example 25.2, "Learning Mode Exception: Defining Permissions for an Entry"*).

   Each of these categories results in a series of questions that you must answer to add the resource or program to the profile. *Example 25.1, "Learning Mode Exception: Controlling Access to Specific Resources"* and *Example 25.2, "Learning Mode Exception: Defining Permissions for an Entry"* provide examples of each one. Subsequent steps describe your options in answering these questions.

   - Dealing with execute accesses is complex. You must decide how to proceed with this entry regarding which execute permission type to grant to this entry:

     EXAMPLE 25.1: LEARNING MODE EXCEPTION: CONTROLLING ACCESS TO SPECIFIC RESOURCES

     ```
     Reading log entries from /var/log/audit/audit.log.
     ```

```
Updating AppArmor profiles in /etc/apparmor.d.

Profile:  /usr/sbin/cupsd
Program:  cupsd
Execute:  /usr/lib/cups/daemon/cups-lpd
Severity: unknown

(I)nherit / (P)rofile / (C)hild / (N)ame / (U)nconfined / (X)ix / (D)eny /
 Abo(r)t / (F)inish
```

Inherit (ix)

The child inherits the parent's profile, running with the same access controls as the parent. This mode is useful when a confined program needs to call another confined program without gaining the permissions of the target's profile or losing the permissions of the current profile. This mode is often used when the child program is a *helper application*, such as the **/usr/bin/mail** client using **less** as a pager.

Profile (px/Px)

The child runs using its own profile, which must be loaded into the kernel. If the profile is not present, attempts to execute the child fail with permission denied. This is most useful if the parent program is invoking a global service, such as DNS lookups or sending mail with your system's MTA.

Choose the *profile with clean exec* (Px) option to scrub the environment of environment variables that could modify execution behavior when passed to the child process.

Child (cx/Cx)

Sets up a transition to a subprofile. It is like px/Px transition, except to a child profile.

Choose the *profile with clean exec* (Cx) option to scrub the environment of environment variables that could modify execution behavior when passed to the child process.

Unconfined (ux/Ux)

The child runs completely unconfined without any AppArmor profile applied to the executed resource.

Choose the *unconfined with clean exec* (Ux) option to scrub the environment of environment variables that could modify execution behavior when passed to the child process. Note that running unconfined profiles introduces a security vulnerability that could be used to evade AppArmor. Only use it as a last resort.

mmap (m)

This permission denotes that the program running under the profile can access the resource using the mmap system call with the flag `PROT_EXEC`. This means that the data mapped in it can be executed. You are prompted to include this permission if it is requested during a profiling run.

Deny

Adds a `deny` rule to the profile, and permanently prevents the program from accessing the specified directory path entries. AppArmor then continues to the next event.

Abort

Aborts **`aa-logprof`**, losing all rule changes entered so far and leaving all profiles unmodified.

Finish

Closes **`aa-logprof`**, saving all rule changes entered so far and modifying all profiles.

- *Example 25.2, "Learning Mode Exception: Defining Permissions for an Entry"* shows AppArmor suggest allowing a globbing pattern `/var/run/nscd/*` for reading, then using an abstraction to cover common Apache-related access rules.

EXAMPLE 25.2: LEARNING MODE EXCEPTION: DEFINING PERMISSIONS FOR AN ENTRY

```
Profile:  /usr/sbin/httpd2-prefork
Path:     /var/run/nscd/dbSz9CTr
Mode:     r
Severity: 3

  1 - /var/run/nscd/dbSz9CTr
 [2 - /var/run/nscd/*]

(A)llow / [(D)eny] / (G)lob / Glob w/(E)xt / (N)ew / Abo(r)t / (F)inish /
 (O)pts
Adding /var/run/nscd/* r to profile.
```

```
Profile:  /usr/sbin/httpd2-prefork
Path:     /proc/11769/attr/current
Mode:     w
Severity: 9

 [1 - #include <abstractions/apache2-common>]
  2 - /proc/11769/attr/current
  3 - /proc/*/attr/current

(A)llow / [(D)eny] / (G)lob / Glob w/(E)xt / (N)ew / Abo(r)t / (F)inish /
 (O)pts
Adding #include <abstractions/apache2-common> to profile.
```

AppArmor provides one or more paths or includes. By entering the option number, select the desired options then proceed to the next step.

Note

Not all of these options are always presented in the AppArmor menu.

`#include`

This is the section of an AppArmor profile that refers to an include file, which procures access permissions for programs. By using an include, you can give the program access to directory paths or files that are also required by other programs. Using includes can reduce the size of a profile. It is good practice to select includes when suggested.

Globbed Version

This is accessed by selecting *Glob* as described in the next step. For information about globbing syntax, refer to *Section 22.6, “Profile Names, Flags, Paths, and Globbing”*.

Actual Path

This is the literal path to which the program needs access so that it can run properly.

After you select the path or include, process it as an entry into the AppArmor profile by selecting *Allow* or *Deny*. If you are not satisfied with the directory path entry as it is displayed, you can also *Glob* it.

The following options are available to process the learning mode entries and build the profile:

Select `Enter`

Allows access to the selected directory path.

Allow

Allows access to the specified directory path entries. AppArmor suggests file permission access. For more information, refer to *Section 22.7, "File Permission Access Modes"*.

Deny

Prevents the program from accessing the specified directory path entries. AppArmor then continues to the next event.

New

Prompts you to enter your own rule for this event, allowing you to specify a regular expression. If the expression does not actually satisfy the event that prompted the question in the first place, AppArmor asks for confirmation and lets you reenter the expression.

Glob

Select a specific path or create a general rule using wild cards that match a broader set of paths. To select any of the offered paths, enter the number that is printed in front of the path then decide how to proceed with the selected item. For more information about globbing syntax, refer to *Section 22.6, "Profile Names, Flags, Paths, and Globbing"*.

Glob w/Ext

This modifies the original directory path while retaining the file name extension. For example, `/etc/apache2/file.ext` becomes `/etc/apache2/*.ext`, adding the wild card (asterisk) in place of the file name. This allows the program to access all files in the suggested directory that end with the `.ext` extension.

Abort

Aborts **`aa-logprof`**, losing all rule changes entered so far and leaving all profiles unmodified.

Finish

Closes **aa-logprof**, saving all rule changes entered so far and modifying all profiles.

6. To view and edit your profile using **vi**, enter **vi /etc/apparmor.d/** *PROFILENAME* in a terminal window. To enable syntax highlighting when editing an AppArmor profile in vim, use the commands **:syntax on** then **:set syntax=apparmor**. For more information about vim and syntax highlighting, refer to *Section 25.7.3.14, "apparmor.vim"*.

7. Restart AppArmor and reload the profile set including the newly created one using the **systemctl reload apparmor** command.

Like the graphical front-end for building AppArmor profiles, the YaST Add Profile Wizard, **aa-genprof** also supports the use of the local profile repository under `/etc/apparmor/profiles/extras` and the remote AppArmor profile repository.

To use a profile from the local repository, proceed as follows:

1. Start **aa-genprof** as described above.
   If **aa-genprof** finds an inactive local profile, the following lines appear on your terminal window:

```
Profile: /usr/bin/opera

 [1 - Inactive local profile for /usr/bin/opera]

[(V)iew Profile] / (U)se Profile / (C)reate New Profile / Abo(r)t / (F)inish
```

2. To use this profile, press U (*Use Profile*) and follow the profile generation procedure outlined above.
   To examine the profile before activating it, press V (*View Profile*).
   To ignore the existing profile, press C (*Create New Profile*) and follow the profile generation procedure outlined above to create the profile from scratch.

3. Leave **aa-genprof** by pressing F (*Finish*) when you are done and save your changes.

25.7.3.9 aa-logprof—Scanning the System Log

aa-logprof is an interactive tool used to review the complain and enforce mode events found in the log entries in `/var/log/audit/audit.log`, or directly in the `systemd` journal (see *Book "Reference", Chapter 11 "***journalctl***: Query the* `systemd` *Journal"*), and generate new entries in AppArmor security profiles.

When you run **aa-logprof**, it begins to scan the log files produced in complain and enforce mode and, if there are new security events that are not covered by the existing profile set, it gives suggestions for modifying the profile. **aa-logprof** uses this information to observe program behavior.

If a confined program forks and executes another program, **aa-logprof** sees this and asks the user which execution mode should be used when launching the child process. The execution modes *ix*, *px*, *Px*, *ux*, *Ux*, *cx*, *Cx*, and named profiles, are options for starting the child process. If a separate profile exists for the child process, the default selection is *Px*. If one does not exist, the profile defaults to *ix*. Child processes with separate profiles have **aa-autodep** run on them and are loaded into AppArmor, if it is running.

When **aa-logprof** exits, profiles are updated with the changes. If AppArmor is active, the updated profiles are reloaded and, if any processes that generated security events are still running in the null-XXXX profiles (unique profiles temporarily created in complain mode), those processes are set to run under their proper profiles.

To run **aa-logprof**, enter **aa-logprof** into a terminal window while logged in as `root`. The following options can be used for **aa-logprof**:

aa-logprof -d */path/to/profile/directory/*

Specifies the full path to the location of the profiles if the profiles are not located in the standard directory, `/etc/apparmor.d/`.

aa-logprof -f */path/to/logfile/*

Specifies the full path to the location of the log file if the log file is not located in the default directory or `/var/log/audit/audit.log`.

aa-logprof -m "string marker in logfile"

Marks the starting point for **aa-logprof** to look in the system log. **aa-logprof** ignores all events in the system log before the specified mark. If the mark contains spaces, it must be surrounded by quotes to work correctly. For example:

```
root # aa-logprof -m "17:04:21"
```

or

```
root # aa-logprof -m e2ff78636296f16d0b5301209a04430d
```

aa-logprof scans the log, asking you how to handle each logged event. Each question presents a numbered list of AppArmor rules that can be added by pressing the number of the item on the list.

By default, **aa-logprof** looks for profiles in `/etc/apparmor.d/`. Often running **aa-logprof** as `root` is enough to update the profile. However, there might be times when you need to search archived log files, such as if the program exercise period exceeds the log rotation window (when the log file is archived and a new log file is started). If this is the case, you can enter **zcat -f `ls -1tr** */path/to/logfile** ` | aa-logprof -f -.

25.7.3.10 aa-logprof Example 1

The following is an example of how **aa-logprof** addresses httpd2-prefork accessing the file `/etc/group`. `[]` indicates the default option.

In this example, the access to **/etc/group** is part of httpd2-prefork accessing name services. The appropriate response is `1`, which includes a predefined set of AppArmor rules. Selecting `1` to `#include` the name service package resolves all of the future questions pertaining to DNS lookups and makes the profile less brittle in that any changes to DNS configuration and the associated name service profile package can be made once, rather than needing to revise many profiles.

```
Profile:  /usr/sbin/httpd2-prefork
Path:     /etc/group
New Mode: r

[1 - #include <abstractions/nameservice>]
 2 - /etc/group
[(A)llow] / (D)eny / (N)ew / (G)lob / Glob w/(E)xt / Abo(r)t / (F)inish
```

Select one of the following responses:

Select `Enter`

Triggers the default action, which is, in this example, allowing access to the specified directory path entry.

Allow

Allows access to the specified directory path entries. AppArmor suggests file permission access. For more information about this, refer to *Section 22.7, "File Permission Access Modes"*.

Deny

Permanently prevents the program from accessing the specified directory path entries. AppArmor then continues to the next event.

New

Prompts you to enter your own rule for this event, allowing you to specify whatever form of regular expression you want. If the expression entered does not actually satisfy the event that prompted the question in the first place, AppArmor asks for confirmation and lets you reenter the expression.

Glob

Select either a specific path or create a general rule using wild cards that matches on a broader set of paths. To select any of the offered paths, enter the number that is printed in front of the paths then decide how to proceed with the selected item.

For more information about globbing syntax, refer to *Section 22.6, "Profile Names, Flags, Paths, and Globbing"*.

Glob w/Ext

This modifies the original directory path while retaining the file name extension. For example, `/etc/apache2/file.ext` becomes `/etc/apache2/*.ext`, adding the wild card (asterisk) in place of the file name. This allows the program to access all files in the suggested directory that end with the `.ext` extension.

Abort

Aborts **`aa-logprof`**, losing all rule changes entered so far and leaving all profiles unmodified.

Finish

Closes **`aa-logprof`**, saving all rule changes entered so far and modifying all profiles.

25.7.3.11 aa-logprof Example 2

For example, when profiling vsftpd, see this question:

```
Profile:  /usr/sbin/vsftpd
Path:     /y2k.jpg

New Mode: r

[1 - /y2k.jpg]

(A)llow / [(D)eny] / (N)ew / (G)lob / Glob w/(E)xt / Abo(r)t / (F)inish
```

Several items of interest appear in this question. First, note that vsftpd is asking for a path entry at the top of the tree, even though vsftpd on openSUSE Leap serves FTP files from `/srv/ftp` by default. This is because vsftpd uses chroot and, for the portion of the code inside the chroot jail, AppArmor sees file accesses in terms of the chroot environment rather than the global absolute path.

The second item of interest is that you should grant FTP read access to all JPEG files in the directory, so you could use *Glob w/Ext* and use the suggested path of `/*.jpg`. Doing so collapses all previous rules granting access to individual `.jpg` files and forestalls any future questions pertaining to access to `.jpg` files.

Finally, you should grant more general access to FTP files. If you select *Glob* in the last entry, **`aa-logprof`** replaces the suggested path of `/y2k.jpg` with `/*`. Alternatively, you should grant even more access to the entire directory tree, in which case you could use the *New* path option and enter `/**.jpg` (which would grant access to all `.jpg` files in the entire directory tree) or `/**` (which would grant access to all files in the directory tree).

These items deal with read accesses. Write accesses are similar, except that it is good policy to be more conservative in your use of regular expressions for write accesses. Dealing with execute accesses is more complex. Find an example in *Example 25.1, "Learning Mode Exception: Controlling Access to Specific Resources"*.

In the following example, the `/usr/bin/mail` mail client is being profiled and **`aa-logprof`** has discovered that **`/usr/bin/mail`** executes **`/usr/bin/less`** as a helper application to "page" long mail messages. Consequently, it presents this prompt:

```
/usr/bin/nail -> /usr/bin/less
(I)nherit / (P)rofile / (C)hild / (N)ame / (U)nconfined / (X)ix / (D)eny
```

Note

The actual executable file for `/usr/bin/mail` turns out to be `/usr/bin/nail`, which is not a typographical error.

The program `/usr/bin/less` appears to be a simple one for scrolling through text that is more than one screen long and that is in fact what `/usr/bin/mail` is using it for. However, **`less`** is actually a large and powerful program that uses many other helper applications, such as **`tar`** and **`rpm`**.

Tip

Run `less` on a tar file or an RPM file and it shows you the inventory of these containers.

You do not want to run `rpm` automatically when reading mail messages (that leads directly to a Microsoft* Outlook–style virus attack, because RPM has the power to install and modify system programs), so, in this case, the best choice is to use *Inherit*. This results in the less program executed from this context running under the profile for `/usr/bin/mail`. This has two consequences:

- You need to add all of the basic file accesses for `/usr/bin/less` to the profile for `/usr/bin/mail`.
- You can avoid adding the helper applications, such as `tar` and `rpm`, to the `/usr/bin/mail` profile so that when `/usr/bin/mail` runs `/usr/bin/less` in this context, the less program is far less dangerous than it would be without AppArmor protection. Another option is to use the Cx execute modes. For more information on execute modes, see *Section 22.12, “Execute Modes”*.

In other circumstances, you might instead want to use the *Profile* option. This has the following effects on `aa-logprof`:

- The rule written into the profile uses px/Px, which forces the transition to the child's own profile.
- `aa-logprof` constructs a profile for the child and starts building it, in the same way that it built the parent profile, by assigning events for the child process to the child's profile and asking the `aa-logprof` user questions. The profile will also be applied if you run the child as a stand-alone program.

If a confined program forks and executes another program, `aa-logprof` sees this and asks the user which execution mode should be used when launching the child process. The execution modes of inherit, profile, unconfined, child, named profile, or an option to deny the execution are presented.

If a separate profile exists for the child process, the default selection is profile. If a profile does not exist, the default is inherit. The inherit option, or `ix`, is described in *Section 22.7, “File Permission Access Modes”*.

The profile option indicates that the child program should run in its own profile. A secondary question asks whether to sanitize the environment that the child program inherits from the parent. If you choose to sanitize the environment, this places the execution modifier `Px` in your AppArmor profile. If you select not to sanitize, `px` is placed in the profile and no environment sanitizing occurs. The default for the execution mode is `Px` if you select profile execution mode.

The unconfined execution mode is not recommended and should only be used in cases where there is no other option to generate a profile for a program reliably. Selecting unconfined opens a warning dialog asking for confirmation of the choice. If you are sure and choose *Yes*, a second dialog ask whether to sanitize the environment. To use the execution mode `Ux` in your profile, select *Yes*. To use the execution mode `ux` in your profile instead, select *No*. The default value selected is `Ux` for unconfined execution mode.

Important: Running Unconfined

Selecting `ux or Ux` is very dangerous and provides no enforcement of policy (from a security perspective) of the resulting execution behavior of the child program.

25.7.3.12 aa-unconfined—Identifying Unprotected Processes

The **`aa-unconfined`** command examines open network ports on your system, compares that to the set of profiles loaded on your system, and reports network services that do not have AppArmor profiles. It requires `root` privileges and that it not be confined by an AppArmor profile.

`aa-unconfined` must be run as `root` to retrieve the process executable link from the `/proc` file system. This program is susceptible to the following race conditions:

- An unlinked executable is mishandled
- A process that dies between **`netstat(8)`** and further checks is mishandled

Note

This program lists processes using TCP and UDP only. In short, this program is unsuitable for forensics use and is provided only as an aid to profiling all network-accessible processes in the lab.

25.7.3.13 aa-notify

aa-notify is a handy utility that displays AppArmor notifications in your desktop environment. This is very convenient if you do not want to inspect the AppArmor log file, but rather let the desktop inform you about events that violate the policy. To enable AppArmor desktop notifications, run **aa-notify**:

```
tux > sudo aa-notify -p -u USERNAME --display DISPLAY_NUMBER
```

where *USERNAME* is your user name under which you are logged in, and *DISPLAY_NUMBER* is the X Window display number you are currently using, such as `:0`. The process is run in the background, and shows a notification each time a deny event happens.

Tip

The active X Window display number is saved in the `$DISPLAY` variable, so you can use `--display $DISPLAY` to avoid finding out the current display number.

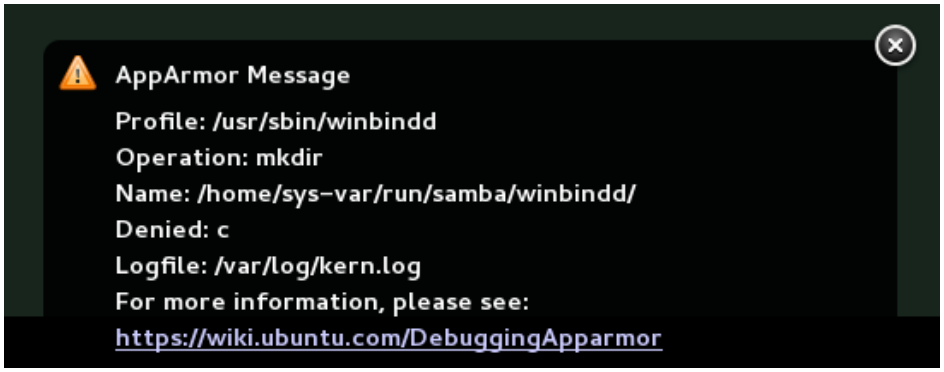


FIGURE 25.1: **aa-notify Message in GNOME**

With the `-s DAYS` option, you can also configure **aa-notify** to display a summary of notifications for the specified number of past days. For more information on **aa-notify**, see its man page **man 8 aa-notify**.

25.7.3.14 apparmor.vim

A syntax highlighting file for the vim text editor highlights various features of an AppArmor profile with colors. Using vim and the AppArmor syntax mode for vim, you can see the semantic implications of your profiles with color highlighting. Use vim to view and edit your profile by typing vim at a terminal window.

To enable the syntax coloring when you edit an AppArmor profile in vim, use the commands `:syntax on` then `:set syntax=apparmor`. To make sure vim recognizes the edited file type correctly as an AppArmor profile, add

```
# vim:ft=apparmor
```

at the end of the profile.

Tip

vim comes with AppArmor highlighting automatically enabled for files in `/etc/apparmor.d/`.

When you enable this feature, vim colors the lines of the profile for you:

Blue
: Comments

White
: Ordinary read access lines

Brown
: Capability statements and complain flags

Yellow
: Lines that grant write access

Green
: Lines that grant execute permission (either ix or px)

Red
: Lines that grant unconfined access (ux)

Red background
: Syntax errors that will not load properly into the AppArmor modules

Use the `apparmor.vim` and `vim` man pages and the `:help syntax` from within the vim editor for further vim help about syntax highlighting. The AppArmor syntax is stored in `/usr/share/vim/current/syntax/apparmor.vim`.

25.8 Important File Names and Directories

The following list contains the most important files and directories used by the AppArmor framework. If you intend to manage and troubleshoot your profiles manually, make sure that you know about these files and directories:

`/sys/kernel/security/apparmor/profiles`

Virtualized file representing the currently loaded set of profiles.

`/etc/apparmor/`

Location of AppArmor configuration files.

`/etc/apparmor/profiles/extras/`

A local repository of profiles shipped with AppArmor, but not enabled by default.

`/etc/apparmor.d/`

Location of profiles, named with the convention of replacing the `/` in paths with `.` (not for the root `/`) so profiles are easier to manage. For example, the profile for the program `/usr/sbin/smbd` is named `usr.sbin.smbd`.

`/etc/apparmor.d/abstractions/`

Location of abstractions.

`/etc/apparmor.d/program-chunks/`

Location of program chunks.

`/proc/*/attr/current`

Check this file to review the confinement status of a process and the profile that is used to confine the process. The **`ps`** `auxZ` command retrieves this information automatically.

26 Profiling Your Web Applications Using ChangeHat

An AppArmor® profile represents the security policy for an individual program instance or process. It applies to an executable program, but if a portion of the program needs different access permissions than other portions, the program can "change hats" to use a different security context, distinctive from the access of the main program. This is known as a *hat* or *subprofile*.

ChangeHat enables programs to change to or from a *hat* within an AppArmor profile. It enables you to define security at a finer level than the process. This feature requires that each application be made "ChangeHat-aware", meaning that it is modified to make a request to the AppArmor module to switch security domains at specific times during the application execution. One example of a ChangeHat-aware application is the Apache Web server.

A profile can have an arbitrary number of subprofiles, but there are only two levels: a subprofile cannot have further child profiles. A subprofile is written as a separate profile. Its name consists of the name of the containing profile followed by the subprofile name, separated by a `^`.

Subprofiles are either stored in the same file as the parent profile, or in a separate file. The latter case is recommended on sites with many hats—it allows the policy caching to handle changes at the per hat level. If all the hats are in the same file as the parent profile, then the parent profile and all hats must be recompiled.

An external subprofile that is going to be used as a hat, must begin with the word `hat` or the `^` character.

The following two subprofiles *cannot* be used as a hat:

```
/foo//bar { }
```

or

```
profile /foo//bar { }
```

While the following two are treated as hats:

```
^/foo//bar { }
```

or

```
hat /foo//bar { } # this syntax is not highlighted in vim
```

Note that the security of hats is considerably weaker than that of full profiles. Using certain types of bugs in a program, an attacker may be able to escape from a hat into the containing profile. This is because the security of hats is determined by a secret key handled by the containing

process, and the code running in the hat must not have access to the key. Thus, change_hat is most useful with application servers, where a language interpreter (such as PERL, PHP, or Java) is isolating pieces of code such that they do not have direct access to the memory of the containing process.

The rest of this chapter describes using change_hat with Apache, to contain Web server components run using `mod_perl` and `mod_php`. Similar approaches can be used with any application server by providing an application module similar to the `mod_apparmor` described next in *Section 26.1.2, "Location and Directory Directives"*.

Tip: For More Information

For more information, see the **`change_hat`** man page.

26.1 Configuring Apache for `mod_apparmor`

AppArmor provides a `mod_apparmor` module (package `apache2-mod-apparmor`) for the Apache program. This module makes the Apache Web server ChangeHat aware. Install it along with Apache.

When Apache is ChangeHat-aware, it checks for the following customized AppArmor security profiles in the order given for every URI request that it receives.

- URI-specific hat. For example, `^www_app_name/templates/classic/images/bar_left.gif`
- `DEFAULT_URI`
- `HANDLING_UNTRUSTED_INPUT`

Note: Apache Configuration

If you install `apache2-mod-apparmor`, make sure the module is enabled, and then restart Apache by executing the following command:

```
tux > a2enmod apparmor && sudo systemctl reload apache2
```

Apache is configured by placing directives in plain text configuration files. The main configuration file is usually `/etc/apache2/httpd.conf`. When you compile Apache, you can indicate the location of this file. Directives can be placed in any of these configuration files to alter the way Apache behaves. When you make changes to the main configuration files, you need to reload Apache with **`sudo systemctl reload apache2`**, so the changes are recognized.

26.1.1 Virtual Host Directives

<VirtualHost> and </VirtualHost> directives are used to enclose a group of directives that will apply only to a particular virtual host. For more information on Apache virtual host directives, refer to http://httpd.apache.org/docs/2.4/en/mod/core.html#virtualhost.

The ChangeHat-specific configuration keyword is `AADefaultHatName`. It is used similarly to `AAHatName`, for example, `AADefaultHatName My_Funky_Default_Hat`.

It allows you to specify a default hat to be used for virtual hosts and other Apache server directives, so that you can have different defaults for different virtual hosts. This can be overridden by the `AAHatName` directive and is checked for only if there is not a matching `AAHatName` or hat named by the URI. If the `AADefaultHatName` hat does not exist, it falls back to the `DEFAULT_URI` hat if it exists/

If none of those are matched, it goes back to the "parent" Apache hat.

26.1.2 Location and Directory Directives

Location and directory directives specify hat names in the program configuration file so the Apache calls the hat regarding its security. For Apache, you can find documentation about the location and directory directives at http://httpd.apache.org/docs/2.4/en/sections.html.

The location directive example below specifies that, for a given location, `mod_apparmor` should use a specific hat:

```
<Location /foo/>
  AAHatName MY_HAT_NAME
</Location>
```

This tries to use `MY_HAT_NAME` for any URI beginning with `/foo/` (`/foo/`, `/foo/bar`, `/foo/cgi/path/blah_blah/blah`, etc.).

The directory directive works similarly to the location directive, except it refers to a path in the file system as in the following example:

```
<Directory "/srv/www/www.example.org/docs">
  # Note lack of trailing slash
  AAHatName example.org
</Directory>
```

26.2 Managing ChangeHat-Aware Applications

In the previous section you learned about `mod_apparmor` and the way it helps you to secure a specific Web application. This section walks you through a real-life example of creating a hat for a Web application, and using AppArmor's change_hat feature to secure it. Note that this chapter focuses on AppArmor's command line tools, as YaST's AppArmor module has limited functionality.

26.2.1 With AppArmor's Command Line Tools

For illustration purposes, let us choose the Web application called *Adminer* (http://www.adminer.org/en/). It is a full-featured SQL database management tool written in PHP, yet consisting of a single PHP file. For Adminer to work, you need to set up an Apache Web server, PHP and its Apache module, and one of the database drivers available for PHP—MariaDB in this example. You can install the required packages with

```
zypper in apache2 apache2-mod_apparmor apache2-mod_php5 php5 php5-mysql
```

To set up the Web environment for running Adminer, follow these steps:

PROCEDURE 26.1: SETTING UP A WEB SERVER ENVIRONMENT

1. Make sure `apparmor` and `php5` modules are enabled for Apache. To enable the modules in any case, use:

   ```
   tux > a2enmod apparmor php5
   ```

   and then restart Apache with

   ```
   tux > sudo systemctl restart apache2
   ```

2. Make sure MariaDB is running. If unsure, restart it with

```
tux > sudo systemctl restart mysql
```

3. Download Adminer from http://www.adminer.org, copy it to `/srv/www/htdocs/adminer/`, and rename it to `adminer.php`, so that its full path is `/srv/www/htdocs/adminer/adminer.php`.

4. Test Adminer in your Web browser by entering `http://localhost/adminer/adminer.php` in its URI address field. If you installed Adminer to a remote server, replace `localhost` with the real host name of the server.

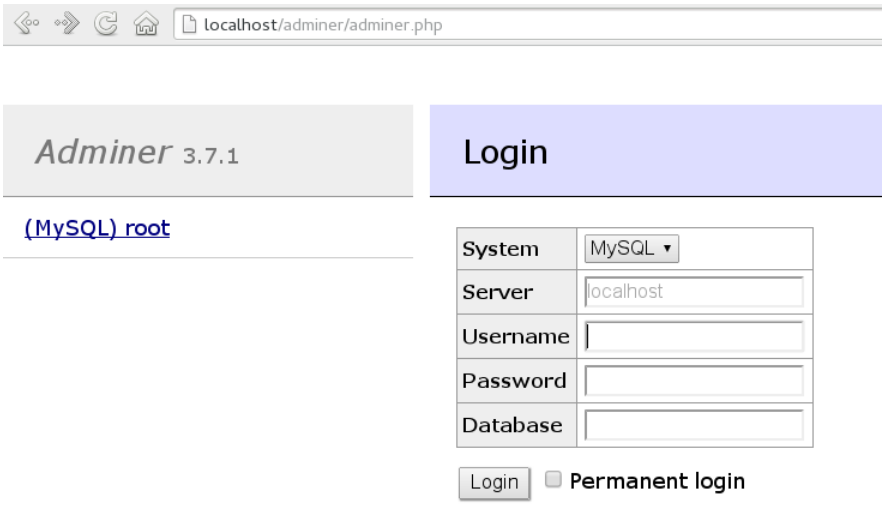


FIGURE 26.1: ADMINER LOGIN PAGE

Tip

If you encounter problems viewing the Adminer login page, try to look for help in the Apache error log `/var/log/apache2/error.log`. Another reason you cannot access the Web page may be that your Apache is already under AppArmor control and its AppArmor profile is too tight to permit viewing Adminer. Check it with **`aa-status`**, and if needed, set Apache temporarily in complain mode with

```
root # sudo aa-complain usr.sbin.httpd2-prefork
```

After the Web environment for Adminer is ready, you need to configure Apache's `mod_apparmor`, so that AppArmor can detect accesses to Adminer and change to the specific "hat".

PROCEDURE 26.2: CONFIGURING `mod_apparmor`

1. Apache has several configuration files under `/etc/apache2/` and `/etc/apache2/conf.d/`. Choose your preferred one and open it in a text editor. In this example, the **`vim`** editor is used to create a new configuration file `/etc/apache2/conf.d/apparmor.conf`.

   ```
   tux > sudo vim /etc/apache2/conf.d/apparmor.conf
   ```

2. Copy the following snippet into the edited file.

   ```
   <Directory /srv/www/htdocs/adminer>
     AAHatName adminer
   </Directory>
   ```

   It tells Apache to let AppArmor know about a change_hat event when the Web user accesses the directory `/adminer` (and any file/directory inside) in Apache's document root. Remember, we placed the `adminer.php` application there.

3. Save the file, close the editor, and restart Apache with

   ```
   tux > sudo systemctl restart apache2
   ```

Apache now knows about our Adminer and changing a "hat" for it. It is time to create the related hat for Adminer in the AppArmor configuration. If you do not have an AppArmor profile yet, create one before proceeding. Remember that if your Apache's main binary is `/usr/sbin/httpd2-prefork`, then the related profile is named `/etc/apparmor.d/usr.sbin.httpd2-prefork`.

PROCEDURE 26.3: CREATING A HAT FOR ADMINER

1. Open (or create one if it does not exist) the file `/etc/apparmor.d/usr.sbin.httpd2-prefork` in a text editor. Its contents should be similar to the following:

   ```
   #include <tunables/global>

   /usr/sbin/httpd2-prefork {
     #include <abstractions/apache2-common>
     #include <abstractions/base>
     #include <abstractions/php5>
   ```

```
  capability kill,
  capability setgid,
  capability setuid,

  /etc/apache2/** r,
  /run/httpd.pid rw,
  /usr/lib{,32,64}/apache2*/** mr,
  /var/log/apache2/** rw,

  ^DEFAULT_URI {
    #include <abstractions/apache2-common>
    /var/log/apache2/** rw,
  }

  ^HANDLING_UNTRUSTED_INPUT {
    #include <abstractions/apache2-common>
    /var/log/apache2/** w,
  }
}
```

2. Before the last closing curly bracket (`}`), insert the following section:

```
^adminer flags=(complain) {
}
```

   Note the `(complain)` addition after the hat name—it tells AppArmor to leave the `adminer` hat in complain mode. That is because we need to learn the hat profile by accessing Adminer later on.

3. Save the file, and then restart AppArmor, then Apache.

```
tux > sudo systemctl reload apparmor apache2
```

4. Check if the `adminer` hat really is in complain mode.

```
tux > sudo aa-status
apparmor module is loaded.
39 profiles are loaded.
37 profiles are in enforce mode.
[...]
   /usr/sbin/httpd2-prefork
   /usr/sbin/httpd2-prefork//DEFAULT_URI
   /usr/sbin/httpd2-prefork//HANDLING_UNTRUSTED_INPUT
[...]
2 profiles are in complain mode.
```

```
   /usr/bin/getopt
   /usr/sbin/httpd2-prefork//adminer
[...]
```

As we can see, the `httpd2-prefork//adminer` is loaded in complain mode.

Our last task is to find out the right set of rules for the `adminer` hat. That is why we set the `adminer` hat into complain mode—the logging facility collects useful information about the access requirements of `adminer.php` as we use it via the Web browser. **`aa-logprof`** then helps us with creating the hat's profile.

PROCEDURE 26.4: GENERATING RULES FOR THE `adminer` HAT

1. Open Adminer in the Web browser. If you installed it locally, then the URI is `http://localhost/adminer/adminer.php`.
2. Choose the database engine you want to use (MariaDB in our case), and log in to Adminer using the existing database user name and password. You do not need to specify the database name as you can do so after logging in. Perform any operations with Adminer you like—create a new database, create a new table for it, set user privileges, and so on.
3. After the short testing of Adminer's user interface, switch back to console and examine the log for collected data.

```
tux > sudo aa-logprof
Reading log entries from /var/log/audit/audit.log.
Updating AppArmor profiles in /etc/apparmor.d.
Complain-mode changes:

Profile:  /usr/sbin/httpd2-prefork^adminer
Path:     /dev/urandom
Mode:     r
Severity: 3

  1 - #include <abstractions/apache2-common>
[...]
 [8 - /dev/urandom]

[(A)llow] / (D)eny / (G)lob / Glob w/(E)xt / (N)ew / Abo(r)t / (F)inish / (O)pts
```

From the **`aa-logprof`** message, it is clear that our new `adminer` hat was correctly detected:

```
Profile:  /usr/sbin/httpd2-prefork^adminer
```

The **aa-logprof** command will ask you to pick the right rule for each discovered AppArmor event. Specify the one you want to use, and confirm with *Allow*. For more information on working with the **aa-genprof** and **aa-logprof** interface, see *Section 25.7.3.8, "aa-genprof—Generating Profiles"*.

Tip

aa-logprof usually offers several valid rules for the examined event. Some are *abstractions*—predefined sets of rules affecting a specific common group of targets. Sometimes it is useful to include such an abstraction instead of a direct URI rule:

```
1 - #include <abstractions/php5>
[2 - /var/lib/php5/sess_3jdmii9cacj1e3jnahbtopajl7p064ai242]
```

In the example above, it is recommended hitting *1* and confirming with *A* to allow the abstraction.

4. After the last change, you will be asked to save the changed profile.

```
The following local profiles were changed. Would you like to save them?
 [1 - /usr/sbin/httpd2-prefork]

 (S)ave Changes / [(V)iew Changes] / Abo(r)t
```

Hit *S* to save the changes.

5. Set the profile to enforce mode with **aa-enforce**

```
tux > sudo aa-enforce usr.sbin.httpd2-prefork
```

and check its status with **aa-status**

```
tux > sudo aa-status
apparmor module is loaded.
39 profiles are loaded.
38 profiles are in enforce mode.
[...]
   /usr/sbin/httpd2-prefork
   /usr/sbin/httpd2-prefork//DEFAULT_URI
   /usr/sbin/httpd2-prefork//HANDLING_UNTRUSTED_INPUT
   /usr/sbin/httpd2-prefork//adminer
[...]
```

As you can see, the `//adminer` hat jumped from *complain* to *enforce* mode.

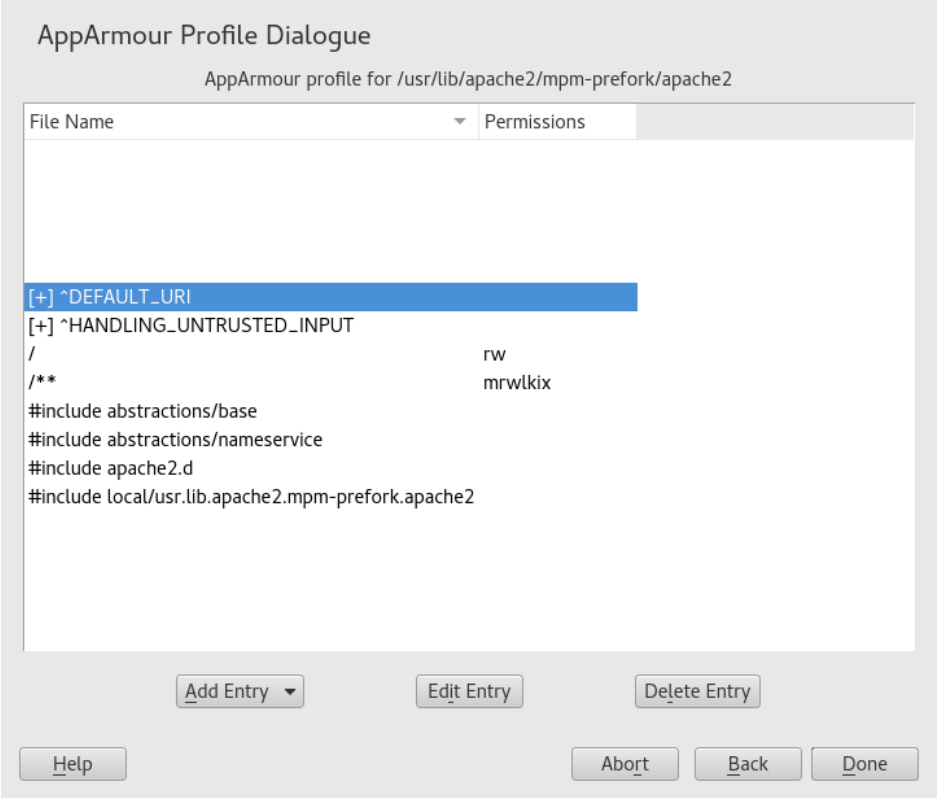
6. Try to run Adminer in the Web browser, and if you encounter problems running it, switch it to the complain mode, repeat the steps that previously did not work well, and update the profile with **`aa-logprof`** until you are satisfied with the application's functionality.

Note: Hat and Parent Profile Relationship

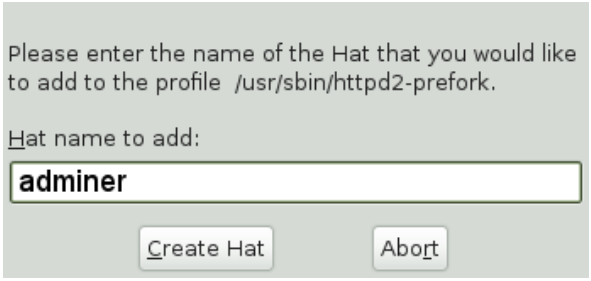
The profile `^adminer` is only available in the context of a process running under the parent profile `usr.sbin.httpd2-prefork`.

26.2.2 Adding Hats and Entries to Hats in YaST

When you use the *Edit Profile* dialog (for instructions, refer to *Section 24.2, "Editing Profiles"*) or when you add a new profile using *Manually Add Profile* (for instructions, refer to *Section 24.1, "Manually Adding a Profile"*), you are given the option of adding hats (subprofiles) to your AppArmor profiles. Add a ChangeHat subprofile from the *AppArmor Profile Dialog* window as in the following.



1. From the *AppArmor Profile Dialog* window, click *Add Entry* then select *Hat*. The *Enter Hat Name* dialog opens:



2. Enter the name of the hat to add to the AppArmor profile. The name is the URI that, when accessed, receives the permissions set in the hat.
3. Click *Create Hat*. You are returned to the *AppArmor Profile Dialog* screen.
4. After adding the new hat, click *Done*.

27 Confining Users with pam_apparmor

An AppArmor profile applies to an executable program; if a portion of the program needs different access permissions than other portions need, the program can change hats via change_hat to a different role, also known as a subprofile. The `pam_apparmor` PAM module allows applications to confine authenticated users into subprofiles based on group names, user names, or a default profile. To accomplish this, `pam_apparmor` needs to be registered as a PAM session module.

The package `pam_apparmor` is not installed by default, you can install it using YaST or **`zypper`**. Details about how to set up and configure `pam_apparmor` can be found in `/usr/share/doc/packages/pam_apparmor/README` after the package has been installed. For details on PAM, refer to *Chapter 2, Authentication with PAM*.

28 Managing Profiled Applications

After creating profiles and immunizing your applications, openSUSE® Leap becomes more efficient and better protected as long as you perform AppArmor® profile maintenance (which involves analyzing log files, refining your profiles, backing up your set of profiles and keeping it up-to-date). You can deal with these issues before they become a problem by setting up event notification by e-mail, updating profiles from system log entries by running the aa-logprof tool, and dealing with maintenance issues.

28.1 Reacting to Security Event Rejections

When you receive a security event rejection, examine the access violation and determine if that event indicated a threat or was part of normal application behavior. Application-specific knowledge is required to make the determination. If the rejected action is part of normal application behavior, run **`aa-logprof`** at the command line.

If the rejected action is not part of normal application behavior, this access should be considered a possible intrusion attempt (that was prevented) and this notification should be passed to the person responsible for security within your organization.

28.2 Maintaining Your Security Profiles

In a production environment, you should plan on maintaining profiles for all of the deployed applications. The security policies are an integral part of your deployment. You should plan on taking steps to back up and restore security policy files, plan for software changes, and allow any needed modification of security policies that your environment dictates.

28.2.1 Backing Up Your Security Profiles

Backing up profiles might save you from having to re-profile all your programs after a disk crash. Also, if profiles are changed, you can easily restore previous settings by using the backed up files.

Back up profiles by copying the profile files to a specified directory.

1. You should first archive the files into one file. To do this, open a terminal window and enter the following as `root`:

   ```
   tux > sudo tar zclpf profiles.tgz /etc/apparmor.d
   ```

   The simplest method to ensure that your security policy files are regularly backed up is to include the directory `/etc/apparmor.d` in the list of directories that your backup system archives.

2. You can also use **`scp`** or a file manager like Nautilus to store the files on some kind of storage media, the network, or another computer.

28.2.2 Changing Your Security Profiles

Maintenance of security profiles includes changing them if you decide that your system requires more or less security for its applications. To change your profiles in AppArmor, refer to *Section 24.2, "Editing Profiles"*.

28.2.3 Introducing New Software into Your Environment

When you add a new application version or patch to your system, you should always update the profile to fit your needs. You have several options, depending on your company's software deployment strategy. You can deploy your patches and upgrades into a test or production environment. The following explains how to do this with each method.

If you intend to deploy a patch or upgrade in a test environment, the best method for updating your profiles is to run **`aa-logprof`** in a terminal as `root`. For detailed instructions, refer to *Section 25.7.3.9, "aa-logprof—Scanning the System Log"*.

If you intend to deploy a patch or upgrade directly into a production environment, the best method for updating your profiles is to monitor the system frequently to determine if any new rejections should be added to the profile and update as needed using **`aa-logprof`**. For detailed instructions, refer to *Section 25.7.3.9, "aa-logprof—Scanning the System Log"*.

29 Support

This chapter outlines maintenance-related tasks. Learn how to update AppArmor® and get a list of available man pages providing basic help for using the command line tools provided by AppArmor. Use the troubleshooting section to learn about some common problems encountered with AppArmor and their solutions. Report defects or enhancement requests for AppArmor following the instructions in this chapter.

29.1 Updating AppArmor Online

Updates for AppArmor packages are provided in the same way as any other update for openSUSE Leap. Retrieve and apply them exactly like for any other package that ships as part of openSUSE Leap.

29.2 Using the Man Pages

There are man pages available for your use. In a terminal, enter `man apparmor` to open the AppArmor man page. Man pages are distributed in sections numbered 1 through 8. Each section is specific to a category of documentation:

TABLE 29.1: MAN PAGES: SECTIONS AND CATEGORIES

Section	Category
1	User commands
2	System calls
3	Library functions
4	Device driver information
5	Configuration file formats
6	Games
7	High level concepts

Section	Category
8	Administrator commands

The section numbers are used to distinguish man pages from each other. For example, `exit(2)` describes the exit system call, while `exit(3)` describes the exit C library function.

The AppArmor man pages are:

- `aa-audit(8)`
- `aa-autodep(8)`
- `aa-complain(8)`
- `aa-decode(8)`
- `aa-disable(8)`
- `aa-easyprof(8)`
- `aa-enforce(8)`
- `aa-enxec(8)`
- `aa-genprof(8)`
- `aa-logprof(8)`
- `aa-notify(8)`
- `aa-status(8)`
- `aa-unconfined(8)`
- `aa_change_hat(8)`
- `logprof.conf(5)`
- `apparmor.d(5)`
- `apparmor.vim(5)`
- `apparmor(7)`
- `apparmor_parser(8)`
- `apparmor_status(8)`

29.3 For More Information

Find more information about the AppArmor product at: http://wiki.apparmor.net. Find the product documentation for AppArmor in the installed system at `/usr/share/doc/manual`.

There is a mailing list for AppArmor that users can post to or join to communicate with developers. See https://lists.ubuntu.com/mailman/listinfo/apparmor for details.

29.4 Troubleshooting

This section lists the most common problems and error messages that may occur using AppArmor.

29.4.1 How to React to odd Application Behavior?

If you notice odd application behavior or any other type of application problem, you should first check the reject messages in the log files to see if AppArmor is too closely constricting your application. If you detect reject messages that indicate that your application or service is too closely restricted by AppArmor, update your profile to properly handle your use case of the application. Do this with **`aa-logprof`** (*Section 25.7.3.9, "aa-logprof—Scanning the System Log"*).

If you decide to run your application or service without AppArmor protection, remove the application's profile from `/etc/apparmor.d` or move it to another location.

29.4.2 My Profiles Do not Seem to Work Anymore ...

If you have been using previous versions of AppArmor and have updated your system (but kept your old set of profiles) you might notice some applications which seemed to work perfectly before you updated behaving strangely, or not working.

This version of AppArmor introduces a set of new features to the profile syntax and the AppArmor tools that might cause trouble with older versions of the AppArmor profiles. Those features are:

- File Locking
- Network Access Control

- The `SYS_PTRACE` Capability
- Directory Path Access

The current version of AppArmor mediates file locking and introduces a new permission mode (`k`) for this. Applications requesting file locking permission might misbehave or fail altogether if confined by older profiles which do not explicitly contain permissions to lock files. If you suspect this being the case, check the log file under `/var/log/audit/audit.log` for entries like the following:

```
type=AVC msg=audit(1389862802.727:13939): apparmor="DENIED" \
operation="file_lock" parent=2692 profile="/usr/bin/opera" \
name="/home/tux/.qt/.qtrc.lock" pid=28730 comm="httpd2-prefork" \
requested_mask="::k" denied_mask="::k" fsuid=30 ouid=0
```

Update the profile using the **`aa-logprof`** command as outlined below.

The new network access control syntax based on the network family and type specification, described in *Section 22.5, "Network Access Control"*, might cause application misbehavior or even stop applications from working. If you notice a network-related application behaving strangely, check the log file under `/var/log/audit/audit.log` for entries like the following:

```
type=AVC msg=audit(1389864332.233:13947): apparmor="DENIED" \
operation="socket_create" family="inet" parent=29985 profile="/bin/ping" \
sock_type="raw" pid=30251 comm="ping"
```

This log entry means that our example application, **`/bin/ping`** in this case, failed to get AppArmor's permission to open a network connection. This permission needs to be explicitly stated to make sure that an application has network access. To update the profile to the new syntax, use the **`aa-logprof`** command as outlined below.

The current kernel requires the `SYS_PTRACE` capability, if a process tries to access files in `/proc/PID/fd/*`. New profiles need an entry for the file and the capability, where old profiles only needed the file entry. For example:

```
/proc/*/fd/**  rw,
```

in the old syntax would translate to the following rules in the new syntax:

```
capability SYS_PTRACE,
/proc/*/fd/**  rw,
```

To update the profile to the new syntax, use the YaST Update Profile Wizard or the **aa-logprof** command as outlined below.

With this version of AppArmor, a few changes have been made to the profile rule syntax to better distinguish directory from file access. Therefore, some rules matching both file and directory paths in the previous version might now match a file path only. This could lead to AppArmor not being able to access a crucial directory, and thus trigger misbehavior of your application and various log messages. The following examples highlight the most important changes to the path syntax.

Using the old syntax, the following rule would allow access to files and directories in `/proc/net`. It would allow directory access only to read the entries in the directory, but not give access to files or directories under the directory, for example `/proc/net/dir/foo` would be matched by the asterisk (*), but as `foo` is a file or directory under `dir`, it cannot be accessed.

```
/proc/net/*  r,
```

To get the same behavior using the new syntax, you need two rules instead of one. The first allows access to the file under `/proc/net` and the second allows access to directories under `/proc/net`. Directory access can only be used for listing the contents, not actually accessing files or directories underneath the directory.

```
/proc/net/*  r,
/proc/net/*/  r,
```

The following rule works similarly both under the old and the new syntax, and allows access to both files and directories under `/proc/net` (but does not allow a directory listing of `/proc/net/` itself):

```
/proc/net/**  r,
```

To distinguish file access from directory access using the above expression in the new syntax, use the following two rules. The first one only allows to recursively access directories under `/proc/net` while the second one explicitly allows for recursive file access only.

```
/proc/net/**/  r,
/proc/net/**[^/]  r,
```

The following rule works similarly both under the old and the new syntax and allows access to both files and directories beginning with `foo` under `/proc/net`:

```
/proc/net/foo**  r,
```

To distinguish file access from directory access in the new syntax and use the `**` globbing pattern, use the following two rules. The first one would have matched both files and directories in the old syntax, but only matches files in the new syntax because of the missing trailing slash. The second rule matched neither file nor directory in the old syntax, but matches directories only in the new syntax:

```
/proc/net/**foo  r,
/proc/net/**foo/  r,
```

The following rules illustrate how the use of the `?` globbing pattern has changed. In the old syntax, the first rule would have matched both files and directories (four characters, last character could be any but a slash). In the new syntax, it matches only files (trailing slash is missing). The second rule would match nothing in the old profile syntax, but matches directories only in the new syntax. The last rule matches explicitly matches a file called `bar` under `/proc/net/foo?`. Using the old syntax, this rule would have applied to both files and directories:

```
/proc/net/foo?  r,
/proc/net/foo?/  r,
/proc/net/foo?/bar  r,
```

To find and resolve issues related to syntax changes, take some time after the update to check the profiles you want to keep and proceed as follows for each application you kept the profile for:

1. Put the application's profile into complain mode:

   ```
   tux > sudo aa-complain /path/to/application
   ```

   Log entries are made for any actions violating the current profile, but the profile is not enforced and the application's behavior not restricted.

2. Run the application covering all the tasks you need this application to be able to perform.

3. Update the profile according to the log entries made while running the application:

   ```
   tux > sudo aa-logprof /path/to/application
   ```

4. Put the resulting profile back into enforce mode:

```
tux > sudo aa-enforce /path/to/application
```

29.4.3 Resolving Issues with Apache

After installing additional Apache modules (like `apache2-mod_apparmor`) or making configuration changes to Apache, profile Apache again to find out if additional rules need to be added to the profile. If you do not profile Apache again, it could be unable to start properly or be unable to serve Web pages.

29.4.4 How to Exclude Certain Profiles from the List of Profiles Used?

Run **`aa-disable`** *`PROGRAMNAME`* to disable the profile for *`PROGRAMNAME`*. This command creates a symbolic link to the profile in `/etc/apparmor.d/disable/`. To reactivate the profile, delete the link, and run **`systemctl reload apparmor`**.

29.4.5 Can I Manage Profiles for Applications not Installed on my System?

Managing profiles with AppArmor requires you to have access to the log of the system on which the application is running. So you do not need to run the application on your profile build host as long as you have access to the machine that runs the application. You can run the application on one system, transfer the logs (`/var/log/audit.log` or, if `audit` is not installed, **`journalctl | grep -i apparmor > path_to_logfile`**) to your profile build host and run **`aa-logprof -f`** *`PATH_TO_LOGFILE`*.

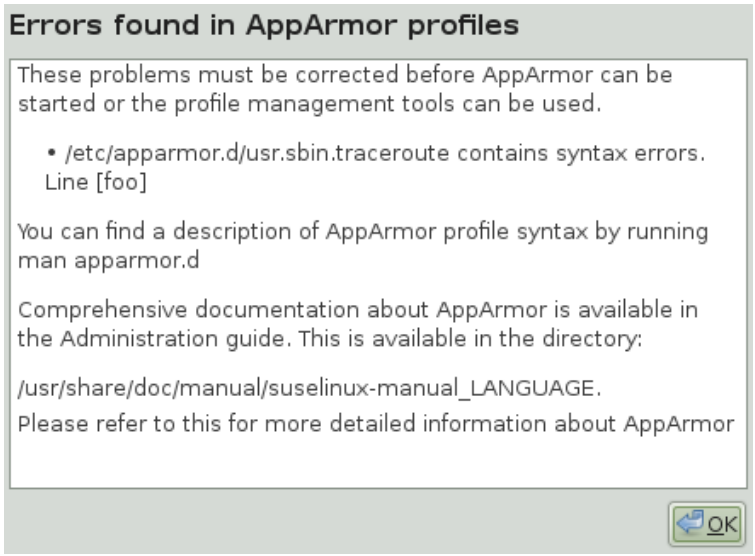
29.4.6 How to Spot and fix AppArmor Syntax Errors?

Manually editing AppArmor profiles can introduce syntax errors. If you attempt to start or restart AppArmor with syntax errors in your profiles, error results are shown. This example shows the syntax of the entire parser error.

```
localhost:~ # rcapparmor start
```

```
Loading AppArmor profiles AppArmor parser error in /etc/apparmor.d/usr.sbin.squid at line
 410: syntax error, unexpected TOK_ID, expecting TOK_MODE
 Profile /etc/apparmor.d/usr.sbin.squid failed to load
```

Using the AppArmor YaST tools, a graphical error message indicates which profile contained the error and requests you to fix it.



To fix a syntax error, log in to a terminal window as `root`, open the profile, and correct the syntax. Reload the profile set with **`systemctl reload apparmor`**.

Tip: AppArmor Syntax Highlighting in `vi`

The editor `vi` on openSUSE Leap supports syntax highlighting for AppArmor profiles. Lines containing syntax errors will be displayed with a red background.

29.5 Reporting Bugs for AppArmor

The developers of AppArmor are eager to deliver products of the highest quality. Your feedback and your bug reports help us keep the quality high. Whenever you encounter a bug in AppArmor, file a bug report against this product:

1. Use your Web browser to go to http://bugzilla.opensuse.org/ and click *Log In*.
2. Enter the account data of your SUSE account and click *Login*. If you do not have a SUSE account, click *Create Account* and provide the required data.

3. If your problem has already been reported, check this bug report and add extra information to it, if necessary.
4. If your problem has not been reported yet, select *New* from the top navigation bar and proceed to the *Enter Bug* page.
5. Select the product against which to file the bug. In your case, this would be your product's release. Click *Submit*.
6. Select the product version, component (AppArmor in this case), hardware platform, and severity.
7. Enter a brief headline describing your problem and add a more elaborate description including log files. You may create attachments to your bug report for screenshots, log files, or test cases.
8. Click *Submit* after you have entered all the details to send your report to the developers.

30 AppArmor Glossary

Abstraction

See *profile foundation classes* below.

Apache

Apache is a freely-available Unix-based Web server. It is currently the most commonly used Web server on the Internet. Find more information about Apache at the Apache Web site at http://www.apache.org.

application fire-walling

AppArmor confines applications and limits the actions they are permitted to take. It uses privilege confinement to prevent attackers from using malicious programs on the protected server and even using trusted applications in unintended ways.

attack signature

Pattern in system or network activity that alerts of a possible virus or hacker attack. Intrusion detection systems might use attack signatures to distinguish between legitimate and potentially malicious activity.

By not relying on attack signatures, AppArmor provides "proactive" instead of "reactive" defense from attacks. This is better because there is no window of vulnerability where the attack signature must be defined for AppArmor as it does for products using attack signatures.

GUI

Graphical user interface. Refers to a software front-end meant to provide an attractive and easy-to-use interface between a computer user and application. Its elements include windows, icons, buttons, cursors, and scrollbars.

globbing

File name substitution. Instead of specifying explicit file name paths, you can use helper characters `*` (substitutes any number of characters except special ones such as `/` or `?`) and `?` (substitutes exactly one character) to address multiple files/directories at once. `**` is a special substitution that matches any file or directory below the current directory.

HIP

Host intrusion prevention. Works with the operating system kernel to block abnormal application behavior in the expectation that the abnormal behavior represents an unknown attack. Blocks malicious packets on the host at the network level before they can “hurt” the application they target.

mandatory access control

A means of restricting access to objects that is based on fixed security attributes assigned to users, files, and other objects. The controls are mandatory in the sense that they cannot be modified by users or their programs.

profile

AppArmor profile completely defines what system resources an individual application can access, and with what privileges.

profile foundation classes

Profile building blocks needed for common application activities, such as DNS lookup and user authentication.

RPM

The RPM Package Manager. An open packaging system available for anyone to use. It works on Red Hat Linux, openSUSE Leap, and other Linux and Unix systems. It is capable of installing, uninstalling, verifying, querying, and updating computer software packages. See http://www.rpm.org/ for more information.

SSH

Secure Shell. A service that allows you to access your server from a remote computer and issue text commands through a secure connection.

streamlined access control

AppArmor provides streamlined access control for network services by specifying which files each program is allowed to read, write, and execute. This ensures that each program does what it is supposed to do and nothing else.

URI

Universal resource identifier. The generic term for all types of names and addresses that refer to objects on the World Wide Web. A URL is one kind of URI.

URL

Uniform Resource Locator. The global address of documents and other resources on the Web.

The first part of the address indicates what protocol to use and the second part specifies the IP address or the domain name where the resource is located.

For example, when you visit `http://www.opensuse.org`, you are using the HTTP protocol, as the beginning of the URL indicates.

vulnerabilities

An aspect of a system or network that leaves it open to attack. Characteristics of computer systems that allow an individual to keep it from correctly operating or that allows unauthorized users to take control of the system. Design, administrative, or implementation weaknesses or flaws in hardware, firmware, or software. If exploited, a vulnerability could lead to an unacceptable impact in the form of unauthorized access to information or the disruption of critical processing.

V SELinux

31 Configuring SELinux

In this chapter, you will learn how to set up and manage SELinux on openSUSE Leap. The following topics are covered:

- Why Use SELinux?
- Understanding SELinux
- Setting Up SELinux
- Managing SELinux

31.1 Why Use SELinux?

SELinux was developed as an additional Linux security solution that uses the security framework in the Linux kernel. The purpose was to allow for a more granular security policy that goes beyond what is offered by the default existing permissions of Read, Write, and Execute, and beyond assigning permissions to the different capabilities that are available on Linux. SELinux does this by trapping all system calls that reach the kernel, and denying them by default. This means that on a system that has SELinux enabled and nothing else configured, nothing will work. To allow your system to do anything, as an administrator you will need to write rules and put them in a policy.

An example explains why a solution such as SELinux (or its counterpart AppArmor) is needed:

“One morning, I found out that my server was hacked. The server was running a fully patched SLES installation. A firewall was configured on it and no unnecessary services were offered by this server. Further analysis revealed that the hacker had come in through a vulnerable PHP script that was a part of one of the Apache virtual hosts that were running on this server. The intruder had managed to get access to a shell, using the `wwwrun` account that was used by the Apache Web server. As this `wwwrun` user, the intruder had created several scripts in the `/var/tmp` and the `/tmp` directories, which were a part of a botnet that was launching a Distributed Denial of Service attack against several servers.”

The interesting thing about this hack is that it occurred on a server where nothing was really wrong. All permissions where set OK, but the intruder had managed to get into the system. What becomes clearly evident from this example is that in some cases additional security is needed—a security that goes beyond what is offered by using SELinux. As a less complete and less complex alternative, AppArmor can be used.

AppArmor confines specific processes in their abilities to read/write and execute files (and other things). Its view is mostly that things that happen inside a process cannot escape.

SELinux instead uses labels attached to objects (for example, files, binaries, network sockets) and uses them to determine privilege boundaries, thereby building up a level of confinement that can span more than a process or even the whole system.

SELinux was developed by the US National Security Agency (NSA), and since the beginning Red Hat has been heavily involved in its development. The first version of SELinux was offered in the era of Red Hat Enterprise Linux 4™, around the year 2006. In the beginning it offered support for essential services only, but over the years it has developed into a system that offers many rules that are collected in policies to offer protection to a broad range of services.

SELinux was developed in accordance with some certification standards like Common Criteria and FIPS 140. Because some customers specifically requested solutions that met these standards, SELinux rapidly became relatively popular.

As an alternative to SELinux, Immunix, a company that was purchased by Novell in 2005, had developed AppArmor. AppArmor was built on top of the same security principles as SELinux, but took a completely different approach, where it was possible to restrict services to exactly what they needed to do by using an easy to use wizard-driven procedure. Nevertheless, AppArmor has never reached the same status as SELinux, even if there are some good arguments to secure a server with AppArmor rather than with SELinux.

Because many organizations are requesting SELinux to be in the Linux distributions they are using, SUSE is offering support for the SELinux framework in openSUSE Leap. This does not mean that the default installation of openSUSE Leap will switch from AppArmor to SELinux in the near future.

31.1.1 Support Status

The SELinux framework is supported on openSUSE Leap. This means that openSUSE Leap offers all binaries and libraries you need to be able to use SELinux on your server. You may however miss some software that you may be familiar with from other Linux distributions.

SELinux support is at a fairly early stage in openSUSE Leap, which means that unexpected behavior may occur. To limit this risk as much as possible, it is best to use only the binaries that have been provided by default on openSUSE Leap.

31.1.2 Understanding SELinux Components

Before starting the configuration of SELinux, you should know a bit about how SELinux is organized. Three components play a role:

- The security framework in the Linux kernel
- The SELinux libraries and binaries
- The SELinux policy

The default kernel of openSUSE Leap supports SELinux and the tools that are needed to manage it. The most important part of the work of the administrator with regard to SELinux is managing the policy.

In the SELinux policy, security labels are applied to different objects on a Linux server. These objects typically are users, ports, processes and files. Using these security labels, rules are created that define what is and what is not allowed on a server. Remember, by default SELinux denies everything, and by creating the appropriate rules you can allow the access that is strictly necessary. Rules should therefore exist for all programs that you want to use on a system. Alternatively, you should configure parts of a system to run in unconfined mode, which means that specific ports, programs, users, files and directories are not protected by SELinux. This mode is useful if you only want to use SELinux to protect some essential services, while you are not specifically worried about other services. To get a really secure system, you should avoid this.

To ensure the appropriate protection of your system, you need an SELinux policy. This must be a tailor-made policy in which all files are provided with a label, and all services and users have a security label as well to express which files and directories can be accessed by which user and processed on the server. Developing such a policy is a tremendous amount of work.

The complexity of SELinux is also one of the main arguments against using it. Because a typical Linux system is so very complex, it is easy to overlook something and leave an opening that intruders can abuse to get into your system. And even if it is set up completely the way it should be, it still is very hard for an administrator to overlook all aspects with SELinux. With regard to the complexity, AppArmor takes a completely different approach and works with automated procedures that allow the administrator to set up AppArmor protection and understand exactly what is happening.

Note that a freely available SELinux policy might work on your server, but is unlikely to offer the same protection as a custom policy. SUSE also does not support third-party policies.

31.2 Policy

As mentioned, the policy is the key component in SELinux. It defines rules that specify which objects can access which files, directories, ports and processes on a system. To do this, a security context is defined for all of these. On an SELinux system where the policy has been applied to label the file system, you can use the `ls -Z` command on any directory to find the security context for the files in that directory. *Example 31.1: "Security Context Settings Using `ls -Z`"* shows the security context settings for the directories in the `/` directory of a openSUSE Leap system with an SELinux-labeled file system.

EXAMPLE 31.1: SECURITY CONTEXT SETTINGS USING `ls -Z`

```
ls -Z
system_u:object_r:bin_t bin
system_u:object_r:boot_t boot
system_u:object_r:device_t dev
system_u:object_r:etc_t etc
system_u:object_r:home_root_t home
system_u:object_r:lib_t lib
system_u:object_r:lib_t lib64
system_u:object_r:lost_found_t lost+found
system_u:object_r:mnt_t media
system_u:object_r:mnt_t mnt
system_u:object_r:usr_t opt
system_u:object_r:proc_t proc
system_u:object_r:default_t root
system_u:object_r:bin_t sbin
system_u:object_r:security_t selinux
system_u:object_r:var_t srv
system_u:object_r:sysfs_t sys
system_u:object_r:tmp_t tmp
system_u:object_r:usr_t usr
system_u:object_r:var_t var
```

The most important line in the security context is the context type. This is the part of the security context that ends in _t. It tells SELinux which kind of access the object is allowed. In the policy, rules are specified to define which type of user or which type of role has access to which type of context. For example, this can happen by using a rule like the following:

```
allow user_t bin_t:file {read execute gettattr};
```

This example rule states that the user who has the context type `user_t` (this user is called the source object) is allowed to access objects of class "file" with the context type `bin_t` (the target), using the permissions read, execute and getattr.

The standard policy that you are going to use contains a huge amount of rules. To make it more manageable, policies are often split into modules. This allows administrator to switch protection on or off for different parts of the system.

When compiling the policy for your system, you will have a choice to either work with a modular policy, or a monolithic policy, where one huge policy is used to protect everything on your system. It is strongly recommended to use a modular policy and not a monolithic policy. Modular policies are much easier to manage.

31.3 Installing SELinux Packages and Modifying GRUB 2

The easiest way to make sure that all SELinux components are installed is by using YaST. The procedure described below shows what to do on an installed openSUSE Leap:

1. Log in to your server as `root` and start YaST.
2. Select *Software* › *Software Management*
3. *Select View* › *Patterns* and select the entire C/C++ *Development* category for installation.
4. *Select View* › *Search* and make sure that *Search in Name*, *Keywords* and *Summary* are selected. Now enter the keyword `selinux` and click *Search*. You now see a list of packages.
5. Make sure that all the packages you have found are selected and click *Accept* to install them.

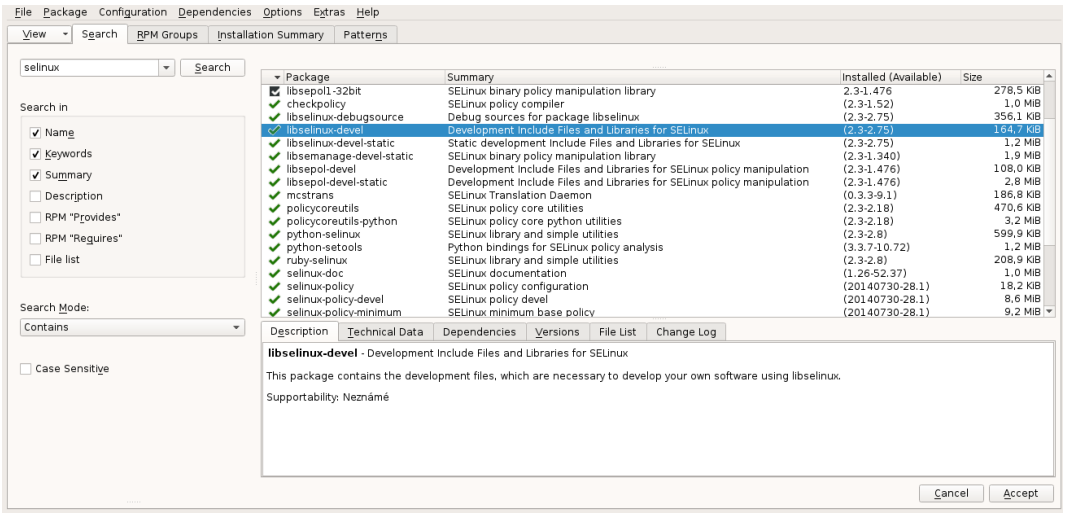


FIGURE 31.1: SELECTING ALL SELINUX PACKAGES IN YAST

After installing the SELinux packages, you need to modify the GRUB 2 boot loader. Do this from YaST, select *System* › *Boot Loader* › *Kernel Parameters*. Now add the following parameters to the *Optional Kernel Command Line Parameters*:

```
security=selinux selinux=1 enforcing=0
```

These options are used for the following purposes:

`security=selinux`
: This option tells the kernel to use SELinux and not AppArmor

`selinux=1`
: This option switches on SELinux

`enforcing=0`
: This option puts SELinux in permissive mode. In this mode, SELinux is fully functional, but does not enforce any of the security settings in the policy. Use this mode for configuring your system. To switch on SELinux protection, when the system is fully operational, change the option to `enforcing=1` and add `SELINUX=enforcing` in `/etc/selinux/config`.

After installing the SELinux packages and enabling the SELinux GRUB 2 boot parameters, reboot your server to activate the configuration.

31.4 SELinux Policy

The policy is an essential component of SELinux. openSUSE Leap 15.1 includes the *minimum* SELinux reference policy in the package `selinux-policy-minimum`. The examples in this chapter refer to this policy if not stated otherwise.

After installing the policy, you are ready to start file system labeling. Run

```
tux > sudo restorecon -Rp /
```

to start the **`/sbin/setfiles`** command to label all files on your system. The `/etc/selinux/minimum/contexts/files/file_contexts` input file is used. The `file_contexts` file needs to match your actual file system as much as possible. Otherwise, it can lead to a completely unbootable system. If that happens, modify the records in `file_contexts` with the **`semanage fcontext`** command to match the real structure of the file system your server is using. For example

```
tux > sudo semanage fcontext -a -t samba_share_t /etc/example_file
```

changes the file type from the default `etc_t` to `samba_share_t` and adds the following record to the related `file_contexts.local` file:

```
/etc/example_file    unconfined_u:object_r:samba_share_t:s0
```

Then run

```
tux > sudo restorecon -v /etc/example_file
```

for the type change to take effect.

Before doing this, make sure to read the rest of this chapter, so you fully understand how context type is applied to files and directories. Do not forget to make a backup of the `file_contexts` file before starting.

Note: The User nobody

While using **`semanage`**, you may get a message that complains about the home directory of `nobody`. In this case, change the login shell of user `nobody` to `/sbin/nologin`. Then the settings of `nobody` match the current policy settings.

After another reboot SELinux should be operational. To verify this, use the command **`sestatus -v`**. It should give you an output similar to *Example 31.2: "Verifying that SELinux is functional"*.

EXAMPLE 31.2: VERIFYING THAT SELINUX IS FUNCTIONAL

```
tux > sudo sestatus -v
SELinux status:                 enabled
SELinuxfs mount:                /selinux
Current mode:                   permissive
Mode from config file:          permissive
Policy version:                 26
Policy from config file:        minimum

Process contexts:
Current context:                root:staff_r:staff_t
Init context:                   system_u:system_r:init_t
/sbin/mingetty                  system_u:system_r:sysadm_t
/usr/sbin/sshd                  system_u:system_r:sshd_t

File contexts:
Controlling term:               root:object_r:user_devpts_t
/etc/passwd                     system_u:object_r:etc_t
/etc/shadow                     system_u:object_r:shadow_t
/bin/bash                       system_u:object_r:shell_exec_t
/bin/login                      system_u:object_r:login_exec_t
```

```
/bin/sh                         system_u:object_r:bin_t -> system_u:object_r:shell_exec_t
/sbin/agetty                    system_u:object_r:getty_exec_t
/sbin/init                      system_u:object_r:init_exec_t
/sbin/mingetty                  system_u:object_r:getty_exec_t
/usr/sbin/sshd                  system_u:object_r:sshd_exec_t
/lib/libc.so.6                  system_u:object_r:lib_t -> system_u:object_r:lib_t
/lib/ld-linux.so.2              system_u:object_r:lib_t -> system_u:object_r:ld_so_t
```

31.5 Configuring SELinux

At this point you have a completely functional SELinux system and it is time to further configure it. In the current status, SELinux is operational but not in enforcing mode. This means that it does not limit you in doing anything, it logs everything that it should be doing if it were in enforcing mode. This is good, because based on the log files you can find what it is that it would prevent you from doing. As a first test, put SELinux in enforcing mode and find out if you can still use your server after doing so: check that the option `enforcing=1` is set in the GRUB 2 configuration file, while `SELINUX=enforcing` is set in `/etc/selinux/config`. Reboot your server and see if it still comes up the way you expect it to. If it does, leave it like that and start modifying the server in a way that everything works as expected. However, you may not even be able to boot the server properly. In that case, switch back to the mode where SELinux is not enforcing and start tuning your server.

Before you start tuning your server, verify the SELinux installation. You have already used the command **`sestatus -v`** to view the current mode, process, and file contexts. Next, run

```
tux > sudo semanage boolean -l
```

which lists all Boolean switches that are available, and at the same time verifies that you can access the policy. *Example 31.3, "Getting a List of Booleans and Verifying Policy Access"* shows part of the output of this command.

EXAMPLE 31.3: GETTING A LIST OF BOOLEANS AND VERIFYING POLICY ACCESS

```
tux > sudo semanage boolean -l
SELinux boolean                              Description
ftp_home_dir                   -> off   ftp_home_dir
mozilla_read_content           -> off   mozilla_read_content
spamassassin_can_network       -> off   spamassassin_can_network
httpd_can_network_relay        -> off   httpd_can_network_relay
openvpn_enable_homedirs        -> off   openvpn_enable_homedirs
gpg_agent_env_file             -> off   gpg_agent_env_file
allow_httpd_awstats_script_anon_write -> off   allow_httpd_awstats_script_anon_write
httpd_can_network_connect_db   -> off   httpd_can_network_connect_db
```

```
allow_ftpd_full_access          -> off   allow_ftpd_full_access
samba_domain_controller         -> off   samba_domain_controller
httpd_enable_cgi                -> off   httpd_enable_cgi
virt_use_nfs                    -> off   virt_use_nfs
```

Another command that outputs useful information at this stage is

```
tux > sudo semanage fcontext -l
```

It shows the default file context settings as provided by the policy (see *Example 31.4: "Getting File Context Information"* for partial output of this command).

EXAMPLE 31.4: GETTING FILE CONTEXT INFORMATION

```
tux > sudo semanage fcontext -l
/var/run/usb(/.*)?                                  all files
 system_u:object_r:hotplug_var_run_t
/var/run/utmp                                       regular file
 system_u:object_r:initrc_var_run_t
/var/run/vbe.*                                      regular file
 system_u:object_r:hald_var_run_t
/var/run/vmnat.*                                    socket
 system_u:object_r:vmware_var_run_t
/var/run/vmware.*                                   all files
 system_u:object_r:vmware_var_run_t
/var/run/watchdog\.pid                              regular file
 system_u:object_r:watchdog_var_run_t
/var/run/winbindd(/.*)?                             all files
 system_u:object_r:winbind_var_run_t
/var/run/wnn-unix(/.*)                              all files
 system_u:object_r:canna_var_run_t
/var/run/wpa_supplicant(/.*)?                       all files
 system_u:object_r:NetworkManager_var_run_t
/var/run/wpa_supplicant-global                      socket
 system_u:object_r:NetworkManager_var_run_t
/var/run/xdmctl(/.*)?                               all files
 system_u:object_r:xdm_var_run_t
/var/run/yiff-[0-9]+\.pid                           regular file
 system_u:object_r:soundd_var_run_t
```

31.6 Managing SELinux

The base SELinux configuration is now operational and it can now be configured to secure your server. In SELinux, an additional set of rules is used to define exactly which process or user can access which files, directories, or ports. To do this, SELinux applies a context to every file,

directory, process, and port. This context is a security label that defines how this file, directory, process, or port should be treated. These context labels are used by the SELinux policy, which defines exactly what should be done with the context labels. By default, the policy blocks all non-default access, which means that, as an administrator, you need to enable all features that are non-default on your server.

31.6.1 Viewing the Security Context

As already mentioned, files, directories, and ports can be labeled. Within each label, different contexts are used. To be able to perform your daily administration work, the type context is what you are most interested in. As an administrator, you will mostly work with the type context. Many commands allow you to use the `-Z` option to list current context settings. In *Example 31.5: "The default context for directories in the root directory"* you can see what the context settings are for the directories in the root directory.

EXAMPLE 31.5: THE DEFAULT CONTEXT FOR DIRECTORIES IN THE ROOT DIRECTORY

```
tux > sudo ls -Z
dr-xr-xr-x. root root system_u:object_r:bin_t:s0       bin
dr-xr-xr-x. root root system_u:object_r:boot_t:s0      boot
drwxr-xr-x. root root system_u:object_r:cgroup_t:s0    cgroup
drwxr-xr-x+ root root unconfined_u:object_r:default_t:s0 data
drwxr-xr-x. root root system_u:object_r:device_t:s0    dev
drwxr-xr-x. root root system_u:object_r:etc_t:s0       etc
drwxr-xr-x. root root system_u:object_r:home_root_t:s0 home
dr-xr-xr-x. root root system_u:object_r:lib_t:s0       lib
dr-xr-xr-x. root root system_u:object_r:lib_t:s0       lib64
drwx------. root root system_u:object_r:lost_found_t:s0 lost+found
drwxr-xr-x. root root system_u:object_r:mnt_t:s0       media
drwxr-xr-x. root root system_u:object_r:autofs_t:s0    misc
drwxr-xr-x. root root system_u:object_r:mnt_t:s0       mnt
drwxr-xr-x. root root unconfined_u:object_r:default_t:s0 mnt2
drwxr-xr-x. root root unconfined_u:object_r:default_t:s0 mounts
drwxr-xr-x. root root system_u:object_r:autofs_t:s0    net
drwxr-xr-x. root root system_u:object_r:usr_t:s0       opt
dr-xr-xr-x. root root system_u:object_r:proc_t:s0      proc
drwxr-xr-x. root root unconfined_u:object_r:default_t:s0 repo
dr-xr-x---. root root system_u:object_r:admin_home_t:s0 root
dr-xr-xr-x. root root system_u:object_r:bin_t:s0       sbin
drwxr-xr-x. root root system_u:object_r:security_t:s0  selinux
drwxr-xr-x. root root system_u:object_r:var_t:s0       srv
-rw-r--r--. root root unconfined_u:object_r:swapfile_t:s0 swapfile
drwxr-xr-x. root root system_u:object_r:sysfs_t:s0     sys
```

```
drwxrwxrwt. root root system_u:object_r:tmp_t:s0       tmp
-rw-r--r--. root root unconfined_u:object_r:etc_runtime_t:s0 tmp2.tar
-rw-r--r--. root root unconfined_u:object_r:etc_runtime_t:s0 tmp.tar
drwxr-xr-x. root root system_u:object_r:usr_t:s0       usr
drwxr-xr-x. root root system_u:object_r:var_t:s0       var
```

In the listing above, you can see the complete context for all directories. It consists of a user, a role, and a type. The s0 setting indicates the security level in Multi Level Security environments. These environments are not discussed here. In such an environment, make sure that s0 is set. The Context Type defines what kind of activity is permitted in the directory. Compare, for example, the `/root` directory, which has the `admin_home_t` context type, and the `/home` directory, which has the `home_root_t` context type. In the SELinux policy, different kinds of access are defined for these context types.

Security labels are not only associated with files, but also with other items, such as ports and processes. In *Example 31.6: "Showing SELinux settings for processes with* **ps Zaux***"* for example you can see the context settings for processes on your server.

EXAMPLE 31.6: SHOWING SELINUX SETTINGS FOR PROCESSES WITH **ps Zaux**

```
tux > sudo ps Zaux
LABEL                             USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START
   TIME COMMAND
system_u:system_r:init_t          root         1  0.0  0.0  10640   808 ?        Ss   05:31
   0:00 init [5]
system_u:system_r:kernel_t        root         2  0.0  0.0      0     0 ?        S    05:31
   0:00 [kthreadd]
system_u:system_r:kernel_t        root         3  0.0  0.0      0     0 ?        S    05:31
   0:00 [ksoftirqd/0]
system_u:system_r:kernel_t        root         6  0.0  0.0      0     0 ?        S    05:31
   0:00 [migration/0]
system_u:system_r:kernel_t        root         7  0.0  0.0      0     0 ?        S    05:31
   0:00 [watchdog/0]
system_u:system_r:sysadm_t        root      2344  0.0  0.0  27640   852 ?        Ss   05:32
   0:00 /usr/sbin/mcelog --daemon --config-file /etc/mcelog/mcelog.conf
system_u:system_r:sshd_t          root      3245  0.0  0.0  69300  1492 ?        Ss   05:32
   0:00 /usr/sbin/sshd -o PidFile=/var/run/sshd.init.pid
system_u:system_r:cupsd_t         root      3265  0.0  0.0  68176  2852 ?        Ss   05:32
   0:00 /usr/sbin/cupsd
system_u:system_r:nscd_t          root      3267  0.0  0.0 772876  1380 ?        Ssl  05:32
   0:00 /usr/sbin/nscd
system_u:system_r:postfix_master_t root   3334  0.0  0.0  38320  2424 ?        Ss   05:32
   0:00 /usr/lib/postfix/master
system_u:system_r:postfix_qmgr_t postfix  3358  0.0  0.0  40216  2252 ?        S    05:32
   0:00 qmgr -l -t fifo -u
```

```
system_u:system_r:crond_t       root      3415  0.0  0.0  14900   800 ?        Ss   05:32
   0:00 /usr/sbin/cron
system_u:system_r:fsdaemon_t    root      3437  0.0  0.0  16468  1040 ?        S    05:32
   0:00 /usr/sbin/smartd
system_u:system_r:sysadm_t      root      3441  0.0  0.0  66916  2152 ?        Ss   05:32
   0:00 login -- root
system_u:system_r:sysadm_t      root      3442  0.0  0.0   4596   800 tty2     Ss+  05:32
   0:00 /sbin/mingetty tty2
```

31.6.2 Selecting the SELinux Mode

In SELinux, three different modes can be used:

Enforcing:

This is the default mode. SELinux protects your server according to the rules in the policy, and SELinux logs all of its activity to the audit log.

Permissive:

This mode is useful for troubleshooting. If set to Permissive, SELinux does not protect your server, but it still logs everything that happens to the log files.

Disabled:

In this mode, SELinux is switched off completely and no logging occurs. The file system labels however are not removed from the file system.

You have already read how you can set the current SELinux mode from GRUB 2 while booting using the enforcing boot parameter.

31.6.3 Modifying SELinux Context Types

An important part of the work of an administrator is setting context types on files to ensure appropriate working of SELinux.

If a file is created within a specific directory, it inherits the context type of the parent directory by default. If, however, a file is moved from one location to another location, it retains the context type that it had in the old location.

To set the context type for files, you can use the **`semanage fcontext`** command. With this command, you write the new context type to the policy, but it does not change the actual context type immediately! To apply the context types that are in the policy, you need to run the **`restorecon`** command afterward.

The challenge when working with `semanage fcontext` is to find out which context you actually need. You can use

```
tux > sudo semanage fcontext -l
```

to list all contexts in the policy, but it may be a bit hard to find out the actual context you need from that list as it is rather long (see *Example 31.7: "Viewing Default File Contexts"*).

EXAMPLE 31.7: VIEWING DEFAULT FILE CONTEXTS

```
tux > sudo semanage fcontext -l | less
SELinux fcontext                                    type               Context

/                                                   directory
 system_u:object_r:root_t:s0
/.*                                                 all files
 system_u:object_r:default_t:s0
/[^/]+                                              regular file
 system_u:object_r:etc_runtime_t:s0
/\.autofsck                                         regular file
 system_u:object_r:etc_runtime_t:s0
/\.autorelabel                                      regular file
 system_u:object_r:etc_runtime_t:s0
/\.journal                                          all files          X:>>None>>
/\.suspended                                        regular file
 system_u:object_r:etc_runtime_t:s0
/a?quota\.(user|group)                              regular file
 system_u:object_r:quota_db_t:s0
/afs                                                directory
 system_u:object_r:mnt_t:s0
/bin                                                directory
 system_u:object_r:bin_t:s0
/bin/.*                                             all files
 system_u:object_r:bin_t:s0
```

There are three ways to find out which context settings are available for your services:

- Install the service and look at the default context settings that are used. This is the easiest and recommended option.
- Consult the man page for the specific service. Some services have a man page that ends in `_selinux`, which contains all the information you need to find the correct context settings. When you have found the right context setting, apply it using `semanage fcontext`. This command takes `-t` context type as its first argument, followed by the name of the directory or file to which you want to apply the context settings. To apply the context to everything

that already exists in the directory where you want to apply the context, you add the regular expression `(/.*)?` to the name of the directory. This means: optionally, match a slash followed by any character. The examples section of the **`semanage`** man page has some useful usage examples for **`semanage`**. For more information on regular expressions, see for example the tutorial at http://www.regular-expressions.info/.

- Display a list of all context types that are available on your system:

```
tux > sudo seinfo -t
```

Since the command by itself outputs an overwhelming amount of information, it should be used in combination with **`grep`** or a similar command for filtering.

31.6.4 Applying File Contexts

To help you apply the SELinux context properly, the following procedure shows how to set a context using **`semanage fcontext`** and **`restorecon`**. You will notice that at first attempt, the Web server with a non-default document root does not work. After changing the SELinux context, it will:

1. Create the `/web` directory and then change to it:

```
tux > sudo mkdir /web  && cd /web
```

2. Use a text editor to create the file `/web/index.html` that contains the text welcome to my Web site.

3. Open the file `/etc/apache2/default-server.conf` with an editor, and change the DocumentRoot line to `DocumentRoot /web`

4. Start the Apache Web server:

```
tux > sudo systemctl start apache2
```

5. Open a session to your local Web server:

```
tux > w3m localhost
```

You will receive a *Connection refused* message. Press `Enter`, and then **`q`** to quit w3m.

6. Find the current context type for the default Apache `DocumentRoot`, which is `/srv/www/htdocs`. It should be set to `httpd_sys_content_t`:

```
tux > sudo ls -Z /srv/www
```

7. Set the new context in the policy and press `Enter`:

```
tux > sudo semanage fcontext -a -f "" -t httpd_sys_content_t '/web(/.*) ?'
```

8. Apply the new context type:

```
tux > sudo restorecon /web
```

9. Show the context of the files in the directory `/web`. You will see that the new context type has been set properly to the `/web` directory, but not to its contents.

```
tux > sudo ls -Z /web
```

10. Apply the new context recursively to the `/web` directory. The type context has now been set correctly.

```
tux > sudo restorecon -R /web
```

11. Restart the Web server:

```
tux > sudo systemctl restart apache2
```

You should now be able to access the contents of the `/web` directory.

31.6.5 Configuring SELinux Policies

The easiest way to change the behavior of the policy is by working with Booleans. These are on-off switches that you can use to change the settings in the policy. To find out which Booleans are available, run

```
tux > sudo semanage boolean -l
```

It will show a long list of Booleans, with a short description of what each of these Booleans will do for you. When you have found the Boolean you want to set, you can use **`setsebool -P`**, followed by the name of the Boolean that you want to change. It is important to use the `-`

`P` option at all times when using **`setsebool`**. This option writes the setting to the policy file on disk, and this is the only way to make sure that the Boolean is applied automatically after a reboot.

The procedure below gives an example of changing Boolean settings

1. List Booleans that are related to FTP servers.

   ```
   tux > sudo semanage boolean -l | grep ftp
   ```

2. Turn the Boolean off:

   ```
   tux > sudo setsebool allow_ftpd_anon_write off
   ```

   Note that it does not take much time to write the change. Then verify that the Boolean is indeed turned off:

   ```
   tux > sudo semanage boolean -l|grep ftpd_anon
   ```

3. Reboot your server.

4. Check again to see if the `allow_ftpd_anon_write` Boolean is still turned on. As it has not yet been written to the policy, you will notice that it is off.

5. Switch the Boolean and write the setting to the policy:

   ```
   tux > sudo setsebool -P allow_ftpd_anon_write
   ```

31.6.6 Working with SELinux Modules

By default, SELinux uses a modular policy. This means that the policy that implements SELinux features is not just one huge policy, but it consists of many smaller modules. Each module covers a specific part of the SELinux configuration. The concept of the SELinux module was introduced to make it easier for third party vendors to make their services compatible with SELinux. To get an overview of the SELinux modules, you can use the **`semodule -l`** command. This command lists all current modules in use by SELinux and their version numbers.

As an administrator, you can switch modules on or off. This can be useful if you want to disable only a part of SELinux and not everything to run a specific service without SELinux protection. Especially in the case of openSUSE Leap, where there is not a completely supported SELinux policy yet, it can make sense to switch off all modules that you do not need so that you can focus on the services that really do need SELinux protection. To switch off an SELinux module, use

```
tux > sudo semodule -d MODULENAME
```

To switch it on again, you can use

```
tux > sudo semodule -e modulename
```

As an administrator, you do not typically change the contents of the policy files that come from the SELinux Policy RPM. You would rather use **`semanage fcontext`** to change file contexts. If you are using **`audit2allow`** to generate policies for your server, you should change the policy files after all.

To change the contents of any of the policy module files, compile the changes into a new policy module file. To do this, first install the `selinux-policy-devel` package. Then, in the directory where the files created by **`audit2allow`** are located, run:

```
tux > make -f /usr/share/selinux/devel/Makefile
```

When **`make`** has completed, you can manually load the modules into the system, using **`semodule -i`**.

31.7 Troubleshooting

By default, if SELinux is the reason something is not working, a log message to this effect is sent to the `/var/log/audit/audit.log` file. That is, if the auditd service is running. If you see an empty `/var/log/audit`, start the auditd service using

```
tux > sudo systemctl start auditd
```

and enable it in the targets of your system, using

```
tux > sudo systemctl enable auditd
```

In *Example 31.8: "Example Lines from `/etc/audit/audit.log`"* you can see a partial example of the contents of `/var/log/audit/audit.log`

EXAMPLE 31.8: EXAMPLE LINES FROM `/etc/audit/audit.log`

```
type=DAEMON_START msg=audit(1348173810.874:6248): auditd start, ver=1.7.7 format=raw
 kernel=3.0.13-0.27-default auid=0 pid=4235 subj=system_u:system_r:auditd_t res=success
type=AVC msg=audit(1348173901.081:292): avc:  denied  { write } for
  pid=3426 comm="smartd" name="smartmontools" dev=sda6 ino=581743
 scontext=system_u:system_r:fsdaemon_t tcontext=system_u:object_r:var_lib_t tclass=dir
```

```
type=AVC msg=audit(1348173901.081:293): avc:  denied  { remove_name } for  pid=3426
 comm="smartd" name="smartd.WDC_WD2500BEKT_75PVMT0-WD_WXC1A21E0454.ata.state~" dev=sda6
 ino=582390 scontext=system_u:system_r:fsdaemon_t tcontext=system_u:object_r:var_lib_t
 tclass=dir
type=AVC msg=audit(1348173901.081:294): avc:  denied  { unlink } for  pid=3426
 comm="smartd" name="smartd.WDC_WD2500BEKT_75PVMT0-WD_WXC1A21E0454.ata.state~" dev=sda6
 ino=582390 scontext=system_u:system_r:fsdaemon_t tcontext=system_u:object_r:var_lib_t
 tclass=file
type=AVC msg=audit(1348173901.081:295): avc:  denied  { rename } for  pid=3426
 comm="smartd" name="smartd.WDC_WD2500BEKT_75PVMT0-WD_WXC1A21E0454.ata.state" dev=sda6
 ino=582373 scontext=system_u:system_r:fsdaemon_t tcontext=system_u:object_r:var_lib_t
 tclass=file
type=AVC msg=audit(1348173901.081:296): avc:  denied  { add_name } for  pid=3426
 comm="smartd" name="smartd.WDC_WD2500BEKT_75PVMT0-WD_WXC1A21E0454.ata.state~"
 scontext=system_u:system_r:fsdaemon_t tcontext=system_u:object_r:var_lib_t tclass=dir
type=AVC msg=audit(1348173901.081:297): avc:  denied  { create } for  pid=3426
 comm="smartd" name="smartd.WDC_WD2500BEKT_75PVMT0-WD_WXC1A21E0454.ata.state"
 scontext=system_u:system_r:fsdaemon_t tcontext=system_u:object_r:var_lib_t tclass=file
type=AVC msg=audit(1348173901.081:298): avc:  denied  { write open } for  pid=3426
 comm="smartd" name="smartd.WDC_WD2500BEKT_75PVMT0-WD_WXC1A21E0454.ata.state" dev=sda6
 ino=582390 scontext=system_u:system_r:fsdaemon_t tcontext=system_u:object_r:var_lib_t
 tclass=file
type=AVC msg=audit(1348173901.081:299): avc:  denied  { getattr } for  pid=3426
 comm="smartd" path="/var/lib/smartmontools/smartd.WDC_WD2500BEKT_75PVMT0-
WD_WXC1A21E0454.ata.state" dev=sda6 ino=582390 scontext=system_u:system_r:fsdaemon_t
 tcontext=system_u:object_r:var_lib_t tclass=file
type=AVC msg=audit(1348173901.309:300): avc:  denied  { append } for  pid=1316
```

At first look, the lines in `audit.log` are a bit hard to read. However, on closer examination they are not that hard to understand. Every line can be broken down into sections. For example, the sections in the last line are:

`type=AVC`:

every SELinux-related audit log line starts with the type identification `type=AVC`

`msg=audit(1348173901.309:300)`:

This is the time stamp, which unfortunately is written in epoch time, the number of seconds that have passed since Jan 1, 1970. You can use **`date -d`** on the part up to the dot in the epoch time notation to find out when the event has happened:

```
tux > date -d @1348173901
Thu Sep 20 16:45:01 EDT 2012
```

`avc: denied { append }`:

the specific action that was denied. In this case the system has denied the appending of data to a file. While browsing through the audit log file, you can see other system actions, such as write open, getattr and more.

`for pid=1316`:
: the process ID of the command or process that initiated the action

`comm="rsyslogd"`:
: the specific command that was associated with that PID

`name="smartmontools"`:
: the name of the subject of the action

`dev=sda6 ino=582296`:
: the block device and inode number of the file that was involved

`scontext=system_u:system_r:syslogd_t`:
: the source context, which is the context of the initiator of the action

`tclass=file`:
: a class identification of the subject

Instead of interpreting the events in audit.log yourself, there is another approach. You can use the **`audit2allow`** command, which helps analyze the cryptic log messages in `/var/log/audit/audit.log`. An audit2allow troubleshooting session always consists of three different commands. First, you would use **`audit2allow -w -a`** to present the audit information in a more readable way. The **`audit2allow -w -a`** by default works on the audit.log file. If you want to analyze a specific message in the audit.log file, copy it to a temporary file and analyze the file with:

```
tux > sudo audit2allow -w -i FILENAME
```

EXAMPLE 31.9: ANALYZING AUDIT MESSAGES

```
tux > sudo audit2allow -w -i testfile
type=AVC msg=audit(1348173901.309:300): avc:  denied  { append } for  pid=1316
comm="rsyslogd" name="acpid" dev=sda6 ino=582296
scontext=system_u:system_r:syslogd_t tcontext=system_u:object_r:apmd_log_t tclass=file
```

This was caused by:

Missing type enforcement (TE) allow rule.

To generate a loadable module to allow this access, run

```
tux > sudo audit2allow
```

To find out which specific rule has denied access, you can use **audit2allow -a** to show the enforcing rules from all events that were logged to the audit.log file, or **audit2allow -i** *FILENAME* to show it for messages that you have stored in a specific file:

EXAMPLE 31.10: VIEWING WHICH LINES DENY ACCESS

```
tux > sudo audit2allow -i testfile
#============= syslogd_t ==============
allow syslogd_t apmd_log_t:file append;
```

To create an SELinux module with the name mymodule that you can load to allow the access that was previously denied, run

```
tux > sudo audit2allow -a -R -M mymodule
```

If you want to do this for all events that have been logged to the audit.log, use the -a -M command arguments. To do it only for specific messages that are in a specific file, use -i -M as in the example below:

EXAMPLE 31.11: CREATING A POLICY MODULE ALLOWING AN ACTION PREVIOUSLY DENIED

```
tux > sudo audit2allow -i testfile -M example
****************** IMPORTANT ***********************
To make this policy package active, execute:

semodule -i example.pp
```

As indicated by the **audit2allow** command, you can now run this module by using the **semodule -i** command, followed by the name of the module that **audit2allow** has created for you (example.pp in the above example).

VI *The Linux Audit Framework*

32 Understanding Linux Audit

The Linux audit framework as shipped with this version of openSUSE Leap provides a CAPP-compliant (Controlled Access Protection Profiles) auditing system that reliably collects information about any security-relevant event. The audit records can be examined to determine whether any violation of the security policies has been committed, and by whom.

Providing an audit framework is an important requirement for a CC-CAPP/EAL (Common Criteria-Controlled Access Protection Profiles/Evaluation Assurance Level) certification. Common Criteria (CC) for Information Technology Security Information is an international standard for independent security evaluations. Common Criteria helps customers judge the security level of any IT product they intend to deploy in mission-critical setups.

Common Criteria security evaluations have two sets of evaluation requirements, functional and assurance requirements. Functional requirements describe the security attributes of the product under evaluation and are summarized under the Controlled Access Protection Profiles (CAPP). Assurance requirements are summarized under the Evaluation Assurance Level (EAL). EAL describes any activities that must take place for the evaluators to be confident that security attributes are present, effective, and implemented. Examples for activities of this kind include documenting the developers' search for security vulnerabilities, the patch process, and testing.

This guide provides a basic understanding of how audit works and how it can be set up. For more information about Common Criteria itself, refer to the Common Criteria Web site (https://www.commoncriteriaportal.org/).

Linux audit helps make your system more secure by providing you with a means to analyze what is happening on your system in great detail. It does not, however, provide additional security itself—it does not protect your system from code malfunctions or any kind of exploits. Instead, audit is useful for tracking these issues and helps you take additional security measures, like AppArmor, to prevent them.

Audit consists of several components, each contributing crucial functionality to the overall framework. The audit kernel module intercepts the system calls and records the relevant events. The `auditd` daemon writes the audit reports to disk. Various command line utilities take care of displaying, querying, and archiving the audit trail.

Audit enables you to do the following:

Associate Users with Processes

Audit maps processes to the user ID that started them. This makes it possible for the administrator or security officer to exactly trace which user owns which process and is potentially doing malicious operations on the system.

Important: Renaming User IDs

Audit does not handle the renaming of UIDs. Therefore avoid renaming UIDs (for example, changing `tux` from `uid=1001` to `uid=2000`) and obsolete UIDs rather than renaming them. Otherwise you would need to change **`auditctl`** data (audit rules) and would have problems retrieving old data correctly.

Review the Audit Trail

Linux audit provides tools that write the audit reports to disk and translate them into human readable format.

Review Particular Audit Events

Audit provides a utility that allows you to filter the audit reports for certain events of interest. You can filter for:

- User
- Group
- Audit ID
- Remote Host Name
- Remote Host Address
- System Call
- System Call Arguments

- File
- File Operations
- Success or Failure

Apply a Selective Audit

Audit provides the means to filter the audit reports for events of interest and to tune audit to record only selected events. You can create your own set of rules and have the audit daemon record only those of interest to you.

Guarantee the Availability of the Report Data

Audit reports are owned by `root` and therefore only removable by `root`. Unauthorized users cannot remove the audit logs.

Prevent Audit Data Loss

If the kernel runs out of memory, the audit daemon's backlog is exceeded, or its rate limit is exceeded, audit can trigger a shutdown of the system to keep events from escaping audit's control. This shutdown would be an immediate halt of the system triggered by the audit kernel component without synchronizing the latest logs to disk. The default configuration is to log a warning to syslog rather than to halt the system.

If the system runs out of disk space when logging, the audit system can be configured to perform clean shutdown. The default configuration tells the audit daemon to stop logging when it runs out of disk space.

32.1 Introducing the Components of Linux Audit

The following figure illustrates how the various components of audit interact with each other:

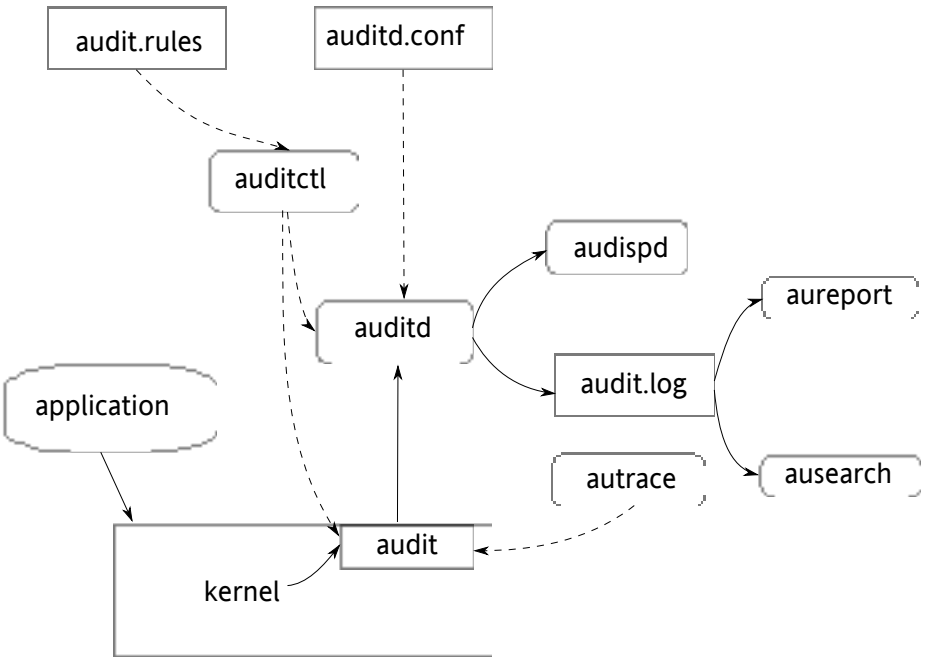


FIGURE 32.1: INTRODUCING THE COMPONENTS OF LINUX AUDIT

Straight arrows represent the data flow between components while dashed arrows represent lines of control between components.

auditd

The audit daemon is responsible for writing the audit messages that were generated through the audit kernel interface and triggered by application and system activity to disk. The way the audit daemon is started is controlled by `systemd`. The audit system functions (when started) are controlled by `/etc/audit/auditd.conf`. For more information about `auditd` and its configuration, refer to *Section 32.2, "Configuring the Audit Daemon"*.

`auditctl`

The **`auditctl`** utility controls the audit system. It controls the log generation parameters and kernel settings of the audit interface and the rule sets that determine which events are tracked. For more information about **`auditctl`**, refer to *Section 32.3, "Controlling the Audit System Using* **`auditctl`***"*.

audit rules

The file `/etc/audit/audit.rules` contains a sequence of **`auditctl`** commands that are loaded at system boot time immediately after the audit daemon is started. For more information about audit rules, refer to *Section 32.4, "Passing Parameters to the Audit System"*.

aureport

The **`aureport`** utility allows you to create custom reports from the audit event log. This report generation can easily be scripted, and the output can be used by various other applications, for example, to plot these results. For more information about **`aureport`**, refer to *Section 32.5, "Understanding the Audit Logs and Generating Reports"*.

ausearch

The **`ausearch`** utility can search the audit log file for certain events using various keys or other characteristics of the logged format. For more information about **`ausearch`**, refer to *Section 32.6, "Querying the Audit Daemon Logs with **`ausearch`**"*.

audispd

The audit dispatcher daemon (`audispd`) can be used to relay event notifications to other applications instead of (or in addition to) writing them to disk in the audit log. For more information about `audispd`, refer to *Section 32.9, "Relaying Audit Event Notifications"*.

autrace

The **`autrace`** utility traces individual processes in a fashion similar to **`strace`**. The output of **`autrace`** is logged to the audit log. For more information about **`autrace`**, refer to *Section 32.7, "Analyzing Processes with **`autrace`**"*.

aulast

Prints a list of the last logged-in users, similarly to **`last`**. **`aulast`** searches back through the audit logs (or the given audit log file) and displays a list of all users logged in and out based on the range of time in the audit logs.

aulastlog

Prints the last login for all users of a machine similar to the way **`lastlog`** does. The login name, port, and last login time will be printed.

32.2 Configuring the Audit Daemon

Before you can actually start generating audit logs and processing them, configure the audit daemon itself. The `/etc/audit/auditd.conf` configuration file determines how the audit system functions when the daemon has been started. For most use cases, the default settings shipped with openSUSE Leap should suffice. For CAPP environments, most of these parameters need tweaking. The following list briefly introduces the parameters available:

```
log_file = /var/log/audit/audit.log
log_format = RAW
log_group = root
priority_boost = 4
flush = INCREMENTAL
freq = 20
num_logs = 5
disp_qos = lossy
dispatcher = /sbin/audispd
name_format = NONE
##name = mydomain
max_log_file = 6
max_log_file_action = ROTATE
space_left = 75
space_left_action = SYSLOG
action_mail_acct = root
admin_space_left = 50
admin_space_left_action = SUSPEND
disk_full_action = SUSPEND
disk_error_action = SUSPEND
##tcp_listen_port =
tcp_listen_queue = 5
tcp_max_per_addr = 1
##tcp_client_ports = 1024-65535
tcp_client_max_idle = 0
cp_client_max_idle = 0
```

Depending on whether you want your environment to satisfy the requirements of CAPP, you need to be extra restrictive when configuring the audit daemon. Where you need to use particular settings to meet the CAPP requirements, a “CAPP Environment” note tells you how to adjust the configuration.

`log_file`, `log_format` and `log_group`

`log_file` specifies the location where the audit logs should be stored. `log_format` determines how the audit information is written to disk and `log_group` defines the group that owns the log files. Possible values for `log_format` are `raw` (messages are stored

exactly as the kernel sends them) or `nolog` (messages are discarded and not written to disk). The data sent to the audit dispatcher is not affected if you use the `nolog` mode. The default setting is `raw` and you should keep it if you want to be able to create reports and queries against the audit logs using the **`aureport`** and **`ausearch`** tools. The value for `log_group` can either be specified literally or using the group's ID.

Note: CAPP Environment

In a CAPP environment, have the audit log reside on its own partition. By doing so, you can be sure that the space detection of the audit daemon is accurate and that you do not have other processes consuming this space.

`priority_boost`

Determine how much of a priority boost the audit daemon should get. Possible values are 0 to 20. The resulting nice value calculates like this: 0 - priority_boost

`flush` and `freq`

Specifies whether, how, and how often the audit logs should be written to disk. Valid values for `flush` are `none`, `incremental`, `data`, and `sync`. `none` tells the audit daemon not to make any special effort to write the audit data to disk. `incremental` tells the audit daemon to explicitly flush the data to disk. A frequency must be specified if `incremental` is used. A `freq` value of `20` tells the audit daemon to request that the kernel flush the data to disk after every 20 records. The `data` option keeps the data portion of the disk file synchronized at all times while the `sync` option takes care of both metadata and data.

Note: CAPP Environment

In a CAPP environment, make sure that the audit trail is always fully up to date and complete. Therefore, use `sync` or `data` with the `flush` parameter.

`num_logs`

Specify the number of log files to keep if you have given `rotate` as the `max_log_file_action`. Possible values range from `0` to `99`. A value less than `2` means that the log files are not rotated. As you increase the number of files to rotate, you increase the amount of work required of the audit daemon. While doing this rotation, `auditd` cannot always service new data arriving from the kernel as quickly, which can result in a backlog condition (triggering `auditd` to react according to the failure flag, described in

Section 32.3, "Controlling the Audit System Using **auditctl***"*). In this situation, increasing the backlog limit is recommended. Do so by changing the value of the `-b` parameter in the `/etc/audit/audit.rules` file.

`disp_qos` and `dispatcher`

The dispatcher is started by the audit daemon during its start. The audit daemon relays the audit messages to the application specified in `dispatcher`. This application must be a highly trusted one, because it needs to run as `root`. `disp_qos` determines whether you allow for `lossy` or `lossless` communication between the audit daemon and the dispatcher.

If you select `lossy`, the audit daemon might discard some audit messages when the message queue is full. These events still get written to disk if `log_format` is set to `raw`, but they might not get through to the dispatcher. If you select `lossless` the audit logging to disk is blocked until there is an empty spot in the message queue. The default value is `lossy`.

`name_format` and `name`

`name_format` controls how computer names are resolved. Possible values are `none` (no name will be used), `hostname` (value returned by **gethostname**), `fqd` (fully qualified host name as received through a DNS lookup), `numeric` (IP address) and `user`. `user` is a custom string that needs to be defined with the `name` parameter.

`max_log_file` and `max_log_file_action`

`max_log_file` takes a numerical value that specifies the maximum file size in megabytes that the log file can reach before a configurable action is triggered. The action to be taken is specified in `max_log_file_action`. Possible values for `max_log_file_action` are `ignore`, `syslog`, `suspend`, `rotate`, and `keep_logs`. `ignore` tells the audit daemon to do nothing when the size limit is reached, `syslog` tells it to issue a warning and send it to syslog, and `suspend` causes the audit daemon to stop writing logs to disk, leaving the daemon itself still alive. `rotate` triggers log rotation using the `num_logs` setting. `keep_logs` also triggers log rotation, but does not use the `num_log` setting, so always keeps all logs.

Note: CAPP Environment

To keep a complete audit trail in CAPP environments, the `keep_logs` option should be used. If using a separate partition to hold your audit logs, adjust `max_log_file` and `num_logs` to use the entire space available on that partition. Note that the more files that need to be rotated, the longer it takes to get back to receiving audit events.

`space_left` **and** `space_left_action`

`space_left` takes a numerical value in megabytes of remaining disk space that triggers a configurable action by the audit daemon. The action is specified in `space_left_action`. Possible values for this parameter are `ignore`, `syslog`, `email`, `exec`, `suspend`, `single`, and `halt`. `ignore` tells the audit daemon to ignore the warning and do nothing, `syslog` has it issue a warning to syslog, and `email` sends an e-mail to the account specified under `action_mail_acct`. `exec` plus a path to a script executes the given script. Note that it is not possible to pass parameters to the script. `suspend` tells the audit daemon to stop writing to disk but remain alive while `single` triggers the system to be brought down to single user mode. `halt` triggers a full shutdown of the system.

Note: CAPP Environment

Make sure that `space_left` is set to a value that gives the administrator enough time to react to the alert and allows it to free enough disk space for the audit daemon to continue to work. Freeing disk space would involve calling **`aureport -t`** and archiving the oldest logs on a separate archiving partition or resource. The actual value for `space_left` depends on the size of your deployment. Set `space_left_action` to `email`.

`action_mail_acct`

Specify an e-mail address or alias to which any alert messages should be sent. The default setting is `root`, but you can enter any local or remote account as long as e-mail and the network are properly configured on your system and `/usr/lib/sendmail` exists.

`admin_space_left` **and** `admin_space_left_action`

`admin_space_left` takes a numerical value in megabytes of remaining disk space. The system is already running low on disk space when this limit is reached and the administrator has one last chance to react to this alert and free disk space for the audit logs. The value of `admin_space_left` should be lower than the value for `space_left`. The possible values for `admin_space_left_action` are the same as for `space_left_action`.

Note: CAPP Environment

Set `admin_space_left` to a value that would allow the administrator's actions to be recorded. The action should be set to `single`.

`disk_full_action`

Specify which action to take when the system runs out of disk space for the audit logs. Valid values are `ignore`, `syslog`, `rotate`, `exec`, `suspend`, `single`, and `halt`. For an explanation of these values refer to `space_left` *and* `space_left_action` .

Note: CAPP Environment

As the `disk_full_action` is triggered when there is absolutely no more room for any audit logs, you should bring the system down to single-user mode (`single`) or shut it down completely (`halt`).

`disk_error_action`

Specify which action to take when the audit daemon encounters any kind of disk error while writing the logs to disk or rotating the logs. The possible value are the same as for `space_left_action`.

Note: CAPP Environment

Use `syslog`, `single`, or `halt` depending on your site's policies regarding the handling of any kind of hardware failure.

`tcp_listen_port`, `tcp_listen_queue`, `tcp_client_ports`, `tcp_client_max_idle`, and `tcp_max_per_addr`

The audit daemon can receive audit events from other audit daemons. The TCP parameters let you control incoming connections. Specify a port between 1 and 65535 with `tcp_listen_port` on which the `auditd` will listen. `tcp_listen_queue` lets you

configure a maximum value for pending connections. Make sure not to set a value too small, since the number of pending connections may be high under certain circumstances, such as after a power outage. `tcp_client_ports` defines which client ports are allowed. Either specify a single port or a port range with numbers separated by a dash (for example 1-1023 for all privileged ports).

Specifying a single allowed client port may make it difficult for the client to restart their audit subsystem, as it will be unable to re-create a connection with the same host addresses and ports until the connection closure TIME_WAIT state times out. If a client does not respond anymore, `auditd` complains. Specify the number of seconds after which this will happen with `tcp_client_max_idle`. Keep in mind that this setting is valid for all clients and therefore should be higher than any individual client heartbeat setting, preferably by a factor of two. `tcp_max_per_addr` is a numeric value representing how many concurrent connections from one IP address are allowed.

Tip

We recommend using privileged ports for client and server to prevent non-root (CAP_NET_BIND_SERVICE) programs from binding to those ports.

When the daemon configuration in `/etc/audit/auditd.conf` is complete, the next step is to focus on controlling the amount of auditing the daemon does, and to assign sufficient resources and limits to the daemon so it can operate smoothly.

32.3 Controlling the Audit System Using **auditctl**

`auditctl` is responsible for controlling the status and some basic system parameters of the audit daemon. It controls the amount of auditing performed on the system. Using audit rules, **`auditctl`** controls which components of your system are subjected to the audit and to what extent they are audited. Audit rules can be passed to the audit daemon on the **`auditctl`** command line or by composing a rule set and instructing the audit daemon to process this file. By default, the `auditd` daemon is configured to check for audit rules under `/etc/audit/audit.rules`. For more details on audit rules, refer to *Section 32.4, "Passing Parameters to the Audit System"*.

The main **auditctl** commands to control basic audit system parameters are:

- **auditctl** `-e` to enable or disable audit
- **auditctl** `-f` to control the failure flag
- **auditctl** `-r` to control the rate limit for audit messages
- **auditctl** `-b` to control the backlog limit
- **auditctl** `-s` to query the current status of the audit daemon

Tip

Before running **auditctl -S** on your system, add `-F arch=b64` to prevent the architecture mismatch warning.

The `-e`, `-f`, `-r`, and `-b` options can also be specified in the `audit.rules` file to avoid having to enter them each time the audit daemon is started.

Any time you query the status of the audit daemon with **auditctl** `-s` or change the status flag with **auditctl** `-e`*FLAG*, a status message (including information on each of the above-mentioned parameters) is printed. The following example highlights the typical audit status message.

EXAMPLE 32.1: EXAMPLE OUTPUT OF **auditctl** `-s`

```
AUDIT_STATUS: enabled=1 flag=2 pid=3105 rate_limit=0 backlog_limit=8192 lost=0 backlog=0
```

TABLE 32.1: AUDIT STATUS FLAGS

Flag	Meaning [Possible Values]	Command
`enabled`	Set the enable flag. [0..2] 0 = disable, 1 = enable, 2 = enable and lock down the configuration	**auditctl** `-e [0|1|2]`
`flag`	Set the failure flag. [0..2] 0 = silent, 1 = printk, 2 = panic (immediate halt without synchronizing pending data to disk)	**auditctl** `-f [0|1|2]`

Flag	Meaning [Possible Values]	Command
`pid`	Process ID under which `auditd` is running.	—
`rate_limit`	Set a limit in messages per second. If the rate is not zero and is exceeded, the action specified in the failure flag is triggered.	**`auditctl`** `-r` *RATE*
`backlog_limit`	Specify the maximum number of outstanding audit buffers allowed. If all buffers are full, the action specified in the failure flag is triggered.	**`auditctl`** `-b` *BACKLOG*
`lost`	Count the current number of lost audit messages.	—
`backlog`	Count the current number of outstanding audit buffers.	—

32.4 Passing Parameters to the Audit System

Commands to control the audit system can be invoked individually from the shell using **`auditctl`** or batch read from a file using **`auditctl -`** `R`. This latter method is used by the init scripts to load rules from the file `/etc/audit/audit.rules` after the audit daemon has been started. The rules are executed in order from top to bottom. Each of these rules would expand to a separate **`auditctl`** command. The syntax used in the rules file is the same as that used for the **`auditctl`** command.

Changes made to the running audit system by executing **`auditctl`** on the command line are not persistent across system restarts. For changes to persist, add them to the `/etc/audit/audit.rules` file and, if they are not currently loaded into audit, restart the audit system to load the modified rule set by using the **`systemctl restart auditd`** command.

EXAMPLE 32.2: EXAMPLE AUDIT RULES—AUDIT SYSTEM PARAMETERS

```
-b 1000 ❶
-f 1 ❷
-r 10 ❸
-e 1 ❹
```

❶ Specify the maximum number of outstanding audit buffers. Depending on the level of logging activity, you might need to adjust the number of buffers to avoid causing too heavy an audit load on your system.

❷ Specify the failure flag to use. See *Table 32.1, "Audit Status Flags"* for possible values.

❸ Specify the maximum number of messages per second that may be issued by the kernel. See *Table 32.1, "Audit Status Flags"* for details.

❹ Enable or disable the audit subsystem.

Using audit, you can track any kind of file system access to important files, configurations or resources. You can add watches on these and assign keys to each kind of watch for better identification in the logs.

EXAMPLE 32.3: EXAMPLE AUDIT RULES—FILE SYSTEM AUDITING

```
-w /etc/shadow ❶
-w /etc -p rx ❷
-w /etc/passwd -k fk_passwd -p rwxa ❸
```

❶ The `-w` option tells audit to add a watch to the file specified, in this case `/etc/shadow`. All system calls requesting access permissions to this file are analyzed.

❷ This rule adds a watch to the `/etc` directory and applies permission filtering for read and execute access to this directory (`-p rx`). Any system call requesting any of these two permissions is analyzed. Only the creation of new files and the deletion of existing ones are logged as directory-related events. To get more specific events for files located under this particular directory, you should add a separate rule for each file. A file must exist before you add a rule containing a watch on it. Auditing files as they are created is not supported.

❸ This rule adds a file watch to `/etc/passwd`. Permission filtering is applied for read, write, execute, and attribute change permissions. The `-k` option allows you to specify a key to use to filter the audit logs for this particular event later (for example with **`ausearch`**). You may use the same key on different rules to be able to group rules when searching for them. It is also possible to apply multiple keys to a rule.

System call auditing lets you track your system's behavior on a level even below the application level. When designing these rules, consider that auditing a great many system calls may increase your system load and cause you to run out of disk space. Consider carefully which events need tracking and how they can be filtered to be even more specific.

EXAMPLE 32.4: EXAMPLE AUDIT RULES—SYSTEM CALL AUDITING

```
-a exit,always -S mkdir❶
-a exit,always -S access -F a1=4❷
-a exit,always -S ipc -F a0=2❸
-a exit,always -S open -F success!=0❹
-a task,always -F auid=0❺
-a task,always -F uid=0 -F auid=501 -F gid=wheel❻
```

❶ This rule activates auditing for the `mkdir` system call. The `-a` option adds system call rules. This rule triggers an event whenever the `mkdir` system call is entered (`exit`, `always`). The `-S` option specifies the system call to which this rule should be applied.

❷ This rule adds auditing to the access system call, but only if the second argument of the system call (`mode`) is `4` (`R_OK`). `exit,always` tells audit to add an audit context to this system call when entering it, and to write out a report when it gets audited.

❸ This rule adds an audit context to the IPC multiplexed system call. The specific `ipc` system call is passed as the first syscall argument and can be selected using `-F a0=`*`IPC_CALL_NUMBER`*.

❹ This rule audits failed attempts to call open.

❺ This rule is an example of a task rule (keyword: `task`). It is different from the other rules above in that it applies to processes that are forked or cloned. To filter these kind of events, you can only use fields that are known at fork time, such as UID, GID, and AUID. This example rule filters for all tasks carrying an audit ID of `0`.

❻ This last rule makes heavy use of filters. All filter options are combined with a logical AND operator, meaning that this rule applies to all tasks that carry the audit ID of `501`, run as `root`, and have `wheel` as the group. A process is given an audit ID on user login. This ID is then handed down to any child process started by the initial process of the user. Even if the user changes their identity, the audit ID stays the same and allows tracing actions to the original user.

Tip: Filtering System Call Arguments

For more details on filtering system call arguments, refer to *Section 34.6, "Filtering System Call Arguments"*.

You cannot only add rules to the audit system, but also remove them. There are different methods for deleting the entire rule set at once or for deleting system call rules or file and directory watches:

EXAMPLE 32.5: DELETING AUDIT RULES AND EVENTS

```
-D ❶
-d exit,always -S mkdir ❷
-W /etc ❸
```

❶ Clear the queue of audit rules and delete any preexisting rules. This rule is used as the first rule in `/etc/audit/audit.rules` files to make sure that the rules that are about to be added do not clash with any preexisting ones. The **`auditctl`** `-D` command is also used before doing an **`autrace`** to avoid having the trace rules clash with any rules present in the `audit.rules` file.

❷ This rule deletes a system call rule. The `-d` option must precede any system call rule that needs to be deleted from the rule queue, and must match exactly.

❸ This rule tells audit to discard the rule with the directory watch on `/etc` from the rules queue. This rule deletes any rule containing a directory watch on `/etc`, regardless of any permission filtering or key options.

To get an overview of which rules are currently in use in your audit setup, run **`auditctl`** `-l`. This command displays all rules with one rule per line.

EXAMPLE 32.6: LISTING RULES WITH **`auditctl`** `-l`

```
exit,always watch=/etc perm=rx
exit,always watch=/etc/passwd perm=rwxa key=fk_passwd
exit,always watch=/etc/shadow perm=rwxa
exit,always syscall=mkdir
exit,always a1=4 (0x4) syscall=access
exit,always a0=2 (0x2) syscall=ipc
exit,always success!=0 syscall=open
```

Note: Creating Filter Rules

You can build very sophisticated audit rules by using the various filter options. Refer to the **auditctl(8)** man page for more information about the options available for building audit filter rules, and audit rules in general.

32.5 Understanding the Audit Logs and Generating Reports

To understand what the **aureport** utility does, it is vital to know how the logs generated by the audit daemon are structured, and what exactly is recorded for an event. Only then can you decide which report types are most appropriate for your needs.

32.5.1 Understanding the Audit Logs

The following examples highlight two typical events that are logged by audit and how their trails in the audit log are read. The audit log or logs (if log rotation is enabled) are stored in the `/var/log/audit` directory. The first example is a simple **less** command. The second example covers a great deal of PAM activity in the logs when a user tries to remotely log in to a machine running audit.

EXAMPLE 32.7: A SIMPLE AUDIT EVENT—VIEWING THE AUDIT LOG

```
type=SYSCALL msg=audit(1234874638.599:5207): arch=c000003e syscall=2 success=yes exit=4
 a0=62fb60 a1=0 a2=31 a3=0 items=1 ppid=25400 pid
=25616 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts1 ses=1164
 comm="less" exe="/usr/bin/less" key="doc_log"
type=CWD msg=audit(1234874638.599:5207):  cwd="/root"
type=PATH msg=audit(1234874638.599:5207): item=0 name="/var/log/audit/audit.log"
 inode=1219041 dev=08:06 mode=0100644 ouid=0 ogid=0 rdev=00:00
```

The above event, a simple **less /var/log/audit/audit.log**, wrote three messages to the log. All of them are closely linked together and you would not be able to make sense of one of them without the others. The first message reveals the following information:

`type`

The type of event recorded. In this case, it assigns the `SYSCALL` type to an event triggered by a system call. The `CWD` event was recorded to record the current working directory at the time of the syscall. A `PATH` event is generated for each path passed to the system call. The open system call takes only one path argument, so only generates one `PATH` event. It is important to understand that the `PATH` event reports the path name string argument without any further interpretation, so a relative path requires manual combination with the path reported by the `CWD` event to determine the object accessed.

`msg`

A message ID enclosed in brackets. The ID splits into two parts. All characters before the `:` represent a Unix epoch time stamp. The number after the colon represents the actual event ID. All events that are logged from one application's system call have the same event ID. If the application makes a second system call, it gets another event ID.

`arch`

References the CPU architecture of the system call. Decode this information using the `-i` option on any of your **`ausearch`** commands when searching the logs.

`syscall`

The type of system call as it would have been printed by an **`strace`** on this particular system call. This data is taken from the list of system calls under `/usr/include/asm/unistd.h` and may vary depending on the architecture. In this case, `syscall=2` refers to the open system call (see **`man open(2)`**) invoked by the less application.

`success`

Whether the system call succeeded or failed.

`exit`

The exit value returned by the system call. For the **`open`** system call used in this example, this is the file descriptor number. This varies by system call.

`a0` to `a3`

The first four arguments to the system call in numeric form. The values of these are system call dependent. In this example (an **`open`** system call), the following are used:

```
a0=62fb60 a1=8000 a2=31 a3=0
```

`a0` is the start address of the passed path name. `a1` is the flags. `8000` in hex notation translates to `100000` in octal notation, which in turn translates to `O_LARGEFILE`. `a2` is the mode, which, because `O_CREAT` was not specified, is unused. `a3` is not passed by the **`open`** system call. Check the manual page of the relevant system call to find out which arguments are used with it.

`items`
: The number of strings passed to the application.

`ppid`
: The process ID of the parent of the process analyzed.

`pid`
: The process ID of the process analyzed.

`auid`
: The audit ID. A process is given an audit ID on user login. This ID is then handed down to any child process started by the initial process of the user. Even if the user changes their identity (for example, becomes `root`), the audit ID stays the same. Thus you can always trace actions to the original user who logged in.

`uid`
: The user ID of the user who started the process. In this case, `0` for `root`.

`gid`
: The group ID of the user who started the process. In this case, `0` for `root`.

`euid`, `suid`, `fsuid`
: Effective user ID, set user ID, and file system user ID of the user that started the process.

`egid`, `sgid`, `fsgid`
: Effective group ID, set group ID, and file system group ID of the user that started the process.

`tty`
: The terminal from which the application was started. In this case, a pseudo-terminal used in an SSH session.

`ses`
: The login session ID. This process attribute is set when a user logs in and can tie any process to a particular user login.

`comm`

The application name under which it appears in the task list.

`exe`

The resolved path name to the binary program.

`subj`

`auditd` records whether the process is subject to any security context, such as AppArmor. `unconstrained`, as in this case, means that the process is not confined with AppArmor. If the process had been confined, the binary path name plus the AppArmor profile mode would have been logged.

`key`

If you are auditing many directories or files, assign key strings to each of these watches. You can use these keys with **`ausearch`** to search the logs for events of this type only.

The second message triggered by the example **`less`** call does not reveal anything apart from the current working directory when the **`less`** command was executed.

The third message reveals the following (the `type` and `message` flags have already been introduced):

`item`

In this example, `item` references the `a0` argument—a path—that is associated with the original `SYSCALL` message. Had the original call had more than one path argument (such as a **`cp`** or **`mv`** command), an additional `PATH` event would have been logged for the second path argument.

`name`

Refers to the path name passed as an argument to the open system call.

`inode`

Refers to the inode number corresponding to `name`.

`dev`

Specifies the device on which the file is stored. In this case, `08:06`, which stands for `/dev/sda1` or “first partition on the first IDE device.”

`mode`

Numerical representation of the file's access permissions. In this case, `root` has read and write permissions and their group (`root`) has read access while the entire rest of the world cannot access the file.

`ouid` and `ogid`

Refer to the UID and GID of the inode itself.

`rdev`

Not applicable for this example. The `rdev` entry only applies to block or character devices, not to files.

Example 32.8, "An Advanced Audit Event—Login via SSH" highlights the audit events triggered by an incoming SSH connection. Most of the messages are related to the PAM stack and reflect the different stages of the SSH PAM process. Several of the audit messages carry nested PAM messages in them that signify that a particular stage of the PAM process has been reached. Although the PAM messages are logged by audit, audit assigns its own message type to each event:

EXAMPLE 32.8: AN ADVANCED AUDIT EVENT—LOGIN VIA SSH

```
type=USER_AUTH msg=audit(1234877011.791:7731): user pid=26127 uid=0 ❶
auid=4294967295 ses=4294967295 msg='op=PAM:authentication acct="root" exe="/usr/sbin/
sshd"
(hostname=jupiter.example.com, addr=192.168.2.100, terminal=ssh res=success)'
type=USER_ACCT msg=audit(1234877011.795:7732): user pid=26127 uid=0 ❷
auid=4294967295 ses=4294967295 msg='op=PAM:accounting acct="root" exe="/usr/sbin/sshd"
(hostname=jupiter.example.com, addr=192.168.2.100, terminal=ssh res=success)'
type=CRED_ACQ msg=audit(1234877011.799:7733): user pid=26125 uid=0 ❸
auid=4294967295 ses=4294967295 msg='op=PAM:setcred acct="root" exe="/usr/sbin/sshd"
(hostname=jupiter.example.com, addr=192.168.2.100, terminal=/dev/pts/0 res=success)'
type=LOGIN msg=audit(1234877011.799:7734): login pid=26125 uid=0
old auid=4294967295 new auid=0 old ses=4294967295 new ses=1172
type=USER_START msg=audit(1234877011.799:7735): user pid=26125 uid=0 ❹
auid=0 ses=1172 msg='op=PAM:session_open acct="root" exe="/usr/sbin/sshd"
(hostname=jupiter.example.com, addr=192.168.2.100, terminal=/dev/pts/0 res=success)'
type=USER_LOGIN msg=audit(1234877011.823:7736): user pid=26128 uid=0 ❺
auid=0 ses=1172 msg='uid=0: exe="/usr/sbin/sshd"
(hostname=jupiter.example.com, addr=192.168.2.100, terminal=/dev/pts/0 res=success)'
type=CRED_REFR msg=audit(1234877011.828:7737): user pid=26128 uid=0 ❻
auid=0 ses=1172 msg='op=PAM:setcred acct="root" exe="/usr/sbin/sshd"
(hostname=jupiter.example.com, addr=192.168.2.100, terminal=/dev/pts/0 res=success)'
```

❶ PAM reports that is has successfully requested user authentication for `root` from a remote host (jupiter.example.com, 192.168.2.100). The terminal where this is happening is `ssh`.

❷ PAM reports that it has successfully determined whether the user is authorized to log in.

❸ PAM reports that the appropriate credentials to log in have been acquired and that the terminal changed to a normal terminal (`/dev/pts0`).

4. PAM reports that it has successfully opened a session for `root`.
5. The user has successfully logged in. This event is the one used by **`aureport`** `-l` to report about user logins.
6. PAM reports that the credentials have been successfully reacquired.

32.5.2 Generating Custom Audit Reports

The raw audit reports stored in the `/var/log/audit` directory tend to become very bulky and hard to understand. To more easily find relevant messages, use the **`aureport`** utility and create custom reports.

The following use cases highlight a few of the possible report types that you can generate with **`aureport`**:

Read Audit Logs from Another File

When the audit logs have moved to another machine or when you want to analyze the logs of several machines on your local machine without wanting to connect to each of these individually, move the logs to a local file and have **`aureport`** analyze them locally:

```
tux > sudo aureport -if myfile

Summary Report
======================
Range of time in logs: 03/02/09 14:13:38.225 - 17/02/09 14:52:27.971
Selected time for report: 03/02/09 14:13:38 - 17/02/09 14:52:27.971
Number of changes in configuration: 13
Number of changes to accounts, groups, or roles: 0
Number of logins: 6
Number of failed logins: 13
Number of authentications: 7
Number of failed authentications: 573
Number of users: 1
Number of terminals: 9
Number of host names: 4
Number of executables: 17
Number of files: 279
Number of AVC's: 0
Number of MAC events: 0
Number of failed syscalls: 994
Number of anomaly events: 0
Number of responses to anomaly events: 0
Number of crypto events: 0
```

```
Number of keys: 2
Number of process IDs: 1211
Number of events: 5320
```

The above command, **aureport** without any arguments, provides only the standard general summary report generated from the logs contained in `myfile`. To create more detailed reports, combine the `-if` option with any of the options below. For example, generate a login report that is limited to a certain time frame:

```
tux > sudo aureport -l -ts 14:00 -te 15:00 -if myfile

Login Report
============================================
# date time auid host term exe success event
============================================
1. 17/02/09 14:21:09 root: 192.168.2.100 sshd /usr/sbin/sshd no 7718
2. 17/02/09 14:21:15 0 jupiter /dev/pts/3 /usr/sbin/sshd yes 7724
```

Convert Numeric Entities to Text

Some information, such as user IDs, are printed in numeric form. To convert these into a human-readable text format, add the `-i` option to your **aureport** command.

Create a Rough Summary Report

If you are interested in the current audit statistics (events, logins, processes, etc.), run **aureport** without any other option.

Create a Summary Report of Failed Events

If you want to break down the overall statistics of plain **aureport** to the statistics of failed events, use **aureport** `--failed`:

```
tux > sudo aureport --failed

Failed Summary Report
======================
Range of time in logs: 03/02/09 14:13:38.225 - 17/02/09 14:57:35.183
Selected time for report: 03/02/09 14:13:38 - 17/02/09 14:57:35.183
Number of changes in configuration: 0
Number of changes to accounts, groups, or roles: 0
Number of logins: 0
Number of failed logins: 13
Number of authentications: 0
Number of failed authentications: 574
Number of users: 1
Number of terminals: 5
```

```
Number of host names: 4
Number of executables: 11
Number of files: 77
Number of AVC's: 0
Number of MAC events: 0
Number of failed syscalls: 994
Number of anomaly events: 0
Number of responses to anomaly events: 0
Number of crypto events: 0
Number of keys: 2
Number of process IDs: 708
Number of events: 1583
```

Create a Summary Report of Successful Events

If you want to break down the overall statistics of a plain **aureport** to the statistics of successful events, use **aureport** `--success`:

```
tux > sudo aureport --success

Success Summary Report
======================
Range of time in logs: 03/02/09 14:13:38.225 - 17/02/09 15:00:01.535
Selected time for report: 03/02/09 14:13:38 - 17/02/09 15:00:01.535
Number of changes in configuration: 13
Number of changes to accounts, groups, or roles: 0
Number of logins: 6
Number of failed logins: 0
Number of authentications: 7
Number of failed authentications: 0
Number of users: 1
Number of terminals: 7
Number of host names: 3
Number of executables: 16
Number of files: 215
Number of AVC's: 0
Number of MAC events: 0
Number of failed syscalls: 0
Number of anomaly events: 0
Number of responses to anomaly events: 0
Number of crypto events: 0
Number of keys: 2
Number of process IDs: 558
Number of events: 3739
```

Create Summary Reports

In addition to the dedicated summary reports (main summary and failed and success summary), use the `--summary` option with most of the other options to create summary reports for a particular area of interest only. Not all reports support this option, however. This example creates a summary report for user login events:

```
tux > sudo aureport -u -i --summary

User Summary Report
===========================
total  auid
===========================
5640  root
13  tux
3  wilber
```

Create a Report of Events

To get an overview of the events logged by audit, use the **`aureport`** `-e` command. This command generates a numbered list of all events including date, time, event number, event type, and audit ID.

```
tux > sudo aureport -e -ts 14:00 -te 14:21

Event Report
===================================
# date time event type auid success
===================================
1. 17/02/09 14:20:27 7462 DAEMON_START 0 yes
2. 17/02/09 14:20:27 7715 CONFIG_CHANGE 0 yes
3. 17/02/09 14:20:57 7716 USER_END 0 yes
4. 17/02/09 14:20:57 7717 CRED_DISP 0 yes
5. 17/02/09 14:21:09 7718 USER_LOGIN -1 no
6. 17/02/09 14:21:15 7719 USER_AUTH -1 yes
7. 17/02/09 14:21:15 7720 USER_ACCT -1 yes
8. 17/02/09 14:21:15 7721 CRED_ACQ -1 yes
9. 17/02/09 14:21:15 7722 LOGIN 0 yes
10. 17/02/09 14:21:15 7723 USER_START 0 yes
11. 17/02/09 14:21:15 7724 USER_LOGIN 0 yes
12. 17/02/09 14:21:15 7725 CRED_REFR 0 yes
```

Create a Report from All Process Events

To analyze the log from a process's point of view, use the **`aureport`** `-p` command. This command generates a numbered list of all process events including date, time, process ID, name of the executable, system call, audit ID, and event number.

```
aureport -p
```

```
Process ID Report
====================================
# date time pid exe syscall auid event
====================================
1. 13/02/09 15:30:01 32742 /usr/sbin/cron 0 0 35
2. 13/02/09 15:30:01 32742 /usr/sbin/cron 0 0 36
3. 13/02/09 15:38:34 32734 /usr/lib/gdm/gdm-session-worker 0 -1 37
```

Create a Report from All System Call Events

To analyze the audit log from a system call's point of view, use the **aureport** `-s` command. This command generates a numbered list of all system call events including date, time, number of the system call, process ID, name of the command that used this call, audit ID, and event number.

```
tux > sudo aureport -s

Syscall Report
=====================================
# date time syscall pid comm auid event
=====================================
1. 16/02/09 17:45:01 2 20343 cron -1 2279
2. 16/02/09 17:45:02 83 20350 mktemp 0 2284
3. 16/02/09 17:45:02 83 20351 mkdir 0 2285
```

Create a Report from All Executable Events

To analyze the audit log from an executable's point of view, use the **aureport** `-x` command. This command generates a numbered list of all executable events including date, time, name of the executable, the terminal it is run in, the host executing it, the audit ID, and event number.

```
aureport -x

Executable Report
===================================
# date time exe term host auid event
===================================
1. 13/02/09 15:08:26 /usr/sbin/sshd sshd 192.168.2.100 -1 12
2. 13/02/09 15:08:28 /usr/lib/gdm/gdm-session-worker :0 ? -1 13
3. 13/02/09 15:08:28 /usr/sbin/sshd ssh 192.168.2.100 -1 14
```

Create a Report about Files

To generate a report from the audit log that focuses on file access, use the **aureport** `-f` command. This command generates a numbered list of all file-related events including date, time, name of the accessed file, number of the system call accessing it, success or failure of the command, the executable accessing the file, audit ID, and event number.

```
tux > sudo aureport -f

File Report
===============================================
# date time file syscall success exe auid event
===============================================
1. 16/02/09 17:45:01 /etc/shadow 2 yes /usr/sbin/cron -1 2279
2. 16/02/09 17:45:02 /tmp/ 83 yes /bin/mktemp 0 2284
3. 16/02/09 17:45:02 /var 83 no /bin/mkdir 0 2285
```

Create a Report about Users

To generate a report from the audit log that illustrates which users are running what executables on your system, use the **aureport** `-u` command. This command generates a numbered list of all user-related events including date, time, audit ID, terminal used, host, name of the executable, and an event ID.

```
aureport -u

User ID Report
===================================
# date time auid term host exe event
===================================
1. 13/02/09 15:08:26 -1 sshd 192.168.2.100 /usr/sbin/sshd 12
2. 13/02/09 15:08:28 -1 :0 ? /usr/lib/gdm/gdm-session-worker 13
3. 14/02/09 08:25:39 -1 ssh 192.168.2.101 /usr/sbin/sshd 14
```

Create a Report about Logins

To create a report that focuses on login attempts to your machine, run the **aureport** `-l` command. This command generates a numbered list of all login-related events including date, time, audit ID, host and terminal used, name of the executable, success or failure of the attempt, and an event ID.

```
tux > sudo aureport -l -i

Login Report
============================================
# date time auid host term exe success event
============================================
```

```
1. 13/02/09 15:08:31 tux: 192.168.2.100 sshd /usr/sbin/sshd no 19
2. 16/02/09 12:39:05 root: 192.168.2.101 sshd /usr/sbin/sshd no 2108
3. 17/02/09 15:29:07 geeko: ? tty3 /bin/login yes 7809
```

Limit a Report to a Certain Time Frame

To analyze the logs for a particular time frame, such as only the working hours of Feb 16, 2009, first find out whether this data is contained in the current `audit.log` or whether the logs have been rotated in by running **`aureport`** `-t`:

```
aureport -t

Log Time Range Report
=====================
/var/log/audit/audit.log: 03/02/09 14:13:38.225 - 17/02/09 15:30:01.636
```

The current `audit.log` contains all the desired data. Otherwise, use the `-if` option to point the **`aureport`** commands to the log file that contains the needed data.
Then, specify the start date and time and the end date and time of the desired time frame and combine it with the report option needed. This example focuses on login attempts:

```
tux > sudo aureport -ts 02/16/09 8:00 -te 02/16/09 18:00 -l

Login Report
============================================
# date time auid host term exe success event
============================================
1. 16/02/09 12:39:05 root: 192.168.2.100 sshd /usr/sbin/sshd no 2108
2. 16/02/09 12:39:12 0 192.168.2.100 /dev/pts/1 /usr/sbin/sshd yes 2114
3. 16/02/09 13:09:28 root: 192.168.2.100 sshd /usr/sbin/sshd no 2131
4. 16/02/09 13:09:32 root: 192.168.2.100 sshd /usr/sbin/sshd no 2133
5. 16/02/09 13:09:37 0 192.168.2.100 /dev/pts/2 /usr/sbin/sshd yes 2139
```

The start date and time are specified with the `-ts` option. Any event that has a time stamp equal to or after your given start time appears in the report. If you omit the date, **`aureport`** assumes that you meant *today*. If you omit the time, it assumes that the start time should be midnight of the date specified.
Specify the end date and time with the `-te` option. Any event that has a time stamp equal to or before your given event time appears in the report. If you omit the date, **`aureport`** assumes that you meant today. If you omit the time, it assumes that the end time should be now. Use the same format for the date and time as for `-ts`.

All reports except the summary ones are printed in column format and sent to STDOUT, which means that this data can be written to other commands very easily. The visualization scripts introduced in *Section 32.8, "Visualizing Audit Data"* are examples of how to further process the data generated by audit.

32.6 Querying the Audit Daemon Logs with `ausearch`

The `aureport` tool helps you to create overall summaries of what is happening on the system, but if you are interested in the details of a particular event, `ausearch` is the tool to use.

`ausearch` allows you to search the audit logs using special keys and search phrases that relate to most of the flags that appear in event messages in `/var/log/audit/audit.log`. Not all record types contain the same search phrases. There are no `hostname` or `uid` entries in a `PATH` record, for example.

When searching, make sure that you choose appropriate search criteria to catch all records you need. On the other hand, you could be searching for a specific type of record and still get various other related records along with it. This is caused by different parts of the kernel contributing additional records for events that are related to the one to find. For example, you would always get a `PATH` record along with the `SYSCALL` record for an `open` system call.

Tip: Using Multiple Search Options

Any of the command line options can be combined with logical AND operators to narrow down your search.

Read Audit Logs from Another File

When the audit logs have moved to another machine or when you want to analyze the logs of several machines on your local machine without wanting to connect to each of these individually, move the logs to a local file and have `ausearch` search them locally:

```
tux > sudo ausearch - option -if myfile
```

Convert Numeric Results into Text

Some information, such as user IDs are printed in numeric form. To convert these into human readable text format, add the `-i` option to your `ausearch` command.

Search by Audit Event ID

If you have previously run an audit report or done an **autrace**, you should analyze the trail of a particular event in the log. Most of the report types described in *Section 32.5, "Understanding the Audit Logs and Generating Reports"* include audit event IDs in their output. An audit event ID is the second part of an audit message ID, which consists of a Unix epoch time stamp and the audit event ID separated by a colon. All events that are logged from one application's system call have the same event ID. Use this event ID with **ausearch** to retrieve this event's trail from the log.

Use a command similar to the following:

```
tux > sudo ausearch -a 5207
----
time->Tue Feb 17 13:43:58 2009
type=PATH msg=audit(1234874638.599:5207): item=0 name="/var/log/audit/audit.log"
 inode=1219041 dev=08:06 mode=0100644 ouid=0 ogid=0 rdev=00:00
type=CWD msg=audit(1234874638.599:5207):  cwd="/root"
type=SYSCALL msg=audit(1234874638.599:5207): arch=c000003e syscall=2 success=yes
 exit=4 a0=62fb60 a1=0 a2=31 a3=0 items=1 ppid=25400 pid=25616 auid=0 uid=0 gid=0
 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts1 ses=1164 comm="less" exe="/
usr/bin/less" key="doc_log"
```

The **ausearch** `-a` command grabs all records in the logs that are related to the audit event ID provided and displays them. This option can be combined with any other option.

Search by Message Type

To search for audit records of a particular message type, use the **ausearch** `-m` *`MESSAGE_TYPE`* command. Examples of valid message types include `PATH`, `SYSCALL`, and `USER_LOGIN`. Running **ausearch** `-m` without a message type displays a list of all message types.

Search by Login ID

To view records associated with a particular login user ID, use the **ausearch** `-ul` command. It displays any records related to the user login ID specified provided that user had been able to log in successfully.

Search by User ID

View records related to any of the user IDs (both user ID and effective user ID) with **ausearch** `-ua`. View reports related to a particular user ID with **ausearch** `-ui` *`UID`*. Search for records related to a particular effective user ID, use the **ausearch** `-ue` *`EUID`*. Searching for a user ID means the user ID of the user creating a process. Searching for an effective user ID means the user ID and privileges that are required to run this process.

Search by Group ID

View records related to any of the group IDs (both group ID and effective group ID) with the **ausearch** `-ga` command. View reports related to a particular user ID with **ausearch** `-gi GID`. Search for records related to a particular effective group ID, use **ausearch** `-ge EGID`.

Search by Command Line Name

View records related to a certain command, using the **ausearch** `-c COMM_NAME` command, for example, **ausearch** `-c less` for all records related to the **less** command.

Search by Executable Name

View records related to a certain executable with the **ausearch** `-x EXE` command, for example **ausearch** `-x /usr/bin/less` for all records related to the **/usr/bin/less** executable.

Search by System Call Name

View records related to a certain system call with the **ausearch** `-sc SYSCALL` command, for example, **ausearch -sc open** for all records related to the **open** system call.

Search by Process ID

View records related to a certain process ID with the **ausearch** `-p PID` command, for example **ausearch** `-p 13368` for all records related to this process ID.

Search by Event or System Call Success Value

View records containing a certain system call success value with **ausearch** `-sv SUCCESS_VALUE`, for example, **ausearch** `-sv yes` for all successful system calls.

Search by File Name

View records containing a certain file name with **ausearch** `-f FILE_NAME`, for example, **ausearch** `-f /foo/bar` for all records related to the `/foo/bar` file. Using the file name alone would work as well, but using relative paths does not work.

Search by Terminal

View records of events related to a certain terminal only with **ausearch** `-tm TERM`, for example, **ausearch** `-tm ssh` to view all records related to events on the SSH terminal and **ausearch** `-tm tty` to view all events related to the console.

Search by Host Name

View records related to a certain remote host name with **ausearch** `-hn HOSTNAME`, for example, **ausearch** `-hn jupiter.example.com`. You can use a host name, fully qualified domain name, or numeric network address.

Search by Key Field

View records that contain a certain key assigned in the audit rule set to identify events of a particular type. Use the **ausearch** `-k KEY_FIELD`, for example, **ausearch** `-k CFG_etc` to display any records containing the `CFG_etc` key.

Search by Word

View records that contain a certain string assigned in the audit rule set to identify events of a particular type. The whole string will be matched on file name, host name, and terminal. Use the **ausearch** `-w WORD`.

Limit a Search to a Certain Time Frame

Use `-ts` and `-te` to limit the scope of your searches to a certain time frame. The `-ts` option is used to specify the start date and time and the `-te` option is used to specify the end date and time. These options can be combined with any of the above. The use of these options is similar to use with **aureport**.

32.7 Analyzing Processes with **autrace**

In addition to monitoring your system using the rules you set up, you can also perform dedicated audits of individual processes using the **autrace** command. **autrace** works similarly to the **strace** command, but gathers slightly different information. The output of **autrace** is written to `/var/log/audit/audit.log` and does not look any different from the standard audit log entries.

When performing an **autrace** on a process, make sure that any audit rules are purged from the queue to avoid these rules clashing with the ones **autrace** adds itself. Delete the audit rules with the **auditctl** `-D` command. This stops all normal auditing.

```
tux > sudo auditctl -D

No rules

autrace /usr/bin/less

Waiting to execute: /usr/bin/less
Cleaning up...
No rules
Trace complete. You can locate the records with 'ausearch -i -p 7642'
```

Always use the full path to the executable to track with **autrace**. After the trace is complete, **autrace** provides the event ID of the trace, so you can analyze the entire data trail with **ausearch**. To restore the audit system to use the audit rule set again, restart the audit daemon with **systemctl restart auditd**.

32.8 Visualizing Audit Data

Neither the data trail in `/var/log/audit/audit.log` nor the different report types generated by **aureport**, described in *Section 32.5.2, "Generating Custom Audit Reports"*, provide an intuitive reading experience to the user. The **aureport** output is formatted in columns and thus easily available to any sed, Perl, or awk scripts that users might connect to the audit framework to visualize the audit data.

The visualization scripts (see *Section 33.6, "Configuring Log Visualization"*) are one example of how to use standard Linux tools available with openSUSE Leap or any other Linux distribution to create easy-to-read audit output. The following examples help you understand how the plain audit reports can be transformed into human readable graphics.

The first example illustrates the relationship of programs and system calls. To get to this kind of data, you need to determine the appropriate **aureport** command that delivers the source data from which to generate the final graphic:

```
tux > sudo aureport -s -i

Syscall Report
=======================================
# date time syscall pid comm auid event
=======================================
1. 16/02/09 17:45:01 open 20343 cron unset 2279
2. 16/02/09 17:45:02 mkdir 20350 mktemp root 2284
3. 16/02/09 17:45:02 mkdir 20351 mkdir root 2285
...
```

The first thing that the visualization script needs to do on this report is to extract only those columns that are of interest, in this example, the `syscall` and the `comm` columns. The output is sorted and duplicates removed then the final output is written into the visualization program itself:

```
LC_ALL=C aureport -s -i | awk '/^[0-9]/ { print $6" "$4 }' | sort | uniq | mkgraph
```

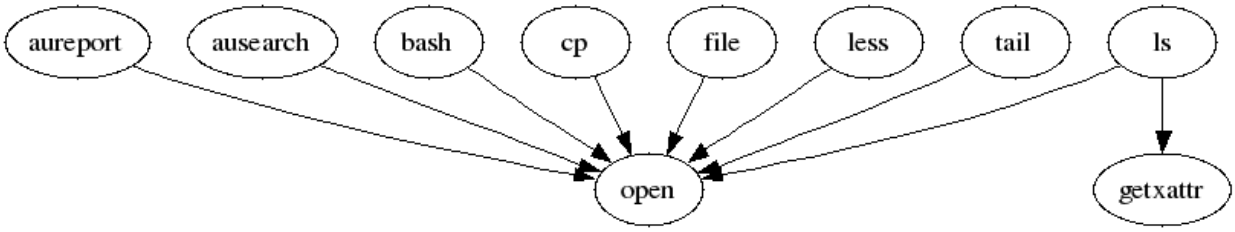


FIGURE 32.2: FLOW GRAPH—PROGRAM VERSUS SYSTEM CALL RELATIONSHIP

The second example illustrates the different types of events and how many of each type have been logged. The appropriate **aureport** command to extract this kind of information is **aureport -e**:

```
tux > sudo aureport -e -i --summary

Event Summary Report
======================
total  type
======================
2434  SYSCALL
816  USER_START
816  USER_ACCT
814  CRED_ACQ
810  LOGIN
806  CRED_DISP
779  USER_END
99  CONFIG_CHANGE
52  USER_LOGIN
```

Because this type of report already contains a two column output, it is only fed into the visualization script and transformed into a bar chart.

```
tux > sudo aureport -e -i --summary  | mkbar events
```

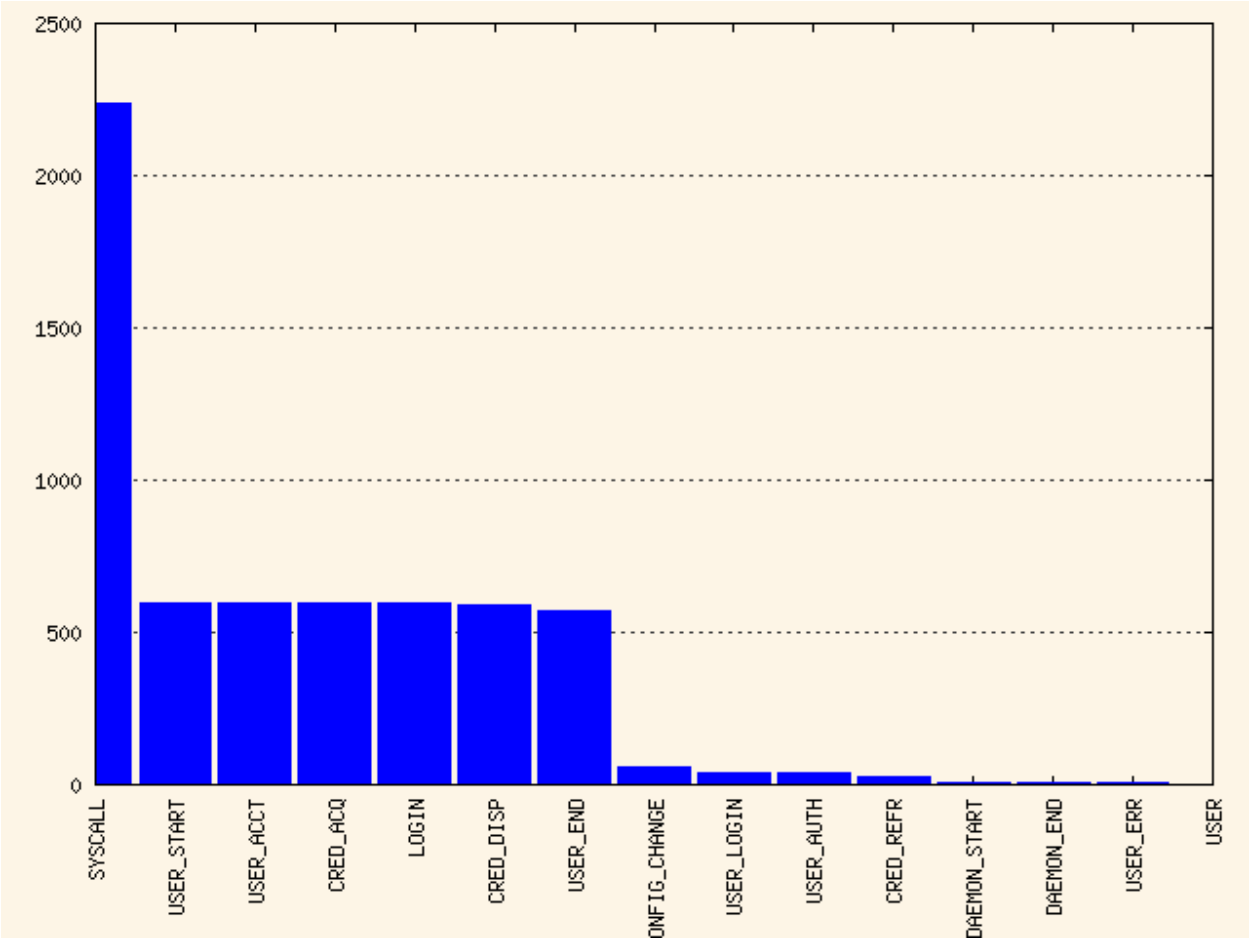


FIGURE 32.3: BAR CHART—COMMON EVENT TYPES

For background information about the visualization of audit data, refer to the Web site of the audit project at http://people.redhat.com/sgrubb/audit/visualize/index.html.

32.9 Relaying Audit Event Notifications

The auditing system also allows external applications to access and use the `auditd` daemon in real time. This feature is provided by so called *audit dispatcher* which allows, for example, intrusion detection systems to use `auditd` to receive enhanced detection information.

`audispd` is a daemon which controls the audit dispatcher. It is normally started by `auditd`. `audispd` takes audit events and distributes them to the programs which want to analyze them in real time. Configuration of `auditd` is stored in `/etc/audisp/audispd.conf`. The file has the following options:

`q_depth`

Specifies the size of the event dispatcher internal queue. If syslog complains about audit events getting dropped, increase this value. Default is 80.

`overflow_action`

Specifies the way the audit daemon will react to the internal queue overflow. Possible values are `ignore` (nothing happens), `syslog` (issues a warning to syslog), `suspend` (audispd will stop processing events), `single` (the computer system will be put in single user mode), or `halt` (shuts the system down).

`priority_boost`

Specifies the priority for the audit event dispatcher (in addition to the audit daemon priority itself). Default is 4 which means no change in priority.

`name_format`

Specifies the way the computer node name is inserted into the audit event. Possible values are `none` (no computer name is inserted), `hostname` (name returned by the `gethostname` system call), `fqd` (fully qualified domain name of the machine), `numeric` (IP address of the machine), or `user` (user defined string from the `name` option). Default is `none`.

`name`

Specifies a user defined string which identifies the machine. The `name_format` option must be set to `user`, otherwise this option is ignored.

`max_restarts`

A non-negative number that tells the audit event dispatcher how many times it can try to restart a crashed plug-in. The default is 10.

EXAMPLE 32.9: EXAMPLE /ETC/AUDISP/AUDISPD.CONF

```
q_depth = 80
overflow_action = SYSLOG
priority_boost = 4
name_format = HOSTNAME
#name = mydomain
```

The plug-in programs install their configuration files in a special directory dedicated to `audispd` plug-ins. It is `/etc/audisp/plugins.d` by default. The plug-in configuration files have the following options:

`active`

Specifies if the program will use `audispd`. Possible values are `yes` or `no`.

`direction`

Specifies the way the plug-in was designed to communicate with audit. It informs the event dispatcher in which directions the events flow. Possible values are `in` or `out`.

`path`

Specifies the absolute path to the plug-in executable. In case of internal plug-ins, this option specifies the plug-in name.

`type`

Specifies the way the plug-in is to be run. Possible values are `builtin` or `always`. Use `builtin` for internal plug-ins (`af_unix` and `syslog`) and `always` for most (if not all) other plug-ins. Default is `always`.

`args`

Specifies the argument that is passed to the plug-in program. Normally, plug-in programs read their arguments from their configuration file and do not need to receive any arguments. There is a limit of two arguments.

`format`

Specifies the format of data that the audit dispatcher passes to the plug-in program. Valid options are `binary` or `string`. `binary` passes the data exactly as the event dispatcher receives them from the audit daemon. `string` instructs the dispatcher to change the event into a string that is parseable by the audit parsing library. Default is `string`.

EXAMPLE 32.10: EXAMPLE /ETC/AUDISP/PLUGINS.D/SYSLOG.CONF

```
active = no
direction = out
path = builtin_syslog
type = builtin
args = LOG_INFO
format = string
```

33 Setting Up the Linux Audit Framework

This chapter shows how to set up a simple audit scenario. Every step involved in configuring and enabling audit is explained in detail. After you have learned to set up audit, consider a real-world example scenario in *Chapter 34, Introducing an Audit Rule Set*.

To set up audit on openSUSE Leap, you need to complete the following steps:

PROCEDURE 33.1: SETTING UP THE LINUX AUDIT FRAMEWORK

1. Make sure that all required packages are installed: `audit`, `audit-libs`, and optionally `audit-libs-python`. To use the log visualization as described in *Section 33.6, "Configuring Log Visualization"*, install `gnuplot` and `graphviz` from the openSUSE Leap media.
2. Determine the components to audit. Refer to *Section 33.1, "Determining the Components to Audit"* for details.
3. Check or modify the basic audit daemon configuration. Refer to *Section 33.2, "Configuring the Audit Daemon"* for details.
4. Enable auditing for system calls. Refer to *Section 33.3, "Enabling Audit for System Calls"* for details.
5. Compose audit rules to suit your scenario. Refer to *Section 33.4, "Setting Up Audit Rules"* for details.
6. Generate logs and configure tailor-made reports. Refer to *Section 33.5, "Configuring Audit Reports"* for details.
7. Configure optional log visualization. Refer to *Section 33.6, "Configuring Log Visualization"* for details.

Important: Controlling the Audit Daemon

Before configuring any of the components of the audit system, make sure that the audit daemon is not running by entering **`systemctl status auditd`** as `root`. On a default openSUSE Leap system, audit is started on boot, so you need to turn it off by entering **`systemctl stop auditd`**. Start the daemon after configuring it with **`systemctl start auditd`**.

33.1 Determining the Components to Audit

Before starting to create your own audit configuration, determine to which degree you want to use it. Check the following general rules to determine which use case best applies to you and your requirements:

- If you require a full security audit for CAPP/EAL certification, enable full audit for system calls and configure watches on various configuration files and directories, similar to the rule set featured in *Chapter 34, Introducing an Audit Rule Set*.
- If you need to trace a process based on the audit rules, use **`autrace`**.
- If you require file and directory watches to track access to important or security-sensitive data, create a rule set matching these requirements. Enable audit as described in *Section 33.3, "Enabling Audit for System Calls"* and proceed to *Section 33.4, "Setting Up Audit Rules"*.

33.2 Configuring the Audit Daemon

The basic setup of the audit daemon is done by editing `/etc/audit/auditd.conf`. You may also use YaST to configure the basic settings by calling *YaST › Security and Users › Linux Audit Framework (LAF)*. Use the tabs *Log File* and *Disk Space* for configuration.

```
log_file = /var/log/audit/audit.log
log_format = RAW
log_group = root
priority_boost = 4
flush = INCREMENTAL
freq = 20
num_logs = 5
disp_qos = lossy
dispatcher = /sbin/audispd
name_format = NONE
##name = mydomain
max_log_file = 6
max_log_file_action = ROTATE
space_left = 75
space_left_action = SYSLOG
action_mail_acct = root
admin_space_left = 50
admin_space_left_action = SUSPEND
disk_full_action = SUSPEND
```

```
disk_error_action = SUSPEND
##tcp_listen_port =
tcp_listen_queue = 5
tcp_max_per_addr = 1
##tcp_client_ports = 1024-65535
tcp_client_max_idle = 0
cp_client_max_idle = 0
```

The default settings work reasonably well for many setups. Some values, such as `num_logs`, `max_log_file`, `space_left`, and `admin_space_left` depend on the size of your deployment. If disk space is limited, you should reduce the number of log files to keep if they are rotated and you should get an earlier warning if disk space is running out. For a CAPP-compliant setup, adjust the values for `log_file`, `flush`, `max_log_file`, `max_log_file_action`, `space_left`, `space_left_action`, `admin_space_left`, `admin_space_left_action`, `disk_full_action`, and `disk_error_action`, as described in *Section 32.2, "Configuring the Audit Daemon"*. An example CAPP-compliant configuration looks like this:

```
log_file = PATH_TO_SEPARATE_PARTITION/audit.log
log_format = RAW
priority_boost = 4
flush = SYNC                        ### or DATA
freq = 20
num_logs = 4
dispatcher = /sbin/audispd
disp_qos = lossy
max_log_file = 5
max_log_file_action = KEEP_LOGS
space_left = 75
space_left_action = EMAIL
action_mail_acct = root
admin_space_left = 50
admin_space_left_action = SINGLE   ### or HALT
disk_full_action = SUSPEND         ### or HALT
disk_error_action = SUSPEND        ### or HALT
```

The `###` precedes comments where you can choose from several options. Do not add the comments to your actual configuration files.

Tip: For More Information

Refer to *Section 32.2, "Configuring the Audit Daemon"* for detailed background information about the `auditd.conf` configuration parameters.

33.3 Enabling Audit for System Calls

If the audit framework is not installed, install the `audit` package. A standard openSUSE Leap system does not have auditd running by default. Enable it with:

```
tux > sudo systemctl enable auditd
```

There are different levels of auditing activity available:

Basic Logging

Out of the box (without any further configuration) auditd logs only events concerning its own configuration changes to `/var/log/audit/audit.log`. No events (file access, system call, etc.) are generated by the kernel audit component until requested by **`auditctl`**. However, other kernel components and modules may log audit events outside of the control of **`auditctl`** and these appear in the audit log. By default, the only module that generates audit events is AppArmor.

Advanced Logging with System Call Auditing

To audit system calls and get meaningful file watches, you need to enable audit contexts for system calls.

As you need system call auditing capabilities even when you are configuring plain file or directory watches, you need to enable audit contexts for system calls. To enable audit contexts for the duration of the current session only, execute **`auditctl -e 1`** as `root`. To disable this feature, execute **`auditctl -e 0`** as `root`.

The audit contexts are enabled by default. To turn this feature off temporarily, use **`auditctl -e 0`**.

33.4 Setting Up Audit Rules

Using audit rules, determine which aspects of the system should be analyzed by audit. Normally this includes important databases and security-relevant configuration files. You may also analyze various system calls in detail if a broad analysis of your system is required. A very detailed example configuration that includes most of the rules that are needed in a CAPP compliant environment is available in *Chapter 34, Introducing an Audit Rule Set*.

Audit rules can be passed to the audit daemon on the **auditctl** command line and by composing a rule set in `/etc/audit/audit.rules` which is processed whenever the audit daemon is started. To customize `/etc/audit/audit.rules` either edit it directly, or use YaST: *Security and Users* › *Linux Audit Framework (LAF)* › *Rules for 'auditctl'*. Rules passed on the command line are not persistent and need to be re-entered when the audit daemon is restarted.

A simple rule set for very basic auditing on a few important files and directories could look like this:

```
# basic audit system parameters
-D
-b 8192
-f 1
-e 1

# some file and directory watches with keys
-w /var/log/audit/ -k LOG_audit
-w /etc/audit/auditd.conf -k CFG_audit_conf -p rxwa
-w /etc/audit/audit.rules -k CFG_audit_rules -p rxwa

-w /etc/passwd -k CFG_passwd -p rwxa
-w /etc/sysconfig/ -k CFG_sysconfig

# an example system call rule
-a entry,always -S umask

### add your own rules
```

When configuring the basic audit system parameters (such as the backlog parameter `-b`) test these settings with your intended audit rule set to determine whether the backlog size is appropriate for the level of logging activity caused by your audit rule set. If your chosen backlog size is too small, your system might not be able to handle the audit load and consult the failure flag (`-f`) when the backlog limit is exceeded.

Important: Choosing the Failure Flag

When choosing the failure flag, note that `-f 2` tells your system to perform an immediate shutdown without flushing any pending data to disk when the limits of your audit system are exceeded. Because this shutdown is not a clean shutdown, restrict the use of `-f 2` to only the most security-conscious environments and use `-f 1` (system continues to run, issues a warning and audit stops) for any other setup to avoid loss of data or data corruption.

Directory watches produce less verbose output than separate file watches for the files under these directories. To get detailed logging for your system configuration in `/etc/sysconfig`, for example, add watches for each file. Audit does not support globbing, which means you cannot create a rule that says `-w /etc/*` and watches all files and directories below `/etc`.

For better identification in the log file, a key has been added to each of the file and directory watches. Using the key, it is easier to comb the logs for events related to a certain rule. When creating keys, distinguish between mere log file watches and configuration file watches by using an appropriate prefix with the key, in this case `LOG` for a log file watch and `CFG` for a configuration file watch. Using the file name as part of the key also makes it easier for you to identify events of this type in the log file.

Another thing to keep in mind when creating file and directory watches is that audit cannot deal with files that do not exist when the rules are created. Any file that is added to your system while audit is already running is not watched unless you extend the rule set to watch this new file.

For more information about creating custom rules, refer to *Section 32.4, "Passing Parameters to the Audit System"*.

Important: Changing Audit Rules

After you change audit rules, always restart the audit daemon with **`systemctl restart auditd`** to reread the changed rules.

33.5 Configuring Audit Reports

To avoid having to dig through the raw audit logs to get an impression of what your system is currently doing, run custom audit reports at certain intervals. Custom audit reports enable you to focus on areas of interest and get meaningful statistics on the nature and frequency of the events you are monitoring. To analyze individual events in detail, use the ausearch tool.

Before setting up audit reporting, consider the following:

- What types of events do you want to monitor by generating regular reports? Select the appropriate aureport command lines as described in *Section 32.5.2, "Generating Custom Audit Reports"*.
- What do you want to do with the audit reports? Decide whether to create graphical charts from the data accumulated or whether it should be transferred into any sort of spreadsheet or database. Set up the aureport command line and further processing similar to the examples shown in *Section 33.6, "Configuring Log Visualization"* if you want to visualize your reports.
- When and at which intervals should the reports run? Set up appropriate automated reporting using cron.

For this example, assume that you are interested in finding out about any attempts to access your audit, PAM, and system configuration. Proceed as follows to find out about file events on your system:

1. Generate a full summary report of all events and check for any anomalies in the summary report, for example, have a look at the "failed syscalls" record, because these might have failed because of insufficient permissions to access a file or a file not being there:

```
tux > sudo aureport

Summary Report
======================
Range of time in logs: 03/02/09 14:13:38.225 - 17/02/09 16:30:10.352
Selected time for report: 03/02/09 14:13:38 - 17/02/09 16:30:10.352
Number of changes in configuration: 24
Number of changes to accounts, groups, or roles: 0
Number of logins: 9
Number of failed logins: 15
Number of authentications: 19
Number of failed authentications: 578
Number of users: 3
Number of terminals: 15
Number of host names: 4
Number of executables: 20
Number of files: 279
Number of AVC's: 0
Number of MAC events: 0
Number of failed syscalls: 994
```

```
Number of anomaly events: 0
Number of responses to anomaly events: 0
Number of crypto events: 0
Number of keys: 2
Number of process IDs: 1238
Number of events: 5435
```

2. Run a summary report for failed events and check the “files” record for the number of failed file access events:

```
tux > sudo aureport --failed

Failed Summary Report
======================
Range of time in logs: 03/02/09 14:13:38.225 - 17/02/09 16:30:10.352
Selected time for report: 03/02/09 14:13:38 - 17/02/09 16:30:10.352
Number of changes in configuration: 0
Number of changes to accounts, groups, or roles: 0
Number of logins: 0
Number of failed logins: 15
Number of authentications: 0
Number of failed authentications: 578
Number of users: 1
Number of terminals: 7
Number of host names: 4
Number of executables: 12
Number of files: 77
Number of AVC's: 0
Number of MAC events: 0
Number of failed syscalls: 994
Number of anomaly events: 0
Number of responses to anomaly events: 0
Number of crypto events: 0
Number of keys: 2
Number of process IDs: 713
Number of events: 1589
```

3. To list the files that could not be accessed, run a summary report of failed file events:

```
tux > sudo aureport -f -i --failed --summary

Failed File Summary Report
===========================
total  file
===========================
80  /var
80  spool
```

```
80  cron
80  lastrun
46  /usr/lib/locale/en_GB.UTF-8/LC_CTYPE
45  /usr/lib/locale/locale-archive
38  /usr/lib/locale/en_GB.UTF-8/LC_IDENTIFICATION
38  /usr/lib/locale/en_GB.UTF-8/LC_MEASUREMENT
38  /usr/lib/locale/en_GB.UTF-8/LC_TELEPHONE
38  /usr/lib/locale/en_GB.UTF-8/LC_ADDRESS
38  /usr/lib/locale/en_GB.UTF-8/LC_NAME
38  /usr/lib/locale/en_GB.UTF-8/LC_PAPER
38  /usr/lib/locale/en_GB.UTF-8/LC_MESSAGES
38  /usr/lib/locale/en_GB.UTF-8/LC_MONETARY
38  /usr/lib/locale/en_GB.UTF-8/LC_COLLATE
38  /usr/lib/locale/en_GB.UTF-8/LC_TIME
38  /usr/lib/locale/en_GB.UTF-8/LC_NUMERIC
8  /etc/magic.mgc
...
```

To focus this summary report on a few files or directories of interest only, such as `/etc/audit/auditd.conf`, `/etc/pam.d`, and `/etc/sysconfig`, use a command similar to the following:

```
tux > sudo aureport -f -i --failed --summary |grep -e "/etc/audit/auditd.conf" -e "/
etc/pam.d/" -e "/etc/sysconfig"

1  /etc/sysconfig/displaymanager
```

4. From the summary report, then proceed to isolate these items of interest from the log and find out their event IDs for further analysis:

```
tux > sudo aureport -f -i --failed |grep -e "/etc/audit/auditd.conf" -e "/etc/
pam.d/" -e "/etc/sysconfig"

993. 17/02/09 16:47:34 /etc/sysconfig/displaymanager readlink no /bin/vim-normal
 root 7887
994. 17/02/09 16:48:23 /etc/sysconfig/displaymanager getxattr no /bin/vim-normal
 root 7889
```

5. Use the event ID to get a detailed record for each item of interest:

```
tux > sudo ausearch -a 7887 -i
----
time->Tue Feb 17 16:48:23 2009
type=PATH msg=audit(1234885703.090:7889): item=0 name="/etc/sysconfig/
displaymanager" inode=369282 dev=08:06 mode=0100644 ouid=0 ogid=0 rdev=00:00
type=CWD msg=audit(1234885703.090:7889):  cwd="/root"
```

```
type=SYSCALL msg=audit(1234885703.090:7889): arch=c000003e syscall=191 success=no
 exit=-61 a0=7e1e20 a1=7f90e4cf9187 a2=7fffed5b57d0 a3=84 items=1 ppid=25548
 pid=23045 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts2
 ses=1166 comm="vim" exe="/bin/vim-normal" key=(null)
```

Tip: Focusing on a Certain Time Frame

If you are interested in events during a particular period of time, trim down the reports by using start and end dates and times with your **aureport** commands (`-ts` and `-te`). For more information, refer to *Section 32.5.2, "Generating Custom Audit Reports"*.

All steps (except for the last one) can be run automatically and would easily be scriptable and configured as cron jobs. Any of the `--failed --summary` reports could be transformed easily into a bar chart that plots files versus failed access attempts. For more information about visualizing audit report data, refer to *Section 33.6, "Configuring Log Visualization"*.

33.6 Configuring Log Visualization

Using the scripts **mkbar** and **mkgraph** you can illustrate your audit statistics with various graphs and charts. As with any other **aureport** command, the plotting commands are scriptable and can easily be configured to run as cron jobs.

mkbar and **mkgraph** were created by Steve Grubb at Red Hat. They are available from http://people.redhat.com/sgrubb/audit/visualize/. Because the current version of audit in openSUSE Leap does not ship with these scripts, proceed as follows to make them available on your system:

Warning: Downloaded Content Is Dangerous

Use **mkbar** and **mkgraph** at your own risk. Any content downloaded from the Web is potentially dangerous to your system, even more so when run with `root` privileges.

1. Download the scripts to `root`'s `~/bin` directory:

```
tux > sudo wget http://people.redhat.com/sgrubb/audit/visualize/mkbar -O ~/bin/mkbar
tux > sudo wget http://people.redhat.com/sgrubb/audit/visualize/mkgraph -O ~/bin/
mkgraph
```

2. Adjust the file permissions to read, write, and execute for `root`:

```
tux > sudo chmod 744 ~/bin/mk{bar,graph}
```

To plot summary reports, such as the ones discussed in *Section 33.5, "Configuring Audit Reports"*, use the script **`mkbar`**. Some example commands could look like the following:

Create a Summary of Events

```
tux > sudo aureport -e -i --summary | mkbar events
```

Create a Summary of File Events

```
tux > sudo aureport -f -i --summary | mkbar files
```

Create a Summary of Login Events

```
tux > sudo aureport -l -i --summary | mkbar login
```

Create a Summary of User Events

```
tux > sudo aureport -u -i --summary | mkbar users
```

Create a Summary of System Call Events

```
tux > sudo aureport -s -i --summary | mkbar syscalls
```

To create a summary chart of failed events of any of the above event types, add the `--failed` option to the respective **`aureport`** command. To cover a certain period of time only, use the `-ts` and `-te` options on aureport. Any of these commands can be tweaked further by narrowing down its scope using grep or egrep and regular expressions. See the comments in the **`mkbar`** script for an example. Any of the above commands produces a PNG file containing a bar chart of the requested data.

To illustrate the relationship between different kinds of audit objects, such as users and system calls, use the script **`mkgraph`**. Some example commands could look like the following:

Users versus Executables

```
tux > sudo LC_ALL=C aureport -u -i | awk '/^[0-9]/ { print $4" "$7 }' | sort | uniq
 | mkgraph users_vs_exec
```

Users versus Files

```
tux > sudo LC_ALL=C aureport -f -i | awk '/^[0-9]/ { print $8" "$4 }' | sort | uniq
 | mkgraph users_vs_files
```

System Calls versus Commands

```
tux > sudo LC_ALL=C aureport -s -i | awk '/^[0-9]/ { print $4" "$6 }' | sort | uniq
 | mkgraph syscall_vs_com
```

System Calls versus Files

```
tux > sudo LC_ALL=C aureport -s -i | awk '/^[0-9]/ { print $5" "$4 }' | sort | uniq
 | mkgraph | syscall_vs_file
```

Graphs can also be combined to illustrate complex relationships. See the comments in the **`mkgraph`** script for further information and an example. The graphs produced by this script are created in PostScript format by default, but you can change the output format by changing the `EXT` variable in the script from `ps` to `png` or `jpg`.

34 Introducing an Audit Rule Set

The following example configuration illustrates how audit can be used to monitor your system. It highlights the most important items that need to be audited to cover the list of auditable events specified by Controlled Access Protection Profile (CAPP).

The example rule set is divided into the following sections:

- Basic audit configuration (see *Section 34.1, "Adding Basic Audit Configuration Parameters"*)
- Watches on audit log files and configuration files (see *Section 34.2, "Adding Watches on Audit Log Files and Configuration Files"*)
- Monitoring operations on file system objects (see *Section 34.3, "Monitoring File System Objects"*)
- Monitoring security databases (see *Section 34.4, "Monitoring Security Configuration Files and Databases"*)
- Monitoring miscellaneous system calls (*Section 34.5, "Monitoring Miscellaneous System Calls"*)
- Filtering system call arguments (see *Section 34.6, "Filtering System Call Arguments"*)

To transform this example into a configuration file to use in your live setup, proceed as follows:

1. Choose the appropriate settings for your setup and adjust them.
2. Adjust the file `/etc/audit/audit.rules` by adding rules from the examples below or by modifying existing rules.

Note: Adjusting the Level of Audit Logging

Do not copy the example below into your audit setup without adjusting it to your needs. Determine what and to what extent to audit.

The entire `audit.rules` is a collection of **`auditctl`** commands. Every line in this file expands to a full **`auditctl`** command line. The syntax used in the rule set is the same as that of the **`auditctl`** command.

34.1 Adding Basic Audit Configuration Parameters

```
-D ❶
-b 8192 ❷
-f 2 ❸
```

❶ Delete any preexisting rules before starting to define new ones.

❷ Set the number of buffers to take the audit messages. Depending on the level of audit logging on your system, increase or decrease this figure.

❸ Set the failure flag to use when the kernel needs to handle critical errors. Possible values are `0` (silent), `1` (printk, print a failure message), and `2` (panic, halt the system).

By emptying the rule queue with the `-D` option, you make sure that audit does not use any other rule set than what you are offering it by means of this file. Choosing an appropriate buffer number (`-b`) is vital to avoid having your system fail because of too high an audit load. Choosing the panic failure flag `-f 2` ensures that your audit records are complete even if the system is encountering critical errors. By shutting down the system on a critical error, audit makes sure that no process escapes from its control as it otherwise might if level 1 (`printk`) were chosen.

Important: Choosing the Failure Flag

Before using your audit rule set on a live system, make sure that the setup has been thoroughly evaluated on test systems using the *worst case production workload*. It is even more critical that you do this when specifying the `-f 2` flag, because this instructs the kernel to panic (perform an immediate halt without flushing pending data to disk) if any thresholds are exceeded. Consider the use of the `-f 2` flag for only the most security-conscious environments.

34.2 Adding Watches on Audit Log Files and Configuration Files

Adding watches on your audit configuration files and the log files themselves ensures that you can track any attempt to tamper with the configuration files or detect any attempted accesses to the log files.

Note: Creating Directory and File Watches

Creating watches on a directory is not necessarily sufficient if you need events for file access. Events on directory access are only triggered when the directory's inode is updated with metadata changes. To trigger events on file access, add watches for each file to monitor.

```
-w /var/log/audit/ 1
-w /var/log/audit/audit.log

-w /var/log/audit/audit_log.1
-w /var/log/audit/audit_log.2
-w /var/log/audit/audit_log.3
-w /var/log/audit/audit_log.4

-w /etc/audit/auditd.conf -p wa2
-w /etc/audit/audit.rules -p wa
-w /etc/libaudit.conf -p wa
```

1. Set a watch on the directory where the audit log is located. Trigger an event for any type of access attempt to this directory. If you are using log rotation, add watches for the rotated logs as well.
2. Set a watch on an audit configuration file. Log all write and attribute change attempts to this file.

34.3 Monitoring File System Objects

Auditing system calls helps track your system's activity well beyond the application level. By tracking file system–related system calls, get an idea of how your applications are using these system calls and determine whether that use is appropriate. By tracking mount and unmount operations, track the use of external resources (removable media, remote file systems, etc.).

Important: Auditing System Calls

Auditing system calls results in a high logging activity. This activity, in turn, puts a heavy load on the kernel. With a kernel less responsive than usual, the system's backlog and rate limits might be exceeded. Carefully evaluate which system calls to include in your audit rule set and adjust the log settings accordingly. See *Section 32.2, "Configuring the Audit Daemon"* for details on how to tweak the relevant settings.

```
-a entry,always -S chmod -S fchmod -S chown -S chown32 -S fchown -S fchown32 -S lchown -S
 lchown32 (1)

-a entry,always -S creat -S open -S truncate -S truncate64 -S ftruncate -S ftruncate64 (2)

-a entry,always -S mkdir -S rmdir (3)

-a entry,always -S unlink -S rename -S link -S symlink (4)

-a entry,always -S setxattr (5)
-a entry,always -S lsetxattr
-a entry,always -S fsetxattr
-a entry,always -S removexattr
-a entry,always -S lremovexattr
-a entry,always -S fremovexattr

-a entry,always -S mknod (6)

-a entry,always -S mount -S umount -S umount2 (7)
```

1. Enable an audit context for system calls related to changing file ownership and permissions. Depending on the hardware architecture of your system, enable or disable the `*32` rules. 64-bit systems, like AMD64/Intel 64, require the `*32` rules to be removed.
2. Enable an audit context for system calls related to file content modification. Depending on the hardware architecture of your system, enable or disable the *64 rules. 64-bit systems, like AMD64/Intel 64, require the *64 rules to be removed.
3. Enable an audit context for any directory operation, like creating or removing a directory.
4. Enable an audit context for any linking operation, such as creating a symbolic link, creating a link, unlinking, or renaming.
5. Enable an audit context for any operation related to extended file system attributes.
6. Enable an audit context for the **`mknod`** system call, which creates special (device) files.

7 Enable an audit context for any mount or umount operation. For the x86 architecture, disable the `umount` rule. For the Intel 64 architecture, disable the `umount2` rule.

34.4 Monitoring Security Configuration Files and Databases

To make sure that your system is not made to do undesired things, track any attempts to change the `cron` and `at` configurations or the lists of scheduled jobs. Tracking any write access to the user, group, password and login databases and logs helps you identify any attempts to manipulate your system's user database.

Tracking changes to your system configuration (kernel, services, time, etc.) helps you spot any attempts of others to manipulate essential functionality of your system. Changes to the PAM configuration should also be monitored in a secure environment, because changes in the authentication stack should not be made by anyone other than the administrator, and it should be logged which applications are using PAM and how it is used. The same applies to any other configuration files related to secure authentication and communication.

```
1
-w /var/spool/atspool
-w /etc/at.allow
-w /etc/at.deny

-w /etc/cron.allow -p wa
-w /etc/cron.deny -p wa
-w /etc/cron.d/ -p wa
-w /etc/cron.daily/ -p wa
-w /etc/cron.hourly/ -p wa
-w /etc/cron.monthly/ -p wa
-w /etc/cron.weekly/ -p wa
-w /etc/crontab -p wa
-w /var/spool/cron/root

2
-w /etc/group -p wa
-w /etc/passwd -p wa
-w /etc/shadow

-w /etc/login.defs -p wa
-w /etc/securetty
-w /var/log/lastlog
```

```
❸
-w /etc/hosts -p wa
-w /etc/sysconfig/
w /etc/init.d/
w /etc/ld.so.conf -p wa
w /etc/localtime -p wa
w /etc/sysctl.conf -p wa
w /etc/modprobe.d/
w /etc/modprobe.conf.local -p wa
w /etc/modprobe.conf -p wa
❹
w /etc/pam.d/
❺
-w /etc/aliases -p wa
-w /etc/postfix/ -p wa

❻
-w /etc/ssh/sshd_config

-w /etc/stunnel/stunnel.conf
-w /etc/stunnel/stunnel.pem

-w /etc/vsftpd.ftpusers
-w /etc/vsftpd.conf

❼
-a exit,always -S sethostname
-w /etc/issue -p wa
-w /etc/issue.net -p wa
```

❶ Set watches on the `at` and `cron` configuration and the scheduled jobs and assign labels to these events.

❷ Set watches on the user, group, password, and login databases and logs and set labels to better identify any login-related events, such as failed login attempts.

❸ Set a watch and a label on the static host name configuration in `/etc/hosts`. Track changes to the system configuration directory, `/etc/sysconfig`. Enable per-file watches if you are interested in file events. Set watches and labels for changes to the boot configuration in the `/etc/init.d` directory. Enable per-file watches if you are interested in file events. Set watches and labels for any changes to the linker configuration in `/etc/ld.so.conf`. Set watches and a label for `/etc/localtime`. Set watches and labels for the kernel configuration files `/etc/sysctl.conf`, `/etc/modprobe.d/`, `/etc/modprobe.conf.local`, and `/etc/modprobe.conf`.

4. Set watches on the PAM configuration directory. If you are interested in particular files below the directory level, add explicit watches to these files as well.
5. Set watches to the postfix configuration to log any write attempt or attribute change and use labels for better tracking in the logs.
6. Set watches and labels on the SSH, `stunnel`, and `vsftpd` configuration files.
7. Perform an audit of the `sethostname` system call and set watches and labels on the system identification configuration in `/etc/issue` and `/etc/issue.net`.

34.5 Monitoring Miscellaneous System Calls

Apart from auditing file system related system calls, as described in *Section 34.3, "Monitoring File System Objects"*, you can also track various other system calls. Tracking task creation helps you understand your applications' behavior. Auditing the `umask` system call lets you track how processes modify creation mask. Tracking any attempts to change the system time helps you identify anyone or any process trying to manipulate the system time.

```
❶
-a entry,always -S clone -S fork -S vfork

❷
-a entry,always -S umask

❸
-a entry,always -S adjtimex -S settimeofday
```

1. Track task creation.
2. Add an audit context to the umask system call.
3. Track attempts to change the system time. `adjtimex` can be used to skew the time. `settimeofday` sets the absolute time.

34.6 Filtering System Call Arguments

In addition to the system call auditing introduced in *Section 34.3, "Monitoring File System Objects"* and *Section 34.5, "Monitoring Miscellaneous System Calls"*, you can track application behavior to an even higher degree. Applying filters helps you focus audit on areas of primary interest to you. This section introduces filtering system call arguments for non-multiplexed system calls like

access and for multiplexed ones like socketcall or ipc. Whether system calls are multiplexed depends on the hardware architecture used. Both socketcall and ipc are not multiplexed on 64-bit architectures, such as AMD64/Intel 64.

Important: Auditing System Calls

Auditing system calls results in high logging activity, which in turn puts a heavy load on the kernel. With a kernel less responsive than usual, the system's backlog and rate limits might well be exceeded. Carefully evaluate which system calls to include in your audit rule set and adjust the log settings accordingly. See *Section 32.2, "Configuring the Audit Daemon"* for details on how to tweak the relevant settings.

The access system call checks whether a process would be allowed to read, write or test for the existence of a file or file system object. Using the `-F` filter flag, build rules matching specific access calls in the format `-F a1=ACCESS_MODE`. Check `/usr/include/fcntl.h` for a list of possible arguments to the access system call.

```
-a entry,always -S access -F a1=4 (1)
-a entry,always -S access -F a1=6 (2)
-a entry,always -S access -F a1=7 (3)
```

(1) Audit the access system call, but only if the second argument of the system call (`mode`) is `4` (`R_OK`). This rule filters for all access calls testing for sufficient read permissions to a file or file system object accessed by a user or process.

(2) Audit the access system call, but only if the second argument of the system call (`mode`) is `6`, meaning `4 OR 2`, which translates to `R_OK OR W_OK`. This rule filters for access calls testing for sufficient read and write permissions.

(3) Audit the access system call, but only if the second argument of the system call (`mode`) is `7`, meaning `4 OR 2 OR 1`, which translates to `R_OK OR W_OK OR X_OK`. This rule filters for access calls testing for sufficient read, write, and execute permissions.

The socketcall system call is a multiplexed system call. Multiplexed means that there is only one system call for all possible calls and that libc passes the actual system call to use as the first argument (`a0`). Check the manual page of socketcall for possible system calls and refer to `/usr/src/linux/include/linux/net.h` for a list of possible argument values and system call names. Audit supports filtering for specific system calls using a `-F a0=SYSCALL_NUMBER`.

```
-a entry,always -S socketcall -F a0=1 -F a1=10 (1)
```

```
## Use this line on x86_64, ia64 instead
#-a entry,always -S socket -F a0=10

-a entry,always -S socketcall -F a0=5 ❷
## Use this line on x86_64, ia64 instead
#-a entry, always -S accept
```

❶ Audit the socket(PF_INET6) system call. The `-F a0=1` filter matches all socket system calls and the `-F a1=10` filter narrows the matches down to socket system calls carrying the IPv6 protocol family domain parameter (PF_INET6). Check `/usr/include/linux/net.h` for the first argument (`a0`) and `/usr/src/linux/include/linux/socket.h` for the second parameter (`a1`). 64-bit platforms, like AMD64/Intel 64, do not use multiplexing on socketcall system calls. For these platforms, comment the rule and add the plain system call rules with a filter on PF_INET6.

❷ Audit the socketcall system call. The filter flag is set to filter for `a0=5` as the first argument to socketcall, which translates to the accept system call if you check `/usr/include/linux/net.h`. 64-bit platforms, like AMD64/Intel 64, do not use multiplexing on socketcall system calls. For these platforms, comment the rule and add the plain system call rule without argument filtering.

The ipc system call is another example of multiplexed system calls. The actual call to invoke is determined by the first argument passed to the ipc system call. Filtering for these arguments helps you focus on those IPC calls of interest to you. Check `/usr/include/linux/ipc.h` for possible argument values.

```
❶
## msgctl
-a entry,always -S ipc -F a0=14
## msgget
-a entry,always -S ipc -F a0=13
## Use these lines on x86_64, ia64 instead
#-a entry,always -S msgctl
#-a entry,always -S msgget

❷
## semctl
-a entry,always -S ipc -F a0=3
## semget
-a entry,always -S ipc -F a0=2
## semop
-a entry,always -S ipc -F a0=1
## semtimedop
-a entry,always -S ipc -F a0=4
```

```
## Use these lines on x86_64, ia64 instead
#-a entry,always -S semctl
#-a entry,always -S semget
#-a entry,always -S semop
#-a entry,always -S semtimedop

 3
## shmctl
-a entry,always -S ipc -F a0=24
## shmget
-a entry,always -S ipc -F a0=23
## Use these lines on x86_64, ia64 instead
#-a entry,always -S shmctl
#-a entry,always -S shmget
```

1. Audit system calls related to IPC SYSV message queues. In this case, the `a0` values specify that auditing is added for the msgctl and msgget system calls (`14` and `13`). 64-bit platforms, like AMD64/Intel 64, do not use multiplexing on ipc system calls. For these platforms, comment the first two rules and add the plain system call rules without argument filtering.
2. Audit system calls related to IPC SYSV message semaphores. In this case, the `a0` values specify that auditing is added for the semctl, semget, semop, and semtimedop system calls (`3`, `2`, `1`, and `4`). 64-bit platforms, like AMD64/Intel 64, do not use multiplexing on ipc system calls. For these platforms, comment the first four rules and add the plain system call rules without argument filtering.
3. Audit system calls related to IPC SYSV shared memory. In this case, the `a0` values specify that auditing is added for the shmctl and shmget system calls (`24`, `23`). 64-bit platforms, like AMD64/Intel 64, do not use multiplexing on ipc system calls. For these platforms, comment the first two rules and add the plain system call rules without argument filtering.

34.7 Managing Audit Event Records Using Keys

After configuring a few rules generating events and populating the logs, you need to find a way to tell one event from the other. Using the **`ausearch`** command, you can filter the logs for various criteria. Using **`ausearch`** `-m` *`MESSAGE_TYPE`*, you can at least filter for events of a certain type. However, to be able to filter for events related to a particular rule, you need to add a key to this rule in the `/etc/audit/audit.rules` file. This key is then added to the event record every time the rule logs an event. To retrieve these log entries, simply run **`ausearch`** `-k` *`YOUR_KEY`* to get a list of records related to the rule carrying this particular key.

As an example, assume you have added the following rule to your rule file:

```
-w /etc/audit/audit.rules -p wa
```

Without a key assigned to it, you would probably need to filter for `SYSCALL` or `PATH` events then use grep or similar tools to isolate any events related to the above rule. Now, add a key to the above rule, using the `-k` option:

```
-w /etc/audit/audit.rules -p wa -k CFG_audit.rules
```

You can specify any text string as key. Distinguish watches related to different types of files (configuration files or log files) from one another using different key prefixes (`CFG`, `LOG`, etc.) followed by the file name. Finding any records related to the above rule now comes down to the following:

```
ausearch -k CFG_audit.rules
----
time->Thu Feb 19 09:09:54 2009
type=PATH msg=audit(1235030994.032:8649): item=3 name="audit.rules~" inode=370603
 dev=08:06 mode=0100640 ouid=0 ogid=0 rdev=00:00
type=PATH msg=audit(1235030994.032:8649): item=2 name="audit.rules" inode=370603
 dev=08:06 mode=0100640 ouid=0 ogid=0 rdev=00:00
type=PATH msg=audit(1235030994.032:8649): item=1  name="/etc/audit" inode=368599
 dev=08:06 mode=040750 ouid=0 ogid=0 rdev=00:00
type=PATH msg=audit(1235030994.032:8649): item=0  name="/etc/audit" inode=368599
 dev=08:06 mode=040750 ouid=0 ogid=0 rdev=00:00
type=CWD msg=audit(1235030994.032:8649):  cwd="/etc/audit"
type=SYSCALL msg=audit(1235030994.032:8649): arch=c000003e syscall=82 success=yes exit=0
 a0=7deeb0 a1=883b30 a2=2 a3=ffffffffffffffff items=4 ppid=25400 pid=32619 auid=0 uid=0
 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts1 ses=1164 comm="vim" exe="/
bin/vim-normal" key="CFG_audit.rules"
```

35 Useful Resources

There are other resources available containing valuable information about the Linux audit framework:

The Audit Manual Pages

There are several man pages installed along with the audit tools that provide valuable and very detailed information:

`auditd(8)`
: The Linux audit daemon

`auditd.conf(5)`
: The Linux audit daemon configuration file

`auditctl(8)`
: A utility to assist controlling the kernel's audit system

`autrace(8)`
: A program similar to **`strace`**

`ausearch(8)`
: A tool to query audit daemon logs

`aureport(8)`
: A tool that produces summary reports of audit daemon logs

`audispd.conf(5)`
: The audit event dispatcher configuration file

`audispd(8)`
: The audit event dispatcher daemon talking to plug-in programs.

http://people.redhat.com/sgrubb/audit/index.html
: The home page of the Linux audit project. This site contains several specifications relating to different aspects of Linux audit, and a short FAQ.

`/usr/share/doc/packages/audit`
: The audit package itself contains a README with basic design information and sample `.rules` files for different scenarios:

  `capp.rules`: Controlled Access Protection Profile (CAPP)

`lspp.rules`: Labeled Security Protection Profile (LSPP)

`nispom.rules`: National Industrial Security Program Operating Manual Chapter 8(NISPOM)

`stig.rules`: Secure Technical Implementation Guide (STIG)

https://www.commoncriteriaportal.org/

The official Web site of the Common Criteria project. Learn all about the Common Criteria security certification initiative and which role audit plays in this framework.

A GNU Licenses

This appendix contains the GNU Free Documentation License version 1.2.

GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or 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.

Reference

openSUSE Leap 15.1

Reference

openSUSE Leap 15.1

Publication Date: May 25, 2019

SUSE LLC
10 Canal Park Drive
Suite 200
Cambridge MA 02141
USA
https://www.suse.com/documentation

Contents

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

System

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

Services

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

Mobile Computers

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

1 Available Documentation

Note: Online Documentation and Latest Updates

Documentation for our products is available at http://doc.opensuse.org/, where you can also find the latest updates, and browse or download the documentation in various formats.

In addition, the product documentation is usually available in your installed system under `/usr/share/doc/manual`.

The following documentation is available for this product:

***Book* "Start-Up"**

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

Reference

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

***Book* "Virtualization Guide"**

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

***Book* "AutoYaST Guide"**

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

***Book* "Security 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.

2 Feedback

Several feedback channels are available:

Bug Reports

To report bugs for openSUSE Leap, go to https://bugzilla.opensuse.org/, log in, and click *New*.

Mail

For feedback on the documentation of this product, you can also send a mail to `doc-team@suse.com`. 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 notices and typographical conventions are used in this documentation:

- `/etc/passwd`: directory names and file names
- *`PLACEHOLDER`*: replace *`PLACEHOLDER`* with the actual value
- `PATH`: the environment variable PATH
- **`ls`**, `--help`: commands, options, and parameters
- `user`: users or groups
- `package name` : name of a package
- `Alt`, `Alt`–`F1`: a key to press or a key combination; keys are shown in uppercase as on a keyboard
- *File*, *File* › *Save As*: menu items, buttons
- *Dancing Penguins* (Chapter *Penguins*, ↑Another Manual): This is a reference to a chapter in another manual.
- Commands that must be run with `root` privileges. Often you can also prefix these commands with the **`sudo`** command to run them as non-privileged user.

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

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

```
tux > command
```

- Notices

Warning: Warning Notice

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

Important: Important Notice

Important information you should be aware of before proceeding.

Note: Note Notice

Additional information, for example about differences in software versions.

Tip: Tip Notice

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

4 About the Making of This Documentation

This documentation is written in GeekoDoc (https://github.com/openSUSE/geekodoc), 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

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.

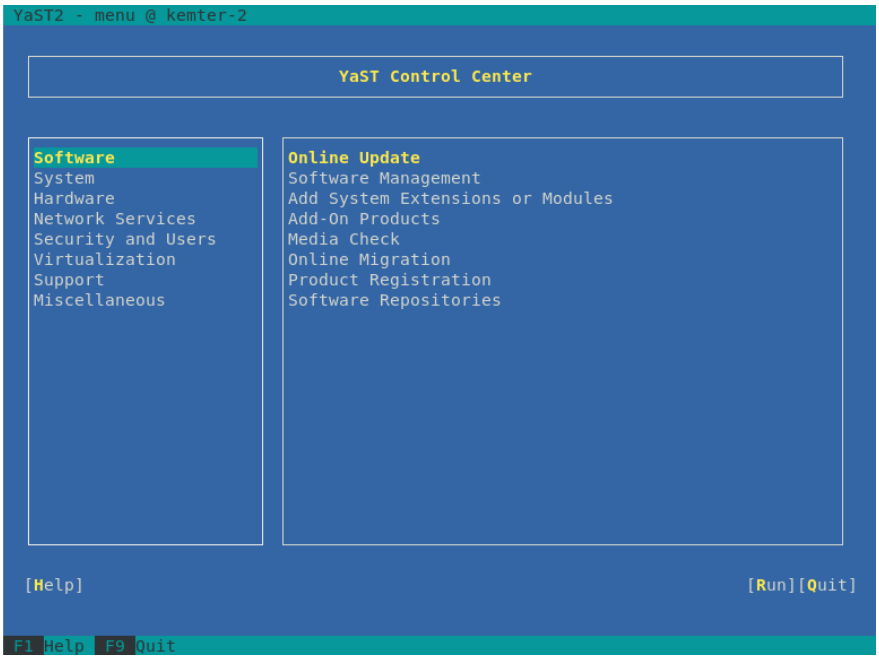


FIGURE 1.1: MAIN WINDOW OF YAST IN TEXT MODE

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 selected. 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. After selecting a module, press Enter to start it.

Various buttons or selection fields in the module contain a highlighted letter (yellow by default). Use Alt–highlighted_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 1.3, "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 function keys (F1 ... 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).

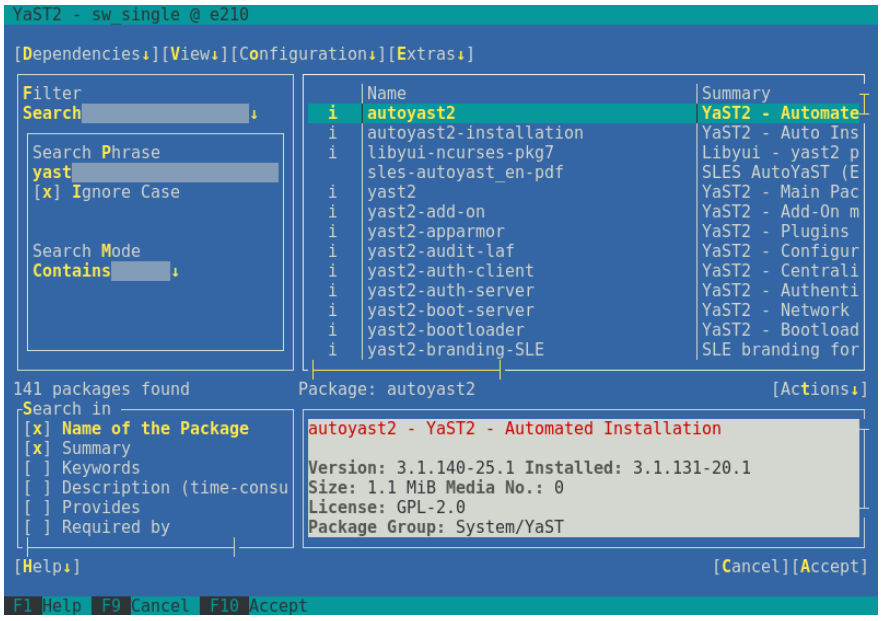


FIGURE 1.2: THE SOFTWARE INSTALLATION MODULE

1.2 Advanced Key Combinations

YaST in text mode has a set of advanced key combinations.

`Shift`–`F1`

List advanced hotkeys.

`Shift`–`F4`

Change color schema.

`Ctrl`–`\`

Quit the application.

`Ctrl`–`L`

Refresh screen.

`Ctrl`–`D` `F1`

List advanced hotkeys.

`Ctrl`–`D` `Shift`–`D`

Dump dialog to the log file as a screenshot.

`Ctrl`–`D` `Shift`–`Y`

Open YDialogSpy to see the widget hierarchy.

1.3 Restriction of Key Combinations

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

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 function keys (F1 ... F12) 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.

1.4 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:

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

1.4.1 Installing Packages from the Command Line

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

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

or

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

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

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"*.

1.4.2 Starting Individual Modules

To save time, you can start individual YaST modules directly. To start a module, enter:

```
tux > sudo 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`**.

1.4.3 Command Line Parameters of 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:

```
tux > sudo yast module_name help
```

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

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

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

1.4.3.1 Common YaST Module Commands

All YaST modules support the following commands:

help

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

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

longhelp

Same as **`help`**, but adds a detailed list of each command's options together with their description:

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

xmlhelp

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

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

interactive

If you need to spend more time querying a module's settings, run the *interactive* mode. The YaST shell opens, where you can enter all the module's commands without the **`sudo yast ...`** prefix. To leave the interactive mode, enter `exit`.

1.4.3.2 yast add-on

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

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

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

1.4.3.3 yast audit-laf

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

set

Sets an option:

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

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

show

Displays settings of an option:

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

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

1.4.3.4 yast dhcp-server

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

disable

Disables the DHCP server service.

enable

Enables the DHCP server service.

host

Configures settings for individual hosts.

interface

Specifies to which network interface to listen to:

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

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

options

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

status

Prints the status of the DHCP service.

subnet

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

1.4.3.5 yast dns-server

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

acls

Displays access control list settings:

```
tux > sudo yast dns-server acls show
ACLs:
-----
 Name       Type        Value
 ---------------------------
 any        Predefined
 localips   Predefined
 localnets  Predefined
```

```
  none       Predefined
```

dnsrecord

Configures zone resource records:

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

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

forwarders

Configures DNS forwarders:

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

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

host

Handles 'A' and its related 'PTR' record at once:

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

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

logging

Configures logging settings:

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

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

mailserver

Configures zone mail servers:

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

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

nameserver

Configures zone name servers:

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

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

soa

Configures the start of authority (SOA) record:

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

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

startup

Manages the DNS server service:

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

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

transport

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

zones

Manages DNS zones:

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

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

1.4.3.6 yast disk

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

disks

Lists all configured disks in the system:

```
tux > sudo yast disk list disks
Device   | Size       | FS Type | Mount Point | Label | Model
---------+------------+---------+-------------+-------+-------------
```

```
/dev/sda | 119.24 GiB |         |             |       | SSD 840
/dev/sdb |  60.84 GiB |         |             |       | WD1003FBYX-0
```

partitions

Lists all partitions in the system:

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

1.4.3.7 yast firewall

Displays information about the firewall settings. **yast firewall** accepts the following commands:

broadcast

Displays settings of broadcast packets.

disable

Disables firewall.

enable

Enables firewall.

interfaces

Displays the configuration related to network interfaces.

logging

Displays the logging settings.

masqredirect

Redirects requests to masqueraded IP.

masquerade

Displays the masquerading settings.

services

Displays information about allowed services, ports, and protocols.

startup

Displays start-up settings.

summary

Displays firewall configuration summary.

zones

Lists known firewall zones.

1.4.3.8 yast ftp-server

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

SSL, SSLv2, SSLv3, TLS

Controls secure connections via SSL up to SSL version 3, and TLS. SSL options are valid for the `vsftpd` only.

```
tux > sudo yast ftp-server SSLv2 enable
tux > sudo yast ftp-server TLS disable
```

access

Configures access permissions:

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

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

anon_access

Configures access permissions for anonymous users:

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

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

anon_dir

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

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

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

chroot

Controls *change root* environment (chroot):

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

idle-time

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

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

logging

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

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

max_clients

Specifies the maximum number of concurrently connected clients:

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

max_clients_ip

Specifies the maximum number of concurrently connected clients via IP:

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

max_rate_anon

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

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

max_rate_authen

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

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

port_range

Specifies the port range for passive connection replies:

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

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

show

Displays FTP server settings.

startup

Controls the FTP start-up method:

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

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

umask

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

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

welcome_message

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

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

1.4.3.9 yast http-server

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

configure

Configures the HTTP server host settings:

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

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

hosts

Configures virtual hosts:

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

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

listen

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

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

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

mode

Enables or disables the wizard mode:

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

modules

Controls the Apache2 server modules:

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

1.4.3.10 yast kdump

Configures `kdump` settings. For more information on `kdump`, refer to the *Book* "System Analysis and Tuning Guide", *Chapter 17 "Kexec and Kdump", Section 17.7 "Basic Kdump Configuration"*. **`yast kdump`** accepts the following commands:

copykernel

Copies the kernel into the dump directory.

customkernel

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

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

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

dumpformat

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

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

dumplevel

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

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

dumptarget

Specifies the destination for saving dump images:

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

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

immediatereboot

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

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

keepolddumps

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

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

kernelcommandline

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

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

kernelcommandlineappend

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

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

notificationcc

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

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

notificationto

Specifies an e-mail address for sending notification messages:

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

show

Displays `kdump` settings:

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

smtppass

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

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

smtpserver

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

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

smtpuser

Specifies the SMTP user name used for sending notification messages:

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

startup

Enables or disables start-up options:

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

1.4.3.11 yast keyboard

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

list

Lists all available keyboard layouts.

set

Activates new keyboard layout setting:

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

summary

Displays the current keyboard configuration.

1.4.3.12 yast lan

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

add

Configures a new network card:

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

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

delete

Deletes an existing network card:

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

edit

Changes the configuration of an existing network card:

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

list

Displays a summary of network card configuration:

```
tux > sudo yast lan list
id name,           bootproto
```

```
0 Ethernet Card 0, NONE
1 Network Bridge,  DHCP
```

1.4.3.13 yast language

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

list

Lists all available languages.

set

Specifies the main system languages and secondary languages as well:

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

1.4.3.14 yast mail

Displays the configuration of the mail system:

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

1.4.3.15 yast nfs

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

add

Adds a new NFS mount:

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

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

delete

Deletes an existing NFS mount:

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

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

edit

Changes an existing NFS mount:

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

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

list

Lists existing NFS mounts:

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

1.4.3.16 yast nfs-server

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

add

Adds a directory to export:

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

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

delete

Deletes a directory from the NFS export:

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

set

Specifies additional parameters for the NFS server:

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

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

start

Starts the NFS server service:

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

stop

Stops the NFS server service:

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

summary

Displays a summary of the NFS server configuration:

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

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

1.4.3.17 yast nis

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

configure

Changes global settings of a NIS client:

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

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

disable

Disables the NIS client:

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

enable

Enables your machine as NIS client:

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

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

find

Shows available NIS servers for a given domain:

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

summary

Displays a configuration summary of a NIS client.

1.4.3.18 yast nis-server

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

master

Configures a NIS master server:

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

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

slave

Configures a NIS slave server:

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

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

stop

Stops a NIS server:

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

summary

Displays a configuration summary of a NIS server:

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

1.4.3.19 yast proxy

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

authentication

Specifies the authentication options for proxy:

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

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

enable, disable

Enables or disables proxy settings.

set

Changes the current proxy settings:

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

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

summary

Displays proxy settings.

1.4.3.20 yast rdp

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

allow

Allows remote access to the server's desktop:

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

list

Displays the remote desktop configuration summary.

1.4.3.21 yast samba-client

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

configure

Changes global settings of Samba:

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

isdomainmember

Verifies if the machine is a member of a domain:

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

joindomain

Makes the machine a member of a domain:

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

winbind

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

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

1.4.3.22 yast samba-server

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

backend

Specifies the back-end for storing user information:

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

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

configure

Configures global settings of the Samba server:

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

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

list

Displays a list of available shares:

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

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

role

Specifies the role of the Samba server:

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

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

service

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

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

share

Manipulates a single Samba share:

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

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

1.4.3.23 yast security

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

level

Specifies the security level of the host:

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

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

set

Sets the value of specific options:

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

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

summary

Displays a summary of the current security configuration:

```
sudoyast security summary
```

1.4.3.24 yast sound

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

add

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

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

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

channels

Lists available volume channels of a sound card:

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

modules

Lists all available sound kernel modules:

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

playtest

Plays a test sound on a sound card:

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

remove

Removes a configured sound card:

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

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

set

Specifies new values for a sound card:

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

show

Displays detailed information about a sound card:

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

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

summary

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

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

volume

Specifies the volume level of a sound card:

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

1.4.3.25 yast sysconfig

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

clear

Sets empty value to a variable:

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

Tip: Variable in Multiple Files

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

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

details

Displays detailed information about a variable:

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

list

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

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

set

Sets a value to a variable:

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

Tip: Variable in Multiple Files

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

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

1.4.3.26 yast tftp-server

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

directory

Specifies the directory of the TFTP server:

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

status

Controls the status of the TFTP server service:

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

1.4.3.27 yast timezone

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

list

Lists all available time zones grouped by region:

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

set

Specifies new values for the time zone configuration:

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

summary

Displays the time zone configuration summary:

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

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

1.4.3.28 yast users

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

add

Adds a new user:

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

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

delete

Deletes an existing user account:

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

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

edit

Changes an existing user account:

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

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

list

Lists existing users filtered by user type:

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

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

show

Displays details about a user:

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

```
Login Shell: /sbin/nologin
Login Name: wwwrun
UID: 456
```

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

2 Managing Software with Command Line Tools

This chapter describes Zypper and RPM, two command line tools for managing software. For a definition of the terminology used in this context (for example, `repository`, `patch`, or `update`) refer to *Book "Start-Up", Chapter 10 "Installing or Removing Software", Section 10.1 "Definition of Terms"*.

2.1 Using Zypper

Zypper is a command line package manager for installing, updating and removing packages. It also manages repositories. It is especially useful for accomplishing remote software management tasks or managing software from shell scripts.

2.1.1 General Usage

The general syntax of Zypper is:

```
zypper [--global-options] COMMAND  [--command-options] [arguments]
```

The components enclosed in brackets are not required. See **`zypper help`** for a list of general options and all commands. To get help for a specific command, type **`zypper help`** *`COMMAND`*.

Zypper Commands

The simplest way to execute Zypper is to type its name, followed by a command. For example, to apply all needed patches to the system, use:

```
tux > sudo zypper patch
```

Global Options

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

```
tux > sudo zypper --non-interactive patch
```

In the above example, the option `--non-interactive` means that the command is run without asking anything (automatically applying the default answers).

Command-Specific Options

To use options that are specific to a particular command, type them immediately after the command:

```
tux > sudo zypper patch --auto-agree-with-licenses
```

In the above example, `--auto-agree-with-licenses` is used to apply all needed patches to a system without you being asked to confirm any licenses. Instead, license will be accepted automatically.

Arguments

Some commands require one or more arguments. For example, when using the command **`install`**, you need to specify which package or which packages you want to *install*:

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

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

```
tux > zypper search -t pattern
```

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

```
tux > sudo zypper -v install --from factory mc vim
```

The `--from` option 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.

```
tux > sudo zypper remove --dry-run MozillaFirefox
```

Zypper supports the global `--userdata` *`STRING`* option. You can specify a string with this option, which gets written to Zypper's log files and plug-ins (such as the Btrfs plug-in). It can be used to mark and identify transactions in log files.

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

2.1.2 Installing and Removing Software with Zypper

To install or remove packages, use the following commands:

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

```
sudo zypper remove PACKAGE_NAME
```

Warning: Do Not Remove Mandatory System Packages

Do not remove mandatory system packages like `glibc` , `zypper` , `kernel` . If they are removed, the system can become unstable or stop working altogether.

2.1.2.1 Selecting Which Packages to Install or Remove

There are various ways to address packages with the commands **`zypper install`** and **`zypper remove`**.

By Exact Package Name

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

By Exact Package Name and Version Number

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

By Repository Alias and Package Name

```
tux > sudo zypper install mozilla:MozillaFirefox
```

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

By Package Name Using Wild Cards

You can select all packages that have names starting or ending with a certain string. Use wild cards with care, especially when removing packages. The following command will install all packages starting with “Moz”:

```
tux > sudo zypper install 'Moz*'
```

Tip: Removing all `-debuginfo` Packages

When debugging a problem, you sometimes need to temporarily install a lot of `-debuginfo` packages which give you more information about running processes. After your debugging session finishes and you need to clean the environment, run the following:

```
tux > sudo zypper remove '*-debuginfo'
```

By Capability

For example, to install a package without knowing its name, capabilities come in handy. The following command will install the package `MozillaFirefox`:

```
tux > sudo zypper install firefox
```

By Capability, Hardware Architecture, or Version

Together with a capability, you can specify a hardware architecture and a version:

- The name of the desired hardware architecture is appended to the capability after a full stop. For example, to specify the AMD64/Intel 64 architectures (which in Zypper is named `x86_64`), use:

  ```
  tux > sudo zypper install 'firefox.x86_64'
  ```

- Versions must be appended to the end of the string and must be preceded by an operator: `<` (lesser than), `<=` (lesser than or equal), `=` (equal), `>=` (greater than or equal), `>` (greater than).

  ```
  tux > sudo zypper install 'firefox>=52.2'
  ```

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

  ```
  tux > sudo zypper install 'firefox.x86_64>=52.2'
  ```

By Path to the RPM file

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

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

2.1.2.2 Combining Installation and Removal of Packages

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

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

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

```
tux > sudo zypper remove emacs +vim
```

To prevent the package name starting with the `-` being interpreted as a command option, always use it as the second argument. If this is not possible, precede it with `--`:

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

2.1.2.3 Cleaning Up Dependencies of Removed Packages

If (together with a certain package), you automatically want to remove any packages that become unneeded after removing the specified package, use the `--clean-deps` option:

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

2.1.2.4 Using Zypper in Scripts

By default, Zypper asks for a confirmation before installing or removing a selected package, or when a problem occurs. You can override this behavior using the `--non-interactive` option. This option must be given before the actual command (`install`, `remove`, and `patch`), as can be seen in the following:

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

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

2.1.2.5 Installing or Downloading Source Packages

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

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

When executed as `root`, the default location to install source packages is `/usr/src/packages/` and `~/rpmbuild` when run as user. These values can be changed in your local **`rpm`** configuration.

This command will also install the build dependencies of the specified package. If you do not want this, add the switch `-D`:

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

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

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

Of course, this will only work if you have the repository with the source packages enabled in your repository list (it is added by default, but not enabled). See *Section 2.1.5, "Managing Repositories with Zypper"* for details on repository management.

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

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

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

```
tux > zypper source-download
```

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

2.1.2.6 Installing Packages from Disabled Repositories

Normally you can only install or refresh packages from enabled repositories. The `--plus-content TAG` option helps you specify repositories to be refreshed, temporarily enabled during the current Zypper session, and disabled after it completes.

For example, to enable repositories that may provide additional `-debuginfo` or `-debugsource` packages, use `--plus-content debug`. You can specify this option multiple times.

To temporarily enable such 'debug' repositories to install a specific `-debuginfo` package, use the option as follows:

```
tux > sudo zypper --plus-content debug \
   install "debuginfo(build-id)=eb844a5c20c70a59fc693cd1061f851fb7d046f4"
```

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

Note: Disabled Installation Media

Repositories from the openSUSE Leap installation media are still configured but disabled after successful installation. You can use the `--plus-content` option to install packages from the installation media instead of the online repositories. Before calling **`zypper`**, ensure the media is available, for example by inserting the DVD into the computer's drive.

2.1.2.7 Utilities

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

```
tux > zypper verify
```

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

```
tux > sudo zypper install-new-recommends
```

This command is very useful after plugging in a Web cam or Wi-Fi device. It will install drivers for the device and related software, if available. Drivers and related software are only installable if certain hardware dependencies are fulfilled.

2.1.3 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 13 "Upgrading the System and System Changes"*.

2.1.3.1 Installing All Needed Patches

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

```
tux > sudo zypper patch
```

All patches available from repositories configured on your computer are checked for their relevance to your installation. If they are relevant (and not classified as `optional` or `feature`), they are installed immediately.

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

The plain **`zypper patch`** command does not apply patches from third party repositories. To update also the third party repositories, use the `with-update` command option as follows:

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

To install also optional patches, use:

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

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

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

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

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

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

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

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

```
tux > sudo zypper patch --updatestack-only
```

Bear in mind that other command options that would also update other repositories will be dropped if you use the `updatestack-only` command option.

2.1.3.2 Listing Patches

To find out whether patches are available, Zypper allows viewing the following information:

Number of Needed Patches

To list the number of needed patches (patches that apply to your system but are not yet installed), use **`patch-check`**:

```
tux > zypper patch-check
```

```
Loading repository data...
Reading installed packages...
5 patches needed (1 security patch)
```

This command can be combined with the `--updatestack-only` option to list only the patches which affect Zypper and the package management itself.

List of Needed Patches

To list all needed patches (patches that apply to your system but are not yet installed), use **`list-patches`**:

```
tux > zypper list-patches
Repository | Name              | Category | Severity | Interactive | Status | S>
-----------+-------------------+----------+----------+-------------+--------+-->
Update     | openSUSE-2017-828 | security | moderate | ---         | needed | S>

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

List of All Patches

To list all patches available for openSUSE Leap, regardless of whether they are already installed or apply to your installation, use **`zypper patches`**.

It is also possible to list and install patches relevant to specific issues. To list specific patches, use the **`zypper list-patches`** command with the following options:

By Bugzilla Issues

To list all needed patches that relate to Bugzilla issues, use the option `--bugzilla`.
To list patches for a specific bug, you can also specify a bug number: `--bugzilla=`*`NUMBER`*.
To search for patches relating to multiple Bugzilla issues, add commas between the bug numbers, for example:

```
tux > zypper list-patches --bugzilla=972197,956917
```

By CVE Number

To list all needed patches that relate to an entry in the CVE database (Common Vulnerabilities and Exposures), use the option `--cve`.
To list patches for a specific CVE database entry, you can also specify a CVE number: `--cve=`*`NUMBER`*. To search for patches relating to multiple CVE database entries, add commas between the CVE numbers, for example:

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

To list all patches regardless of whether they are needed, use the option `--all` additionally. For example, to list all patches with a CVE number assigned, use:

```
tux > zypper list-patches --all --cve
Issue | No.           | Patch              | Category    | Severity  | Status
------+---------------+--------------------+-------------+-----------+----------
cve   | CVE-2015-0287 | SUSE-SLE-Module.. | recommended | moderate  | needed
cve   | CVE-2014-3566 | SUSE-SLE-SERVER.. | recommended | moderate  | not needed
[...]
```

2.1.3.3 Installing New Package Versions

If a repository contains only new packages, but does not provide patches, **`zypper patch`** does not show any effect. To update all installed packages with newer available versions (while maintaining system integrity), use:

```
tux > sudo zypper update
```

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

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

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

```
tux > zypper list-updates
```

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

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

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

```
tux > sudo zypper list-updates --all
```

To find out why a new package cannot be installed, use the **`zypper install`** or **`zypper update`** command as described above.

2.1.3.4 Identifying Orphaned Packages

Whenever you remove a repository from Zypper or upgrade your system, some packages can get in an "orphaned" state. These *orphaned* packages belong to no active repository anymore. The following command gives you a list of these:

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

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

2.1.4 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 list processes using deleted files. In case the process belongs to a known service, the service name is listed, making it easy to restart the service. By default **`zypper ps`** shows a table:

```
tux > zypper ps
PID   | PPID | UID | User   | Command      | Service      | Files
------+------+-----+--------+--------------+--------------+-------------------
814   | 1    | 481 | avahi  | avahi-daemon | avahi-daemon | /lib64/ld-2.19.s->
      |      |     |        |              |              | /lib64/libdl-2.1->
      |      |     |        |              |              | /lib64/libpthrea->
      |      |     |        |              |              | /lib64/libc-2.19->
[...]
```

PID: ID of the process
PPID: ID of the parent process
UID: ID of the user running the process
Login: Login name of the user running the process
Command: Command used to execute the process
Service: Service name (only if command is associated with a system service)
Files: The list of the deleted files

The output format of **`zypper ps`** can be controlled as follows:

`zypper ps -s`

Create a short table not showing the deleted files.

```
tux > zypper ps -s
PID   | PPID | UID  | User    | Command      | Service
------+------+------+---------+--------------+--------------
```

```
814   | 1    | 481  | avahi   | avahi-daemon | avahi-daemon
817   | 1    | 0    | root    | irqbalance   | irqbalance
1567  | 1    | 0    | root    | sshd         | sshd
1761  | 1    | 0    | root    | master       | postfix
1764  | 1761 | 51   | postfix | pickup       | postfix
1765  | 1761 | 51   | postfix | qmgr         | postfix
2031  | 2027 | 1000 | tux     | bash         |
```

zypper ps -ss

Show only processes associated with a system service.

```
PID   | PPID | UID  | User    | Command      | Service
------+------+------+---------+--------------+--------------
814   | 1    | 481  | avahi   | avahi-daemon | avahi-daemon
817   | 1    | 0    | root    | irqbalance   | irqbalance
1567  | 1    | 0    | root    | sshd         | sshd
1761  | 1    | 0    | root    | master       | postfix
1764  | 1761 | 51   | postfix | pickup       | postfix
1765  | 1761 | 51   | postfix | qmgr         | postfix
```

zypper ps -sss

Only show system services using deleted files.

```
avahi-daemon
irqbalance
postfix
sshd
```

zypper ps --print "systemctl status %s"

Show the commands to retrieve status information for services which might need a restart.

```
systemctl status avahi-daemon
systemctl status irqbalance
systemctl status postfix
systemctl status sshd
```

For more information about service handling refer to *Chapter 10, The* `systemd` *Daemon*.

2.1.5 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:

```
tux > zypper repos
```

The result will look similar to the following output:

EXAMPLE 2.1: ZYPPER—LIST OF KNOWN REPOSITORIES

```
tux > zypper repos
#  | Alias                 | Name             | Enabled | GPG Check | Refresh
---+-----------------------+------------------+---------+-----------+--------
 1 | Leap-42.3-Main        | Main (OSS)       | Yes     | (r ) Yes  | Yes
 2 | Leap-42.3-Update      | Update (OSS)     | Yes     | (r ) Yes  | Yes
 3 | Leap-42.3-NOSS        | Main (NON-OSS)   | Yes     | (r ) Yes  | Yes
 4 | Leap-42.3-Update-NOSS | Update (NON-OSS) | Yes     | (r ) Yes  | Yes
[...]
```

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

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

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

2.1.5.1 Adding Repositories

To add a repository, run

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

URI can either be an Internet repository, a network resource, a directory or a CD or DVD (see http://en.opensuse.org/openSUSE:Libzypp_URIs for details). The *ALIAS* is a shorthand and unique identifier of the repository. You can freely choose it, with the only exception that it needs to be unique. Zypper will issue a warning if you specify an alias that is already in use.

2.1.5.2 Refreshing Repositories

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

```
tux > sudo zypper refresh
```

Note: Default Behavior of `zypper`

By default, some commands perform `refresh` automatically, so you do not need to run the command explicitly.

The `refresh` command enables you to view changes also in disabled repositories, by using the `--plus-content` option:

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

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

2.1.5.3 Removing Repositories

To remove a repository from the list, use the command `zypper removerepo` together with the alias or number of the repository you want to delete. For example, to remove the repository `Leap-42.3-NOSS` from *Example 2.1, "Zypper—List of Known Repositories"*, use one of the following commands:

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

2.1.5.4 Modifying Repositories

Enable or disable repositories with `zypper modifyrepo`. You can also alter the repository's properties (such as refreshing behavior, name or priority) with this command. The following command will enable the repository named `updates`, turn on auto-refresh and set its priority to 20:

```
tux > sudo zypper modifyrepo -er -p 20 'updates'
```

Modifying repositories is not limited to a single repository—you can also operate on groups:

`-a`: all repositories
`-l`: local repositories
`-t`: remote repositories
`-m` *TYPE*: repositories of a certain type (where *TYPE* can be one of the following: `http`, `https`, `ftp`, `cd`, `dvd`, `dir`, `file`, `cifs`, `smb`, `nfs`, `hd`, `iso`)

To rename a repository alias, use the `renamerepo` command. The following example changes the alias from `Mozilla Firefox` to `firefox`:

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

2.1.6 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:

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

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

2.1.6.1 Searching for Software

The **`zypper search`** command works on package names, or, optionally, on package summaries and descriptions. Strings wrapped in `/` are interpreted as regular expressions. By default, the search is not case-sensitive.

Simple search for a package name containing `fire`

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

Simple search for the exact package `MozillaFirefox`

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

Also search in package descriptions and summaries

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

Only display packages not already installed

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

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

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

2.1.6.2 Searching for Specific Capability

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

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

The `what-provides` *`PACKAGE_NAME`* is similar to **`rpm -q --whatprovides`** *`PACKAGE_NAME`*, but RPM is only able to query the RPM database (that is the database of all installed packages). Zypper, on the other hand, will tell you about providers of the capability from any repository, not only those that are installed.

2.1.6.3 Showing Package Information

To query single packages, use **`info`** with an exact package name as an argument. This displays detailed information about a package. In case the package name does not match any package name from repositories, the command outputs detailed information for non-package matches. If you request a specific type (by using the `-t` option) and the type does not exist, the command outputs other available matches but without detailed information.

If you specify a source package, the command displays binary packages built from the source package. If you specify a binary package, the command outputs the source packages used to build the binary package.

To also show what is required/recommended by the package, use the options `--requires` and `--recommends`:

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

2.1.7 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

If you have trouble accessing packages from configured repositories (for example, Zypper cannot find a certain package even though you know it exists in one of the repositories), refreshing the repositories may help:

```
tux > sudo zypper refresh
```

If that does not help, try

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

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

2.1.9 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`** when committing changes to the file system to create appropriate file system snapshots. These snapshots can be used to revert any changes made by Zypper. See *Chapter 3, System Recovery and Snapshot Management with Snapper* for more information.

2.1.10 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, `cheat sheets` with the most important commands, and information on how to use Zypper in scripts and applications, refer to http://en.opensuse.org/SDB:Zypper_usage. A list of software changes for the latest openSUSE Leap version can be found at http://en.opensuse.org/openSUSE:Zypper versions.

2.2 RPM—the Package Manager

RPM (RPM Package Manager) is used for managing software packages. Its main commands are **`rpm`** and **`rpmbuild`**. The powerful RPM database can be queried by the users, system administrators and package builders for detailed information about the installed software.

rpm has five modes: installing, uninstalling (or updating) software packages, rebuilding the RPM database, querying RPM bases or individual RPM archives, integrity checking of packages and signing packages. **rpmbuild** can be used to build installable packages from pristine sources.

Installable RPM archives are packed in a special binary format. These archives consist of the program files to install and certain meta information used during the installation by **rpm** to configure the software package or stored in the RPM database for documentation purposes. RPM archives normally have the extension `.rpm`.

Tip: Software Development Packages

For several packages, the components needed for software development (libraries, headers, include files, etc.) have been put into separate packages. These development packages are only needed if you want to compile software yourself (for example, the most recent GNOME packages). They can be identified by the name extension `-devel`, such as the packages `alsa-devel` and `gimp-devel`.

2.2.1 Verifying Package Authenticity

RPM packages have a GPG signature. To verify the signature of an RPM package, use the command **rpm --checksig** *PACKAGE*-1.2.3.rpm to determine whether the package originates from SUSE or from another trustworthy facility. This is especially recommended for update packages from the Internet.

2.2.2 Managing Packages: Install, Update, and Uninstall

Normally, the installation of an RPM archive is quite simple: **rpm -i** *PACKAGE*.rpm. With this command the package is installed, but only if its dependencies are fulfilled and if there are no conflicts with other packages. With an error message, **rpm** requests those packages that need to be installed to meet dependency requirements. In the background, the RPM database ensures that no conflicts arise—a specific file can only belong to one package. By choosing different options, you can force **rpm** to ignore these defaults, but this is only for experts. Otherwise, you risk compromising the integrity of the system and possibly jeopardize the ability to update the system.

The options `-U` or `--upgrade` and `-F` or `--freshen` can be used to update a package (for example, **`rpm -F`** *PACKAGE*.rpm). This command removes the files of the old version and immediately installs the new files. The difference between the two versions is that `-U` installs packages that previously did not exist in the system, while `-F` merely updates previously installed packages. When updating, **`rpm`** updates configuration files carefully using the following strategy:

- If a configuration file was not changed by the system administrator, **`rpm`** installs the new version of the appropriate file. No action by the system administrator is required.
- If a configuration file was changed by the system administrator before the update, **`rpm`** saves the changed file with the extension `.rpmorig` or `.rpmsave` (backup file) and installs the version from the new package. This is done only if the originally installed file and the newer version are different. If this is the case, compare the backup file (`.rpmorig` or `.rpmsave`) with the newly installed file and make your changes again in the new file. Afterward, delete all `.rpmorig` and `.rpmsave` files to avoid problems with future updates.
- `.rpmnew` files appear if the configuration file already exists *and* if the `noreplace` label was specified in the `.spec` file.

Following an update, `.rpmsave` and `.rpmnew` files should be removed after comparing them, so they do not obstruct future updates. The `.rpmorig` extension is assigned if the file has not previously been recognized by the RPM database.

Otherwise, `.rpmsave` is used. In other words, `.rpmorig` results from updating from a foreign format to RPM. `.rpmsave` results from updating from an older RPM to a newer RPM. `.rpmnew` does not disclose any information to whether the system administrator has made any changes to the configuration file. A list of these files is available in `/var/adm/rpmconfigcheck`. Some configuration files (like `/etc/httpd/httpd.conf`) are not overwritten to allow continued operation.

The `-U` switch is *not* only an equivalent to uninstalling with the `-e` option and installing with the `-i` option. Use `-U` whenever possible.

To remove a package, enter **`rpm -e`** *PACKAGE*. This command only deletes the package if there are no unresolved dependencies. It is theoretically impossible to delete Tcl/Tk, for example, as long as another application requires it. Even in this case, RPM calls for assistance from the database. If such a deletion is, for whatever reason, impossible (even if *no* additional dependencies exist), it may be helpful to rebuild the RPM database using the option `--rebuilddb`.

2.2.3 Delta RPM Packages

Delta RPM packages contain the difference between an old and a new version of an RPM package. Applying a delta RPM onto an old RPM results in a completely new RPM. It is not necessary to have a copy of the old RPM because a delta RPM can also work with an installed RPM. The delta RPM packages are even smaller in size than patch RPMs, which is an advantage when transferring update packages over the Internet. The drawback is that update operations with delta RPMs involved consume considerably more CPU cycles than plain or patch RPMs.

The **makedeltarpm** and **applydelta** binaries are part of the delta RPM suite (package `deltarpm`) and help you create and apply delta RPM packages. With the following commands, you can create a delta RPM called `new.delta.rpm`. The following command assumes that `old.rpm` and `new.rpm` are present:

```
tux > sudo makedeltarpm old.rpm new.rpm new.delta.rpm
```

Using **applydeltarpm**, you can reconstruct the new RPM from the file system if the old package is already installed:

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

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

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

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

2.2.4 RPM Queries

With the `-q` option **rpm** initiates queries, making it possible to inspect an RPM archive (by adding the option `-p`) and to query the RPM database of installed packages. Several switches are available to specify the type of information required. See *Table 2.1, "The Most Important RPM Query Options"*.

TABLE 2.1: THE MOST IMPORTANT RPM QUERY OPTIONS

`-i`	Package information
`-l`	File list

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

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

EXAMPLE 2.2: **`rpm -q -i wget`**

```
Name        : wget
Version     : 1.14
Release     : 10.3
Architecture: x86_64
Install Date: Fri 14 Jul 2017 04:09:58 PM CEST
Group       : Productivity/Networking/Web/Utilities
Size        : 2046452
License     : GPL-3.0+
Signature   : RSA/SHA256, Wed 10 May 2017 02:40:21 AM CEST, Key ID b88b2fd43dbdc284
Source RPM  : wget-1.14-10.3.src.rpm
Build Date  : Wed 10 May 2017 02:40:12 AM CEST
Build Host  : lamb55
Relocations : (not relocatable)
Packager    : http://bugs.opensuse.org
Vendor      : openSUSE
URL         : http://www.gnu.org/software/wget/
Summary     : A Tool for Mirroring FTP and HTTP Servers
Description :
Wget enables you to retrieve WWW documents or FTP files from a server.
```

```
This can be done in script files or via the command line.
Distribution: openSUSE Leap 42.3
```

The option `-f` only works if you specify the complete file name with its full path. Provide as many file names as desired. For example:

```
tux > rpm -q -f /bin/rpm /usr/bin/wget
rpm-4.11.2-15.1.x86_64
wget-1.14-17.1.x86_64
```

If only part of the file name is known, use a shell script as shown in *Example 2.3, "Script to Search for Packages"*. Pass the partial file name to the script shown as a parameter when running it.

EXAMPLE 2.3: SCRIPT TO SEARCH FOR PACKAGES

```
#! /bin/sh
for i in $(rpm -q -a -l | grep $1); do
    echo "\"$i\" is in package:"
    rpm -q -f $i
    echo ""
done
```

The command **`rpm -q --changelog`** *PACKAGE* displays a detailed list of change information about a specific package, sorted by date.

With the installed RPM database, verification checks can be made. Initiate these with `-V`, or `--verify`. With this option, **`rpm`** shows all files in a package that have been changed since installation. **`rpm`** uses eight character symbols to give some hints about the following changes:

TABLE 2.2: RPM VERIFY OPTIONS

`5`	MD5 check sum
`S`	File size
`L`	Symbolic link
`T`	Modification time
`D`	Major and minor device numbers
`U`	Owner
`G`	Group
`M`	Mode (permissions and file type)

In the case of configuration files, the letter `c` is printed. For example, for changes to `/etc/wgetrc` (`wget` package):

```
tux > rpm -V wget
S.5....T c /etc/wgetrc
```

The files of the RPM database are placed in `/var/lib/rpm`. If the partition `/usr` has a size of 1 GB, this database can occupy nearly 30 MB, especially after a complete update. If the database is much larger than expected, it is useful to rebuild the database with the option `--rebuilddb`. Before doing this, make a backup of the old database. The **`cron`** script **`cron.daily`** makes daily copies of the database (packed with gzip) and stores them in `/var/adm/backup/rpmdb`. The number of copies is controlled by the variable `MAX_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.

Keep in mind, the `BuildRoot` directive in the spec file is deprecated since openSUSE Leap 42.1. If you still need this feature, use the `--buildroot` option as a workaround.

2.2.6 Compiling RPM Packages with build

The danger with many packages is that unwanted files are added to the running system during the build process. To prevent this use `build`, which creates a defined environment in which the package is built. To establish this chroot environment, the **`build`** script must be provided with a complete package tree. This tree can be made available on the hard disk, via NFS, or from DVD. Set the position with **`build --rpms`** *`DIRECTORY`*. Unlike **`rpm`**, the **`build`** command looks for the `.spec` file in the source directory. To build `wget` (like in the above example) with the DVD mounted in the system under `/media/dvd`, use the following commands as `root`:

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

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

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

2.2.7 Tools for RPM Archives and the RPM Database

Midnight Commander (**`mc`**) can display the contents of RPM archives and copy parts of them. It represents archives as virtual file systems, offering all usual menu options of Midnight Commander. Display the `HEADER` with `F3`. View the archive structure with the cursor keys and `Enter`. Copy archive components with `F5`.

A full-featured package manager is available as a YaST module. For details, see *Book "Start-Up", Chapter 10 "Installing or Removing Software"*.

3 System Recovery and Snapshot Management with Snapper

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, 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 openSUSE Leap 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 16 GB). Taking snapshots on partitions other than `/` is not enabled by default.

Tip: Enabling Snapper in the Installed System

If you disabled Snapper during the installation, you can enable it at any time later. To do so, create a default Snapper configuration for the root file system by running

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

Afterward enable the different snapshot types as described in *Section 3.1.3.1, "Disabling/Enabling Snapshots"*.

Keep in mind that snapshots require a Btrfs root file system with subvolumes set up as proposed by the installer and a partition size of at least 16 GB.

When a snapshot is created, both the snapshot and the original point to the same blocks in the file system. So, initially a snapshot does not occupy additional disk space. If data in the original file system is modified, changed data blocks are copied while the old data blocks are kept for the snapshot. Therefore, a snapshot occupies the same amount of space as the data modified. So, over time, the amount of space a snapshot allocates, constantly grows. As a consequence, deleting files from a `Btrfs` file system containing snapshots may *not* free disk space!

Note: Snapshot Location

Snapshots always reside on the same partition or subvolume on which the snapshot has been taken. It is not possible to store snapshots on a different partition or subvolume.

As a result, partitions containing snapshots need to be larger than "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. To prevent disks from running out of space, old snapshots are automatically cleaned up. Refer to *Section 3.1.3.4, "Controlling Snapshot Archiving"* for details.

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 events that trigger them:

Timeline Snapshots

A single snapshot is created every hour. Old snapshots are automatically deleted. By default, the first snapshot of the last ten days, months, and years are kept. Timeline snapshots are disabled by default.

Installation Snapshots

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

Administration Snapshots

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

3.1.2 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`, `/boot/grub2/powerpc-ieee1275`, `/boot/grub2/s390x-emu`

A rollback of the boot loader configuration is not supported. The directories listed above are architecture-specific. The first two directories are present on AMD64/Intel 64 machines, the latter two on IBM POWER and on IBM Z, respectively.

`/home`

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

`/opt`, `/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/cache`, `/var/crash`

All directories containing temporary files and caches are excluded from snapshots.

`/usr/local`

This directory is used when manually installing software. It is excluded to avoid uninstalling these installations on rollbacks.

`/var/lib/libvirt/images`

The default location for virtual machine images managed with libvirt. Excluded to ensure virtual machine images are not replaced with older versions during a rollback. By default, this subvolume is created with the option `no copy on write`.

`/var/lib/mailman`, `/var/spool`

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

`/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/mariadb`, `/var/lib/mysql`, `/var/lib/pgqsl`

These directories contain database data. By default, these subvolumes are created with the option `no copy on write`.

`/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`).

2 If the given pattern matches and the corresponding package is marked as important (for example kernel packages), the snapshot will also be marked as important.

3 Pattern to match a package name. Based on the setting of the `match` attribute, special characters are either interpreted as shell wild cards or regular expressions. This pattern matches all package names starting with `kernel-`.

4 This line unconditionally matches all packages.

With this configuration snapshot, pairs are made whenever a package is installed (line 9). When the kernel, dracut, glibc, systemd, or udev packages marked as important are installed, the snapshot pair will also be marked as important (lines 4 to 8). All rules are evaluated.

To disable a rule, either delete it or deactivate it using XML comments. To prevent the system from making snapshot pairs for every package installation for example, comment line 9:

```
 1 <?xml version="1.0" encoding="utf-8"?>
 2 <snapper-zypp-plugin-conf>
 3  <solvables>
 4   <solvable match="w" important="true">kernel-*</solvable>
 5   <solvable match="w" important="true">dracut</solvable>
 6   <solvable match="w" important="true">glibc</solvable>
 7   <solvable match="w" important="true">systemd*</solvable>
 8   <solvable match="w" important="true">udev</solvable>
 9   <!-- <solvable match="w">*</solvable> -->
10  </solvables>
11 </snapper-zypp-plugin-conf>
```

3.1.3.3 Creating and Mounting New Subvolumes

Creating a new subvolume underneath the `/` hierarchy and permanently mounting it is supported. Such a subvolume will be excluded from snapshots. You need to make sure not to create it inside an existing snapshot, since you would not be able to delete snapshots anymore after a rollback.

openSUSE Leap is configured with the `/@/` subvolume which serves as an independent root for permanent subvolumes such as `/opt`, `/srv`, `/home` and others. Any new subvolumes you create and permanently mount need to be created in this initial root file system.

To do so, run the following commands. In this example, a new subvolume `/usr/important` is created from `/dev/sda2`.

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

```
tux > sudo umount /mnt
```

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

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

Tip: Disable Copy-On-Write (cow)

A subvolume may contain files that constantly change, such as virtualized disk images, database files, or log files. If so, consider disabling the copy-on-write feature for this volume, to avoid duplication of disk blocks. Use the `nodatacow` mount option in `/etc/fstab` to do so:

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

To alternatively disable copy-on-write for single files or directories, use the command **`chattr +C`** *PATH*.

3.1.3.4 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, up to ten important installation and administration snapshots and up to ten regular installation and administration snapshots are kept. If these snapshots occupy more than 50% of the root file system size, additional snapshots will be deleted. A minimum of four important and two regular snapshots are always kept.

Refer to *Section 3.4.1, "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`, `etx4` or `xfs` are valid values for *`FILESYSTEM`*. Example:

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

You can adjust this configuration according to your needs as described in *Section 3.4.1, "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 Compared to Rollback

When working with snapshots to restore data, it is important to know that there are two fundamentally different scenarios Snapper can handle:

Undoing Changes

When undoing changes as described in the following, two snapshots are being compared and the changes between these two snapshots are made undone. Using this method also allows to explicitly select the files that should be restored.

Rollback

When doing rollbacks as described in *Section 3.3, "System Rollback by Booting from Snapshots"*, the system is reset to the state at which the snapshot was taken.

When undoing changes, it is also possible to compare a snapshot against the current system. When restoring *all* files from such a comparison, this will have the same result as doing a rollback. However, using the method described in *Section 3.3, "System Rollback by Booting from Snapshots"* for rollbacks should be preferred, since it is faster and allows you to review the system before doing the rollback.

Warning: Data Consistency

There is no mechanism to ensure data consistency when creating a snapshot. Whenever a file (for example, a database) is written at the same time as the snapshot is being created, it will result in a corrupted or partly written file. Restoring such a file will cause problems. Furthermore, some system files such as `/etc/mtab` must never be restored. Therefore it is strongly recommended to *always* closely review the list of changed files and their diffs. Only restore files that really belong to the action you want to revert.

3.2.1 Undoing YaST and Zypper Changes

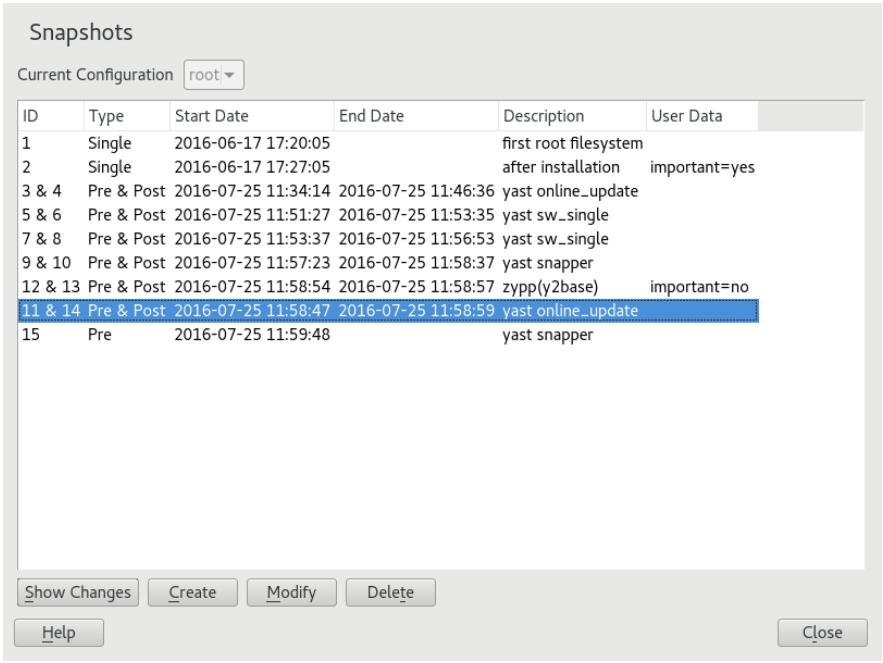
If you set up the root partition with `Btrfs` during the installation, Snapper—preconfigured for doing rollbacks of YaST or Zypper changes—will automatically be installed. Every time you start a YaST module or a Zypper transaction, two snapshots are created: a "pre-snapshot" capturing the state of the file system before the start of the module and a "post-snapshot" after the module has been finished.

Using the YaST Snapper module or the **`snapper`** command line tool, you can undo the changes made by YaST/Zypper by restoring files from the "pre-snapshot". Comparing two snapshots the tools also allow you to see which files have been changed. You can also display the differences between two versions of a file (diff).

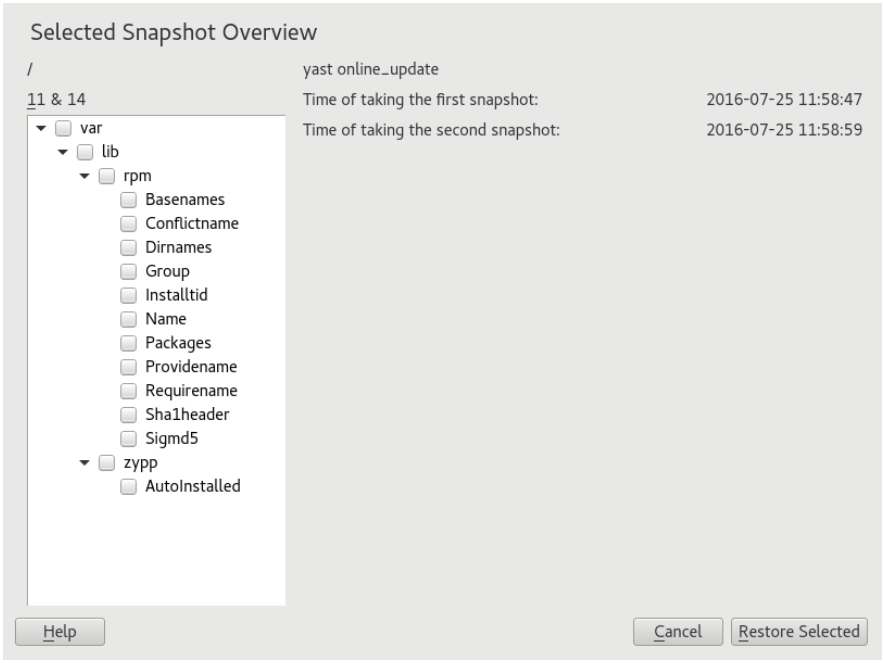
PROCEDURE 3.1: UNDOING CHANGES USING THE YAST *SNAPPER* MODULE

1. Start the *Snapper* module from the *Miscellaneous* section in YaST or by entering **`yast2 snapper`**.
2. Make sure *Current Configuration* is set to *root*. This is always the case unless you have manually added own Snapper configurations.

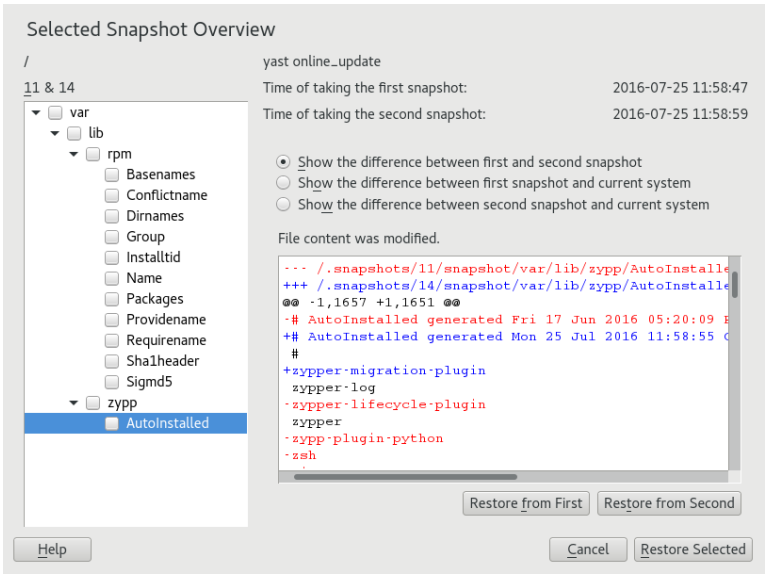
3. Choose a pair of pre- and post-snapshots from the list. Both, YaST and Zypper snapshot pairs are of the type *Pre & Post*. YaST snapshots are labeled as `zypp(y2base)` in the *Description column*; Zypper snapshots are labeled `zypp(zypper)`.



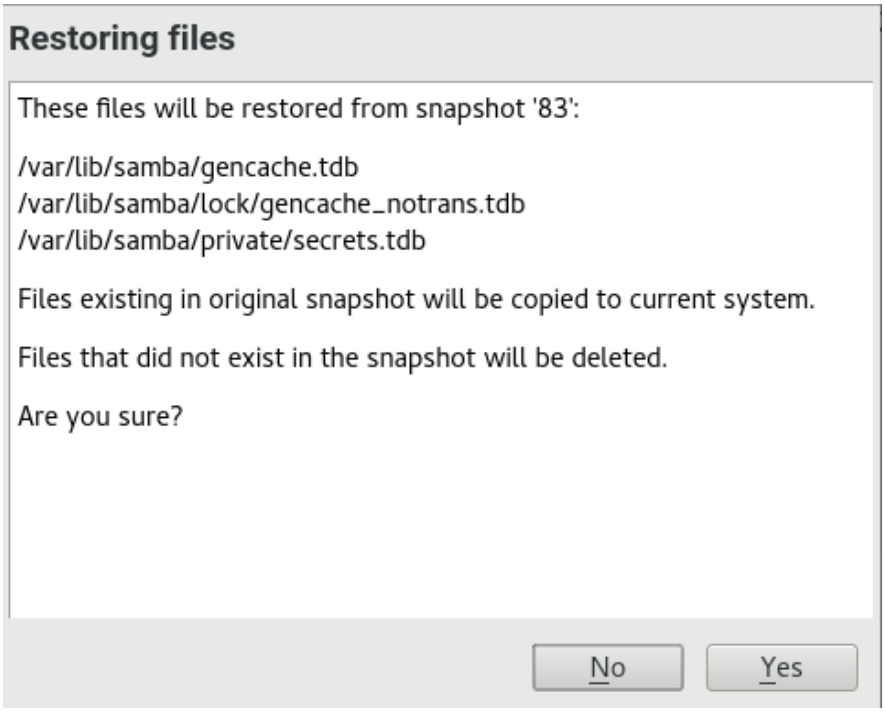
4. Click *Show Changes* to open the list of files that differ between the two snapshots.



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



6. To restore one or more files, select the relevant files or directories by activating the respective check box. Click *Restore Selected* and confirm the action by clicking *Yes*.



To restore a single file, activate its diff view by clicking its name. Click *Restore From First* and confirm your choice with *Yes*.

PROCEDURE 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)`.

```
tux > sudo snapper list -t pre-post
Pre # | Post # | Pre Date                     | Post Date                    | Description
------+--------+------------------------------+------------------------------+-------------
311   | 312    | Tue 06 May 2018 14:05:46 CEST | Tue 06 May 2018 14:05:52 CEST | zypp(y2base)
340   | 341    | Wed 07 May 2018 16:15:10 CEST | Wed 07 May 2018 16:15:16 CEST | zypp(zypper)
342   | 343    | Wed 07 May 2018 16:20:38 CEST | Wed 07 May 2018 16:20:42 CEST | zypp(y2base)
344   | 345    | Wed 07 May 2018 16:21:23 CEST | Wed 07 May 2018 16:21:24 CEST | zypp(zypper)
346   | 347    | Wed 07 May 2018 16:41:06 CEST | Wed 07 May 2018 16:41:10 CEST | zypp(y2base)
348   | 349    | Wed 07 May 2018 16:44:50 CEST | Wed 07 May 2018 16:44:53 CEST | zypp(y2base)
350   | 351    | Wed 07 May 2018 16:46:27 CEST | Wed 07 May 2018 16:46:38 CEST | zypp(y2base)
```

2. Get a list of changed files for a snapshot pair with **`snapper status`** *`PRE`*..*`POST`*. Files with content changes are marked with *c*, files that have been added are marked with + and deleted files are marked with -.

```
tux > sudo snapper status 350..351
+..... /usr/share/doc/packages/mikachan-fonts
+..... /usr/share/doc/packages/mikachan-fonts/COPYING
+..... /usr/share/doc/packages/mikachan-fonts/dl.html
c..... /usr/share/fonts/truetype/fonts.dir
c..... /usr/share/fonts/truetype/fonts.scale
+..... /usr/share/fonts/truetype/#####-p.ttf
+..... /usr/share/fonts/truetype/#####-pb.ttf
+..... /usr/share/fonts/truetype/#####-ps.ttf
+..... /usr/share/fonts/truetype/#####.ttf
c..... /var/cache/fontconfig/7ef2298fde41cc6eeb7af42e48b7d293-x86_64.cache-4
c..... /var/lib/rpm/Basenames
c..... /var/lib/rpm/Dirnames
c..... /var/lib/rpm/Group
c..... /var/lib/rpm/Installtid
c..... /var/lib/rpm/Name
c..... /var/lib/rpm/Packages
c..... /var/lib/rpm/Providename
c..... /var/lib/rpm/Requirename
c..... /var/lib/rpm/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.

```
tux > sudo snapper diff 350..351 /usr/share/fonts/truetype/fonts.scale
```

```
--- /.snapshots/350/snapshot/usr/share/fonts/truetype/fonts.scale       2014-04-23
 15:58:57.000000000 +0200
+++ /.snapshots/351/snapshot/usr/share/fonts/truetype/fonts.scale       2014-05-07
 16:46:31.000000000 +0200
@@ -1,4 +1,4 @@
-1174
+1486
 ds=y:ai=0.2:luximr.ttf -b&h-luxi mono-bold-i-normal--0-0-0-0-c-0-iso10646-1
 ds=y:ai=0.2:luximr.ttf -b&h-luxi mono-bold-i-normal--0-0-0-0-c-0-iso8859-1
[...]
```

4. To restore one or more files run **`snapper -v undochange`** *PRE*..*POST* *FILENAMES*. If you do not specify a *FILENAMES*, all changed files will be restored.

```
tux > sudo snapper -v undochange 350..351
     create:0 modify:13 delete:7
     undoing change...
     deleting /usr/share/doc/packages/mikachan-fonts
     deleting /usr/share/doc/packages/mikachan-fonts/COPYING
     deleting /usr/share/doc/packages/mikachan-fonts/dl.html
     deleting /usr/share/fonts/truetype/#####-p.ttf
     deleting /usr/share/fonts/truetype/#####-pb.ttf
     deleting /usr/share/fonts/truetype/#####-ps.ttf
     deleting /usr/share/fonts/truetype/#####.ttf
     modifying /usr/share/fonts/truetype/fonts.dir
     modifying /usr/share/fonts/truetype/fonts.scale
     modifying /var/cache/fontconfig/7ef2298fde41cc6eeb7af42e48b7d293-x86_64.cache-4
     modifying /var/lib/rpm/Basenames
     modifying /var/lib/rpm/Dirnames
     modifying /var/lib/rpm/Group
     modifying /var/lib/rpm/Installtid
     modifying /var/lib/rpm/Name
     modifying /var/lib/rpm/Packages
     modifying /var/lib/rpm/Providename
     modifying /var/lib/rpm/Requirename
     modifying /var/lib/rpm/Sha1header
     modifying /var/lib/rpm/Sigmd5
     undoing change done
```

Warning: Reverting User Additions

Reverting user additions via undoing changes with Snapper is not recommended. Since certain directories are excluded from snapshots, files belonging to these users will remain in the file system. If a user with the same user ID as a deleted user is created, this user will inherit the files. Therefore it is strongly recommended to use the YaST *User and Group Management* tool to remove users.

3.2.2 Using Snapper to Restore Files

Apart from the installation and administration snapshots, Snapper creates timeline snapshots. You can use these backup snapshots to restore files that have accidentally been deleted or to restore a previous version of a file. By using Snapper's diff feature you can also find out which modifications have been made at a certain point of time.

Being able to restore files is especially interesting for data, which may reside on subvolumes or partitions for which snapshots are not taken by default. To be able to restore files from home directories, for example, create a separate Snapper configuration for `/home` doing automatic timeline snapshots. See *Section 3.4, "Creating and Modifying Snapper Configurations"* for instructions.

Warning: Restoring Files Compared to Rollback

Snapshots taken from the root file system (defined by Snapper's root configuration), can be used to do a system rollback. The recommended way to do such a rollback is to boot from the snapshot and then perform the rollback. See *Section 3.3, "System Rollback by Booting from Snapshots"* for details.

Performing a rollback would also be possible by restoring all files from a root file system snapshot as described below. However, this is not recommended. You may restore single files, for example a configuration file from the `/etc` directory, but not the complete list of files from the snapshot.

This restriction only affects snapshots taken from the root file system!

PROCEDURE 3.3: RESTORING FILES USING THE YAST *SNAPPER* MODULE

1. Start the *Snapper* module from the *Miscellaneous* section in YaST or by entering **`yast2 snapper`**.
2. Choose the *Current Configuration* from which to choose a snapshot.

3. Select a timeline snapshot from which to restore a file and choose *Show Changes*. Timeline snapshots are of the type *Single* with a description value of *timeline*.

4. Select a file from the text box by clicking the file name. The difference between the snapshot version and the current system is shown. Activate the check box to select the file for restore. Do so for all files you want to restore.

5. Click *Restore Selected* and confirm the action by clicking *Yes*.

PROCEDURE 3.4: RESTORING FILES USING THE `snapper` COMMAND

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

```
tux > sudo snapper -c CONFIG list -t single | grep timeline
```

   CONFIG needs to be replaced by an existing Snapper configuration. Use **snapper list-configs** to display a list.

2. Get a list of changed files for a given snapshot by running the following command:

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

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

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

```
tux > sudo snapper -c CONFIG diff SNAPSHOT_ID..0 FILE NAME
```

   If you do not specify *<FILE NAME>*, the difference for all files are shown.

4. To restore one or more files, run

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

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

3.3 System Rollback by Booting from Snapshots

The GRUB 2 version included on openSUSE Leap can boot from Btrfs snapshots. Together with Snapper's rollback feature, this allows to recover a misconfigured system. Only snapshots created for the default Snapper configuration (`root`) are bootable.

Important: Supported Configuration

As of openSUSE Leap 15.1 system rollbacks are only supported if the default subvolume configuration of the root partition has not been changed.

When booting a snapshot, the parts of the file system included in the snapshot are mounted read-only; all other file systems and parts that are excluded from snapshots are mounted read-write and can be modified.

Important: Undoing Changes Compared to Rollback

When working with snapshots to restore data, it is important to know that there are two fundamentally different scenarios Snapper can handle:

Undoing Changes

When undoing changes as described in *Section 3.2, "Using Snapper to Undo Changes"*, two snapshots are compared and the changes between these two snapshots are reverted. Using this method also allows to explicitly exclude selected files from being restored.

Rollback

When doing rollbacks as described in the following, the system is reset to the state at which the snapshot was taken.

To do a rollback from a bootable snapshot, the following requirements must be met. When doing a default installation, the system is set up accordingly.

REQUIREMENTS FOR A ROLLBACK FROM A BOOTABLE SNAPSHOT

- The root file system needs to be Btrfs. Booting from LVM volume snapshots is not supported.

- The root file system needs to be on a single device, a single partition and a single subvolume. Directories that are excluded from snapshots such as `/srv` (see *Section 3.1.2, "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 cannot write to any directory that is part of the snapshot. Data you write to other directories will *not* get lost, regardless of what you do next.
3. Depending on whether you want to perform the rollback or not, choose your next step:
   a. If the system is in a state where you do not want to do a rollback, reboot to boot into the current system state. You can then choose a different snapshot, or start the rescue system.
   b. To perform the rollback, run

      ```
      tux > sudo snapper rollback
      ```

      and reboot afterward. On the boot screen, choose the default boot entry to reboot into the reinstated system. A snapshot of the file system status before the rollback is created. The default subvolume for root will be replaced with a fresh read-write snapshot. For details, see *Section 3.3.1, "Snapshots after Rollback"*.
      It is useful to add a description for the snapshot with the `-d` option. For example:

      ```
      New file system root since rollback on DATE TIME
      ```

Tip: Rolling Back to a Specific Installation State

If snapshots are not disabled during installation, an initial bootable snapshot is created at the end of the initial system installation. You can go back to that state at any time by booting this snapshot. The snapshot can be identified by the description `after installation`.

A bootable snapshot is also created when starting a system upgrade to a service pack or a new major release (provided snapshots are not disabled).

3.3.1 Snapshots after Rollback

Before a rollback is performed, a snapshot of the running file system is created. The description references the ID of the snapshot that was restored in the rollback.

Snapshots created by rollbacks receive the value `number` for the `Cleanup` attribute. The rollback snapshots are therefore automatically deleted when the set number of snapshots is reached. Refer to *Section 3.6, "Automatic Snapshot Clean-Up"* for details. If the snapshot contains important data, extract the data from the snapshot before it is removed.

3.3.1.1 Example of Rollback Snapshot

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

```
root # snapper --iso list
Type   | # |     | Cleanup | Description           | Userdata
-------+---+ ... +---------+-----------------------+--------------
single | 0 |     |         | current               |
single | 1 |     |         | first root filesystem |
single | 2 |     | number  | after installation    | important=yes
```

After running **`sudo snapper rollback`** snapshot `3` is created and contains the state of the system before the rollback was executed. Snapshot `4` is the new default Btrfs subvolume and thus the system after a reboot.

```
root # snapper --iso list
Type   | # |     | Cleanup | Description           | Userdata
-------+---+ ... +---------+-----------------------+--------------
single | 0 |     |         | current               |
single | 1 |     | number  | first root filesystem |
single | 2 |     | number  | after installation    | important=yes
single | 3 |     | number  | rollback backup of #1 | important=yes
single | 4 |     |         |                       |
```

3.3.2 Accessing and Identifying Snapshot Boot Entries

To boot from a snapshot, reboot your machine and choose *Start Bootloader from a read-only snapshot*. A screen listing all bootable snapshots opens. The most recent snapshot is listed first, the oldest last. Use the keys ↓ and ↑ to navigate and press Enter to activate the selected snapshot. Activating a snapshot from the boot menu does not reboot the machine immediately, but rather opens the boot loader of the selected snapshot.

FIGURE 3.1: BOOT LOADER: SNAPSHOTS

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

```
[*]❶OS❷ (KERNEL❸,DATE❹TTIME❺,DESCRIPTION❻)
```

❶ If the snapshot was marked `important`, the entry is marked with a `*`.

❷ Operating system label.

❹ Date in the format `YYYY-MM-DD`.

❺ Time in the format `HH:MM`.

6 This field contains a description of the snapshot. In case of a manually created snapshot this is the string created with the option `--description` or a custom string (see *Tip: Setting a Custom Description for Boot Loader Snapshot Entries*). In case of an automatically created snapshot, it is the tool that was called, for example `zypp(zypper)` or `yast_sw_single`. Long descriptions may be truncated, depending on the size of the boot screen.

Tip: Setting a Custom Description for Boot Loader Snapshot Entries

It is possible to replace the default string in the description field of a snapshot with a custom string. This is for example useful if an automatically created description is not sufficient, or a user-provided description is too long. To set a custom string *`STRING`* for snapshot *`NUMBER`*, use the following command:

```
tux > sudo snapper modify --userdata "bootloader=STRING" NUMBER
```

The description should be no longer than 25 characters—everything that exceeds this size will not be readable on the boot screen.

3.3.3 Limitations

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

3.3.3.1 Directories Excluded from Snapshots

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

3.3.3.2 No Rollback of Boot Loader Data

A rollback of the boot loader is not possible, since all "stages" of the boot loader must fit together. This cannot be guaranteed when doing rollbacks of `/boot`.

3.4 Creating and Modifying Snapper Configurations

The way Snapper behaves is defined in a configuration file that is specific for each partition or `Btrfs` subvolume. These configuration files reside under `/etc/snapper/configs/`.

In case the root file system is big enough (approximately 12 GB), snapshots are automatically enabled for the root file system `/` upon installation. The corresponding default configuration is named `root`. It creates and manages the YaST and Zypper snapshot. See *Section 3.4.1.1, "Configuration Data"* for a list of the default values.

Note: Minimum Root File System Size for Enabling Snapshots

As explained in *Section 3.1, "Default Setup"*, enabling snapshots requires additional free space in the root file system. The amount depends on the amount of packages installed and the amount of changes made to the volume that is included in snapshots. The snapshot frequency and the number of snapshots that get archived also matter.

There is a minimum root file system size that is required to automatically enable snapshots during the installation. Currently this size is approximately 12 GB. This value may change in the future, depending on architecture and the size of the base system. It depends on the values for the following tags in the file `/control.xml` from the installation media:

```
<root_base_size>
<btrfs_increase_percentage>
```

It is calculated with the following formula: *ROOT_BASE_SIZE* * (1 + *BTRFS_INCREASE_PERCENTAGE*/100)

Keep in mind that this value is a minimum size. Consider using more space for the root file system. As a rule of thumb, double the size you would use when not having enabled snapshots.

You may create your own configurations for other partitions formatted with `Btrfs` or existing subvolumes on a `Btrfs` partition. In the following example we will set up a Snapper configuration for backing up the Web server data residing on a separate, `Btrfs`-formatted partition mounted at `/srv/www`.

After a configuration has been created, you can either use **`snapper`** itself or the YaST *Snapper* module to restore files from these snapshots. In YaST you need to select your *Current Configuration*, while you need to specify your configuration for **`snapper`** with the global switch `-c` (for example, **`snapper -c myconfig list`**).

To create a new Snapper configuration, run **`snapper create-config`**:

```
tux > sudo snapper -c www-data❶ create-config /srv/www❷
```

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:

```
tux > sudo 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:

```
tux > sudo snapper list-configs
Config | Subvolume
-------+----------
root   | /
usr    | /usr
local  | /local
```

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.

To display the default configuration run

```
tux > sudo snapper -c root get-config
```

Modify a Configuration

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"*.

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**.

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. To change a value, run **snapper -c** *CONFIG* **set-config "***KEY=VALUE***"**.

`ALLOW_GROUPS`, `ALLOW_USERS`

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

Defines the clean-up algorithm for snapshots pairs with identical pre and post snapshots. See *Section 3.6.3, "Cleaning Up Snapshot Pairs That Do Not Differ"* for details.

`FSTYPE`

File system type of the partition. Do not change.

The default value is `"btrfs"`.

NUMBER_*

Defines the clean-up algorithm for installation and admin snapshots. See *Section 3.6.1, "Cleaning Up Numbered Snapshots"* for details.

`QGROUP` / `SPACE_LIMIT`

Adds quota support to the clean-up algorithms. See *Section 3.6.5, "Adding Disk Quota Support"* for details.

`SUBVOLUME`

Mount point of the partition or subvolume to snapshot. Do not change.
The default value is `"/"`.

`SYNC_ACL`

If Snapper is used by regular users (see *Section 3.4.1.2, "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_CREATE`

If set to `yes`, hourly snapshots are created. Valid values: `yes`, `no`.
The default value is `"no"`.

`TIMELINE_CLEANUP` / `TIMELINE_LIMIT_*`

Defines the clean-up algorithm for timeline snapshots. See *Section 3.6.2, "Cleaning Up Timeline Snapshots"* for details.

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:

```
tux > sudo snapper --config web_data create /srv/www
```

2. The configuration file is created under `/etc/snapper/configs/CONFIG`, where CONFIG is the value you specified with `-c/--config` in the previous step (for example `/etc/snapper/configs/web_data`). Adjust it according to your needs; 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:

```
tux > sudo snapper -c web_data set-config "ALLOW_USERS=www_admin" SYNC_ACL="yes"
```

4. The given Snapper configuration can now be used by the specified user(s) and/or group(s). You can test it with the `list` command, for example:

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

3.5 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. 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. Use **`snapper list`** to show existing snapshots and their metadata:

`snapper --config home list`

Lists snapshots for the configuration `home`. To list snapshots for the default configuration (root), use **`snapper -c root list`** or **`snapper list`**.

`snapper list -a`
: Lists snapshots for all existing configurations.

`snapper list -t pre-post`
: Lists all pre and post snapshot pairs for the default (`root`) configuration.

`snapper list -t single`
: Lists all snapshots of the type `single` for the default (`root`) configuration.

The following metadata is available for each snapshot:

- **Type**: Snapshot type, see *Section 3.5.1.1, "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 3.6, "Automatic Snapshot Clean-Up"* 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 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 several 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 user data of a snapshot. All other metadata cannot be changed. The following examples explain how to modify snapshots from the command line. It should be easy to adopt them when using the YaST interface.

To modify a snapshot on the command line, you need to know its number. Use **`snapper list`** to display all snapshots and their numbers.

The YaST *Snapper* module already lists all snapshots. Choose one from the list and click *Modify*.

`snapper modify --cleanup-algorithm "timeline"` 10

Modifies the metadata of snapshot 10 for the default (`root`) configuration. The cleanup algorithm is set to `timeline`.

`snapper --config home modify --description "daily backup" -cleanup-algorithm "timeline" 120`

Modifies the metadata of snapshot 120 for a custom configuration named `home`. A new description is set and the cleanup algorithm is unset.

3.5.4 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`*.

Deleting the current default subvolume snapshot is not allowed.

When deleting snapshots with Snapper, the freed space will be claimed by a Btrfs process running in the background. Thus the visibility and the availability of free space is delayed. In case you need space freed by deleting a snapshot to be available immediately, use the option `--sync` with the delete command.

Tip: Deleting Snapshot Pairs

When deleting a `pre` snapshot, you should always delete its corresponding `post` snapshot (and vice versa).

`snapper delete 65`

Deletes snapshot 65 for the default (`root`) configuration.

`snapper -c home delete 89 90`

Deletes snapshots 89 and 90 for a custom configuration named `home`.

`snapper delete --sync 23`

Deletes snapshot 23 for the default (`root`) configuration and makes the freed space available immediately.

Tip: Delete Unreferenced Snapshots

Sometimes the Btrfs snapshot is present but the XML file containing the metadata for Snapper is missing. In this case the snapshot is not visible for Snapper and needs to be deleted manually:

```
btrfs subvolume delete /.snapshots/SNAPSHOTNUMBER/snapshot
rm -rf /.snapshots/SNAPSHOTNUMBER
```

Tip: Old Snapshots Occupy More Disk Space

If you delete snapshots to free space on your hard disk, make sure to delete old snapshots first. The older a snapshot is, the more disk space it occupies.

Snapshots are also automatically deleted by a daily cron job. Refer to *Section 3.5.1.2, "Cleanup-algorithms"* for details.

3.6 Automatic Snapshot Clean-Up

Snapshots occupy disk space and over time the amount of disk space occupied by the snapshots may become large. To prevent disks from running out of space, Snapper offers algorithms to automatically delete old snapshots. These algorithms differentiate between timeline snapshots and numbered snapshots (administration plus installation snapshot pairs). You can specify the number of snapshots to keep for each type.

In addition to that, you can optionally specify a disk space quota, defining the maximum amount of disk space the snapshots may occupy. It is also possible to automatically delete pre and post snapshots pairs that do not differ.

A clean-up algorithm is always bound to a single Snapper configuration, so you need to configure algorithms for each configuration. To prevent certain snapshots from being automatically deleted, refer to *Q:*.

The default setup (`root`) is configured to do clean-up for numbered snapshots and empty pre and post snapshot pairs. Quota support is enabled—snapshots may not occupy more than 50% of the available disk space of the root partition. Timeline snapshots are disabled by default, therefore the timeline clean-up algorithm is also disabled.

3.6.1 Cleaning Up Numbered Snapshots

Cleaning up numbered snapshots—administration plus installation snapshot pairs—is controlled by the following parameters of a Snapper configuration.

`NUMBER_CLEANUP`

Enables or disables clean-up of installation and admin snapshot pairs. If enabled, snapshot pairs are deleted when the total snapshot count exceeds a number specified with `NUMBER_LIMIT` and/or `NUMBER_LIMIT_IMPORTANT` *and* an age specified with `NUMBER_MIN_AGE`. Valid values: `yes` (enable), `no` (disable).

The default value is `"yes"`.

Example command to change or set:

```
tux > sudo snapper -c CONFIG set-config "NUMBER_CLEANUP=no"
```

`NUMBER_LIMIT` / `NUMBER_LIMIT_IMPORTANT`

Defines how many regular and/or important installation and administration snapshot pairs to keep. Only the youngest snapshots will be kept. Ignored if `NUMBER_CLEANUP` is set to `"no"`.

The default value is `"2-10"` for `NUMBER_LIMIT` and `"4-10"` for `NUMBER_LIMIT_IMPORTANT`.

Example command to change or set:

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

Important: Ranged Compared to Constant Values

In case quota support is enabled (see *Section 3.6.5, "Adding Disk Quota Support"*) the limit needs to be specified as a minimum-maximum range, for example `2-10`. If quota support is disabled, a constant value, for example `10`, needs to be provided, otherwise cleaning-up will fail with an error.

`NUMBER_MIN_AGE`

Defines the minimum age in seconds a snapshot must have before it can automatically be deleted. Snapshots younger than the value specified here will not be deleted, regardless of how many exist.

The default value is `"1800"`.

Example command to change or set:

```
tux > sudo snapper -c CONFIG set-config "NUMBER_MIN_AGE=864000"
```

Note: Limit and Age

`NUMBER_LIMIT`, `NUMBER_LIMIT_IMPORTANT` and `NUMBER_MIN_AGE` are always evaluated. Snapshots are only deleted when *all* conditions are met.

If you always want to keep the number of snapshots defined with `NUMBER_LIMIT*` regardless of their age, set `NUMBER_MIN_AGE` to `0`.

The following example shows a configuration to keep the last 10 important and regular snapshots regardless of age:

```
NUMBER_CLEANUP=yes
NUMBER_LIMIT_IMPORTANT=10
NUMBER_LIMIT=10
NUMBER_MIN_AGE=0
```

On the other hand, if you do not want to keep snapshots beyond a certain age, set `NUMBER_LIMIT*` to `0` and provide the age with `NUMBER_MIN_AGE`.

The following example shows a configuration to only keep snapshots younger than ten days:

```
NUMBER_CLEANUP=yes
NUMBER_LIMIT_IMPORTANT=0
NUMBER_LIMIT=0
NUMBER_MIN_AGE=864000
```

3.6.2 Cleaning Up Timeline Snapshots

Cleaning up timeline snapshots is controlled by the following parameters of a Snapper configuration.

`TIMELINE_CLEANUP`

Enables or disables clean-up of timeline snapshots. If enabled, snapshots are deleted when the total snapshot count exceeds a number specified with `TIMELINE_LIMIT_*` *and* an age specified with `TIMELINE_MIN_AGE`. Valid values: `yes`, `no`.
The default value is `"yes"`.

Example command to change or set:

```
tux > sudo snapper -c CONFIG set-config "TIMELINE_CLEANUP=yes"
```

`TIMELINE_LIMIT_DAILY`, `TIMELINE_LIMIT_HOURLY`, `TIMELINE_LIMIT_MONTHLY`, `TIMELINE_LIMIT_WEEKLY`, `TIMELINE_LIMIT_YEARLY`

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

The default value for each entry is `"10"`, except for `TIMELINE_LIMIT_WEEKLY`, which is set to `"0"` by default.

`TIMELINE_MIN_AGE`

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

The default value is `"1800"`.

EXAMPLE 3.1: EXAMPLE TIMELINE CONFIGURATION

```
TIMELINE_CLEANUP="yes"
TIMELINE_CREATE="yes"
TIMELINE_LIMIT_DAILY="7"
TIMELINE_LIMIT_HOURLY="24"
TIMELINE_LIMIT_MONTHLY="12"
TIMELINE_LIMIT_WEEKLY="4"
TIMELINE_LIMIT_YEARLY="2"
TIMELINE_MIN_AGE="1800"
```

This example configuration enables hourly snapshots which are automatically cleaned up. `TIMELINE_MIN_AGE` and `TIMELINE_LIMIT_*` are always both evaluated. In this example, the minimum age of a snapshot before it can be deleted is set to 30 minutes (1800 seconds). Since we create hourly snapshots, this ensures that only the latest snapshots are kept. If `TIMELINE_LIMIT_DAILY` is set to not zero, this means that the first snapshot of the day is kept, too.

SNAPSHOTS TO BE KEPT

- Hourly: The last 24 snapshots that have been made.
- Daily: The first daily snapshot that has been made is kept from the last seven days.
- Monthly: The first snapshot made on the last day of the month is kept for the last twelve months.

- Weekly: The first snapshot made on the last day of the week is kept from the last four weeks.
- Yearly: The first snapshot made on the last day of the year is kept for the last two years.

3.6.3 Cleaning Up Snapshot Pairs That Do Not Differ

As explained in *Section 3.1.1, "Types of Snapshots"*, whenever you run a YaST module or execute Zypper, a pre snapshot is created on start-up and a post snapshot is created when exiting. In case you have not made any changes there will be no difference between the pre and post snapshots. Such "empty" snapshot pairs can be automatically be deleted by setting the following parameters in a Snapper configuration:

`EMPTY_PRE_POST_CLEANUP`

If set to `yes`, pre and post snapshot pairs that do not differ will be deleted.
The default value is `"yes"`.

`EMPTY_PRE_POST_MIN_AGE`

Defines the minimum age in seconds a pre and post snapshot pair that does not differ must have before it can automatically be deleted.
The default value is `"1800"`.

3.6.4 Cleaning Up Manually Created Snapshots

Snapper does not offer custom clean-up algorithms for manually created snapshots. However, you can assign the number or timeline clean-up algorithm to a manually created snapshot. If you do so, the snapshot will join the "clean-up queue" for the algorithm you specified. You can specify a clean-up algorithm when creating a snapshot, or by modifying an existing snapshot:

`snapper create --description "Test" --cleanup-algorithm number`

Creates a stand-alone snapshot (type single) for the default (root) configuration and assigns the `number` clean-up algorithm.

`snapper modify --cleanup-algorithm "timeline" 25`

Modifies the snapshot with the number 25 and assigns the clean-up algorithm `timeline`.

3.6.5 Adding Disk Quota Support

In addition to the number and/or timeline clean-up algorithms described above, Snapper supports quotas. You can define what percentage of the available space snapshots are allowed to occupy. This percentage value always applies to the Btrfs subvolume defined in the respective Snapper configuration.

If Snapper was enabled during the installation, quota support is automatically enabled. In case you manually enable Snapper at a later point in time, you can enable quota support by running **`snapper setup-quota`**. This requires a valid configuration (see *Section 3.4, "Creating and Modifying Snapper Configurations"* for more information).

Quota support is controlled by the following parameters of a Snapper configuration.

`QGROUP`

The Btrfs quota group used by Snapper. If not set, run **`snapper setup-quota`**. If already set, only change if you are familiar with **`man 8 btrfs-qgroup`**. This value is set with **`snapper setup-quota`** and should not be changed.

`SPACE_LIMIT`

Limit of space snapshots are allowed to use in fractions of 1 (100%). Valid values range from 0 to 1 (0.1 = 10%, 0.2 = 20%, ...).

The following limitations and guidelines apply:

- Quotas are only activated in *addition* to an existing number and/or timeline clean-up algorithm. If no clean-up algorithm is active, quota restrictions are not applied.
- With quota support enabled, Snapper will perform two clean-up runs if required. The first run will apply the rules specified for number and timeline snapshots. Only if the quota is exceeded after this run, the quota-specific rules will be applied in a second run.
- Even if quota support is enabled, Snapper will always keep the number of snapshots specified with the `NUMBER_LIMIT*` and `TIMELINE_LIMIT*` values, even if the quota will be exceeded. It is therefore recommended to specify ranged values (`MIN-MAX`) for `NUMBER_LIMIT*` and `TIMELINE_LIMIT*` to ensure the quota can be applied.
  If, for example, `NUMBER_LIMIT=5-20` is set, Snapper will perform a first clean-up run and reduce the number of regular numbered snapshots to 20. In case these 20 snapshots exceed the quota, Snapper will delete the oldest ones in a second run until the quota is met. A minimum of five snapshots will always be kept, regardless of the amount of space they occupy.

3.7 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: Displaying the amount of disk space a snapshot allocates is currently not supported by the `Btrfs` tools. However, if you have quota enabled, it is possible to determine how much space would be freed if *all* snapshots would be deleted:

1. Get the quota group ID (`1/0` in the following example):

```
tux > sudo snapper -c root get-config | grep QGROUP
QGROUP                 | 1/0
```

2. Rescan the subvolume quotas:

```
tux > sudo btrfs quota rescan -w /
```

3. Show the data of the quota group (`1/0` in the following example):

```
tux > sudo btrfs qgroup show / | grep "1/0"
1/0           4.80GiB    108.82MiB
```

   The third column shows the amount of space that would be freed when deleting all snapshots (`108.82MiB`).

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: *How to make a snapshot permanent?*

A: Currently Snapper does not offer means to prevent a snapshot from being deleted manually. However, you can prevent snapshots from being automatically deleted by clean-up algorithms. Manually created snapshots (see *Section 3.5.2, "Creating Snapshots"*) have no clean-up algorithm assigned unless you specify one with `--cleanup-algorithm`. Automatically created snapshots always either have the `number` or `timeline` algorithm assigned. To remove such an assignment from one or more snapshots, proceed as follows:

1. List all available snapshots:

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

2. Memorize the number of the snapshot(s) you want to prevent from being deleted.

3. Run the following command and replace the number placeholders with the number(s) you memorized:

```
tux > sudo snapper modify --cleanup-algorithm "" #1 #2 #n
```

4. Check the result by running **`snapper list -a`** again. The entry in the column `Cleanup` should now be empty for the snapshots you modified.

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

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

4 Remote 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.

Important: Supported Display Managers

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

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

4.1 The `vncviewer` Client

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

4.1.1 Connecting Using the vncviewer CLI

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

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

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

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

Note: Display and Port Number

The actual display or port number you specify in the VNC client must be the same as the display or port number picked by the **vncserver** command on the target machine. See *Section 4.4, "Persistent VNC Sessions"* for further info.

4.1.2 Connecting Using the vncviewer GUI

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

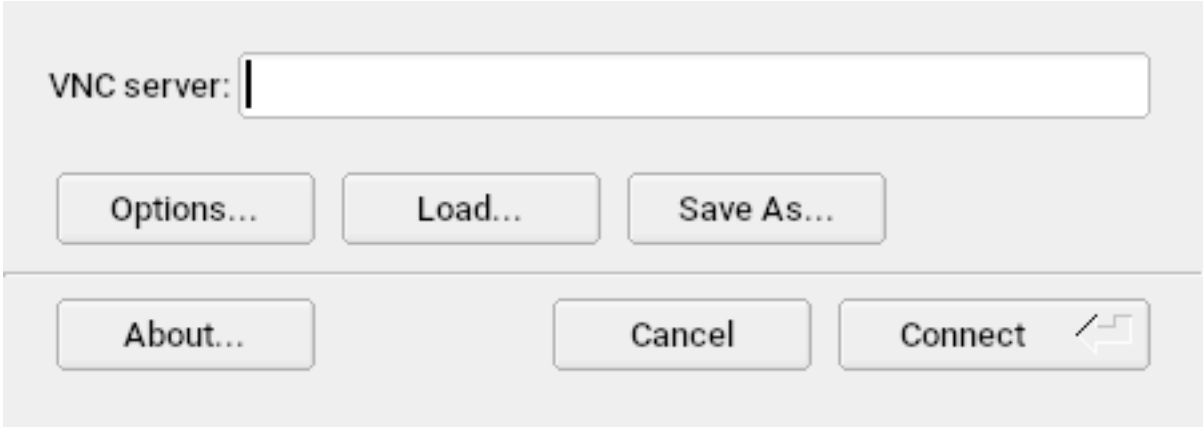


FIGURE 4.1: VNCVIEWER

4.1.3 Notification of Unencrypted Connections

The VNC protocol supports different kinds of encrypted connections, not to be confused with password authentication. If a connection does not use TLS, the text "(Connection not encrypted!)" can be seen in the window title of the VNC viewer.

4.2 Remmina: the Remote Desktop Client

Remmina is a modern and feature rich remote desktop client. It supports several access methods, for example VNC, SSH, RDP, or Spice.

4.2.1 Installation

To use Remmina, verify whether the `remmina` package is installed on your system, and install it if not. Remember to install the VNC plug-in for Remmina as well:

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

4.2.2 Main Window

Run Remmina by entering the **`remmina`** command.

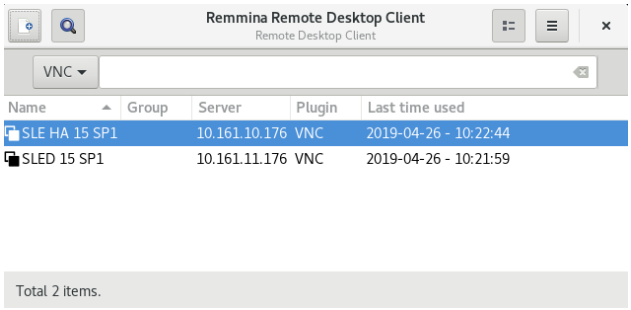


FIGURE 4.2: REMMINA'S MAIN WINDOW

The main application window shows the list of stored remote sessions. Here you can add and save a new remote session, quick-start a new session without saving it, start a previously saved session, or set Remmina's global preferences.

4.2.3 Adding Remote Sessions

To add and save a new remote session, click  in the top left of the main window. The *Remote Desktop Preference* window opens.

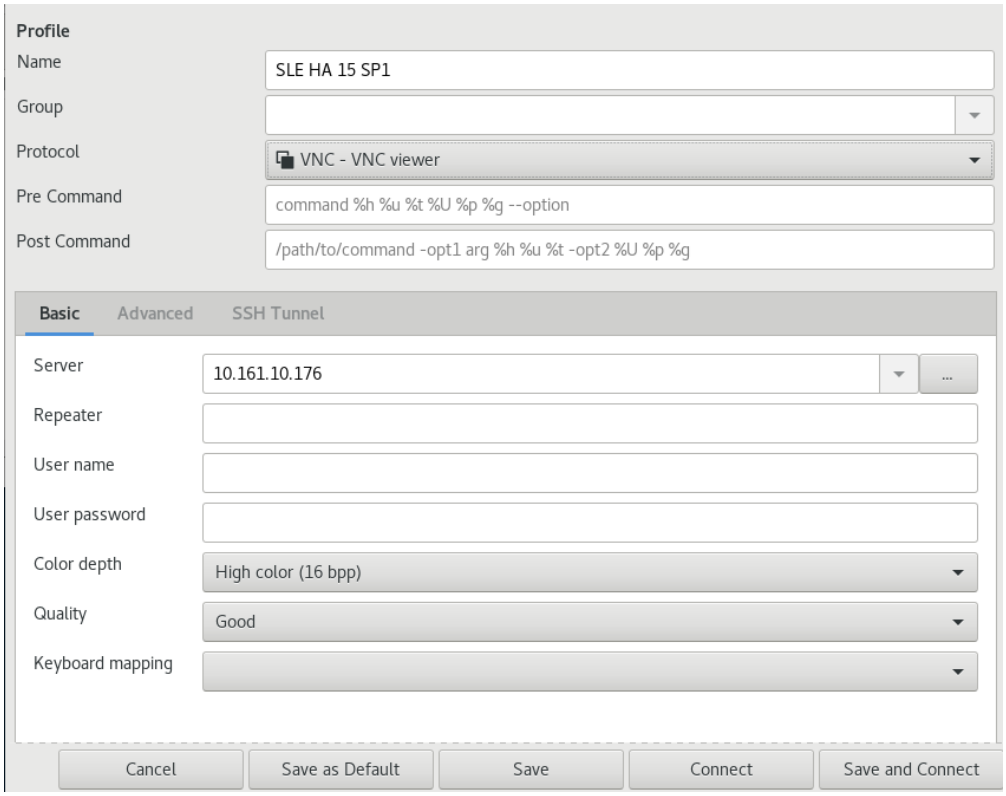


FIGURE 4.3: REMOTE DESKTOP PREFERENCE

Complete the fields that specify your newly added remote session profile. The most important are:

Name

Name of the profile. It will be listed in the main window.

Protocol

The protocol to use when connecting to the remote session, for example VNC.

Server

The IP or DNS address and display number of the remote server.

User name, Password

Credentials to use for remote authentication. Leave empty for no authentication.

Color depth, Quality

Select the best options according to your connection speed and quality.

Select the *Advanced* tab to enter more specific settings.

Tip: Disable Encryption

If the communication between the client and the remote server is not encrypted, activate *Disable encryption*, otherwise the connection fails.

Select the *SSH* tab for advanced SSH tunneling and authentication options.

Confirm with *Save*. Your new profile will be listed in the main window.

4.2.4 Starting Remote Sessions

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

4.2.4.1 Quick-starting Remote Sessions

To start a remote session quickly without adding and saving connection details, use the drop-down box and text box at the top of the main window.

FIGURE 4.4: QUICK-STARTING

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

4.2.4.2 Opening Saved Remote Sessions

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

4.2.4.3 Remote Sessions Window

Remote sessions are opened in tabs of a separate window. Each tab hosts one session. The toolbar on the left of the window helps you manage the windows/sessions, such as toggle fullscreen mode, resize the window to match the display size of the session, send specific keystrokes to the session, take screenshots of the session, or set the image quality.

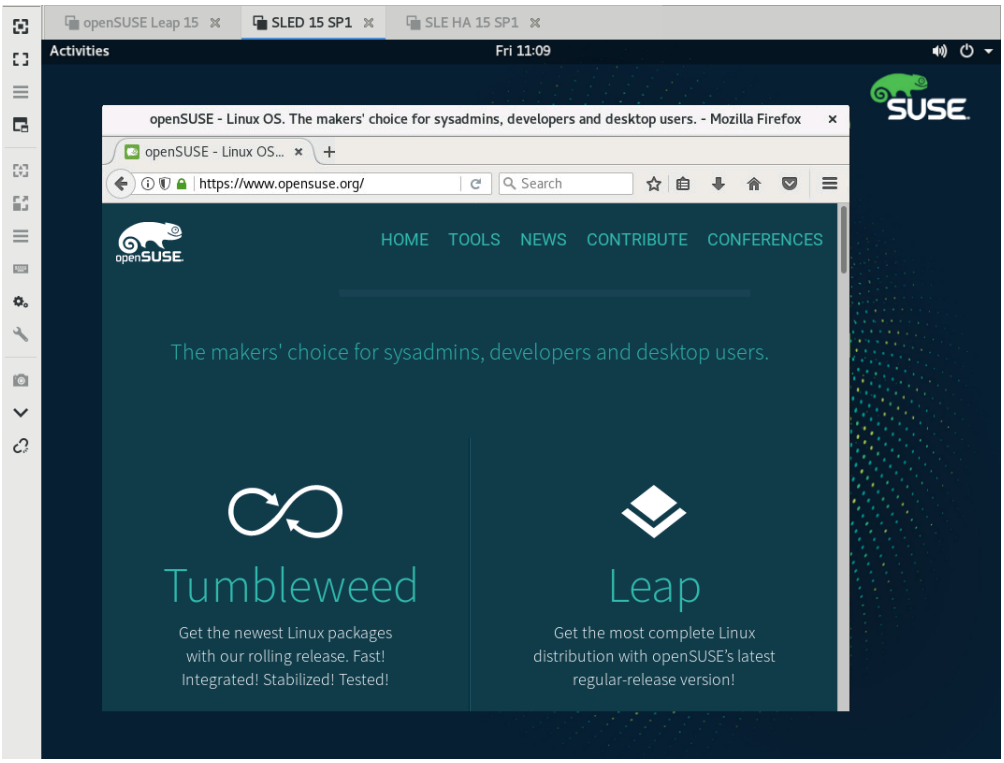


FIGURE 4.5: REMMINA VIEWING REMOTE SESSION

4.2.5 Editing, Copying, and Deleting Saved Sessions

To *edit* a saved remote session, right-click its name in Remmina's main window and select *Edit*. Refer to *Section 4.2.3, "Adding Remote Sessions"* for the description of the relevant fields.

To *copy* a saved remote session, right-click its name in Remmina's main window and select *Copy*. In the *Remote Desktop Preference* window, change the name of the profile, optionally adjust relevant options, and confirm with *Save*.

To *Delete* a saved remote session, right-click its name in Remmina's main window and select *Delete*. Confirm with *Yes* in the next dialog.

4.2.6 Running Remote Sessions from the Command Line

If you need to open a remote session from the command line or from a batch file without first opening the main application window, use the following syntax:

```
tux > remmina -c profile_name.remmina
```

Remmina's profile files are stored in the `.local/share/remmina/` directory in your home directory. To determine which profile file belongs to the session you want to open, run Remmina, click the session name in the main window, and read the path to the profile file in the window's status line at the bottom.

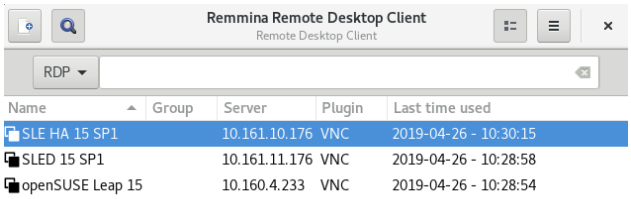


FIGURE 4.6: READING PATH TO THE PROFILE FILE

While Remmina is not running, you can rename the profile file to a more reasonable file name, such as `sle15.remmina`. You can even copy the profile file to your custom directory and run it using the **`remmina -c`** command from there.

4.3 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. When you terminate the client connection to such a VNC session, all applications started within that session will be terminated, too. One-time VNC sessions cannot be shared, but it is possible to have multiple sessions on a single host at the same time.

PROCEDURE 4.1: ENABLING ONE-TIME VNC SESSIONS

1. Start *YaST* › *Network Services* › *Remote Administration (VNC)*.
2. Check *Allow Remote Administration Without Session Management*.
3. Activate *Enable access using a web browser* if you plan to access the VNC session in a Web browser window.

4. If necessary, also check *Open Port in Firewall* (for example, when your network interface is configured to be in the External Zone). If you have more than one network interface, restrict opening the firewall ports to a specific interface via *Firewall Details*.

5. Confirm your settings with *Next*.

6. In case not all needed packages are available yet, you need to approve the installation of missing packages.

Tip: Restart the Display Manager

YaST makes changes to the display manager settings. You need to log out of your current graphical session and restart the display manager for the changes to take effect.

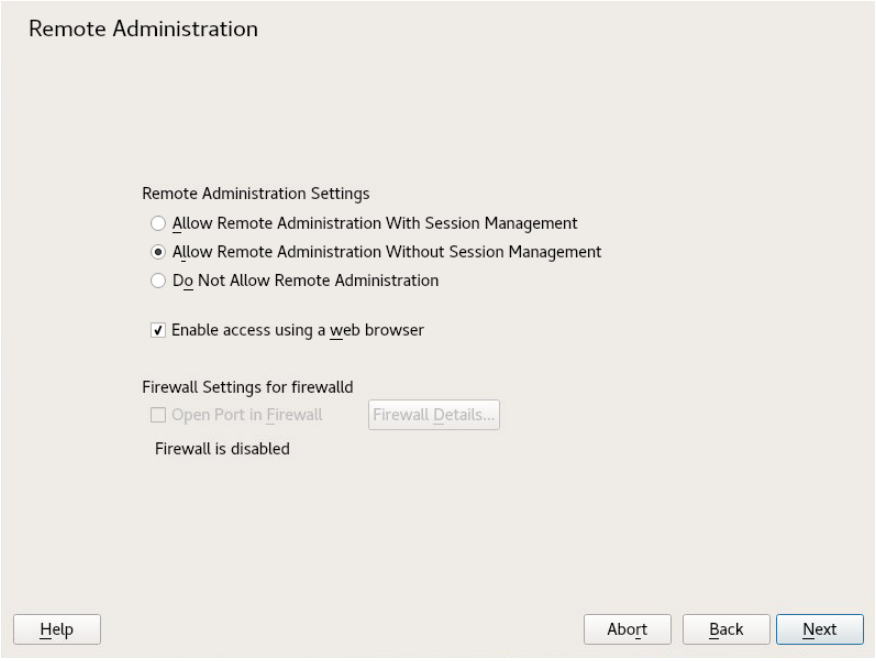


FIGURE 4.7: REMOTE ADMINISTRATION

4.3.1 Available Configurations

The default configuration on openSUSE Leap serves sessions with a resolution of 1024x768 pixels at a color depth of 16-bit. The sessions are available on ports `5901` for “regular” VNC viewers (equivalent to VNC display `1`) and on port `5801` for Web browsers.

Other configurations can be made available on different ports, see *Section 4.3.3, "Configuring One-time VNC Sessions"*.

VNC display numbers and X display numbers are independent in one-time sessions. A VNC display number is manually assigned to every configuration that the server supports (:1 in the example above). Whenever a VNC session is initiated with one of the configurations, it automatically gets a free X display number.

By default, both the VNC client and server try to communicate securely via a self-signed SSL certificate, which is generated after installation. You can either use the default one, or replace it with your own. When using the self-signed certificate, you need to confirm its signature before the first connection—both in the VNC viewer and the Web browser.

4.3.2 Initiating a One-time VNC Session

To connect to a one-time VNC session, a VNC viewer must be installed, see also *Section 4.1, "The* **`vncviewer`** *Client"*. Alternatively use a JavaScript-capable Web browser to view the VNC session by entering the following URL: `http://jupiter.example.com:5801`

4.3.3 Configuring One-time VNC Sessions

You can skip this section, if you do not need or want to modify the default configuration.

One-time VNC sessions are started via the `systemd` socket `xvnc.socket`. By default it offers six configuration blocks: three for VNC viewers (`vnc1` to `vnc3`), and three serving a JavaScript client (`vnchttpd1` to `vnchttpd3`). By default only `vnc1` and `vnchttpd1` are active.

To activate the VNC server socket at boot time, run the following command:

```
sudo systemctl enable xvnc.socket
```

To start the socket immediately, run:

```
sudo systemctl start xvnc.socket
```

The **`Xvnc`** server can be configured via the `server_args` option. For a list of options, see **`Xvnc --help`**.

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

Activate configuration changes by entering the following command:

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

Important: Firewall and VNC Ports

When activating Remote Administration as described in *Procedure 4.1, "Enabling One-time VNC Sessions"*, the ports `5801` and `5901` are opened in the firewall. If the network interface serving the VNC sessions is protected by a firewall, you need to manually open the respective ports when activating additional ports for VNC sessions. See *Book "Security Guide", Chapter 16 "Masquerading and Firewalls"* for instructions.

4.4 Persistent VNC Sessions

A persistent session can be accessed from multiple clients simultaneously. This is ideal for demonstration purposes where one client has full access and all other clients have view-only access. Another use case are trainings where the trainer might need access to the trainee's desktop.

Tip: Connecting to a Persistent VNC Session

To connect to a persistent VNC session, a VNC viewer must be installed. Refer to *Section 4.1, "The* **`vncviewer`** *Client"* for more details. Alternatively use a JavaScript-capable Web browser to view the VNC session by entering the following URL: `http://jupiter.example.com:5801`

There are two types of persistent VNC sessions:

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

4.4.1 VNC Session Initiated Using `vncserver`

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

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

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

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

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

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

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

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

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

Important: Security Considerations

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

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

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

4.4.1.1 Configuring Persistent VNC Sessions

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

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

Note: One Configuration for Each User

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

4.4.2 VNC Session Initiated Using `vncmanager`

PROCEDURE 4.3: ENABLING PERSISTENT VNC SESSIONS

1. Start *YaST* › *Network Services* › *Remote Administration (VNC)*.
2. Activate *Allow Remote Administration With Session Management*.
3. Activate *Enable access using a web browser* if you plan to access the VNC session in a Web browser window.
4. If necessary, also check *Open Port in Firewall* (for example, when your network interface is configured to be in the External Zone). If you have more than one network interface, restrict opening the firewall ports to a specific interface via *Firewall Details*.
5. Confirm your settings with *Next*.

6. In case not all needed packages are available yet, you need to approve the installation of missing packages.

Tip: Restart the Display Manager

YaST makes changes to the display manager settings. You need to log out of your current graphical session and restart the display manager for the changes to take effect.

4.4.2.1 Configuring Persistent VNC Sessions

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

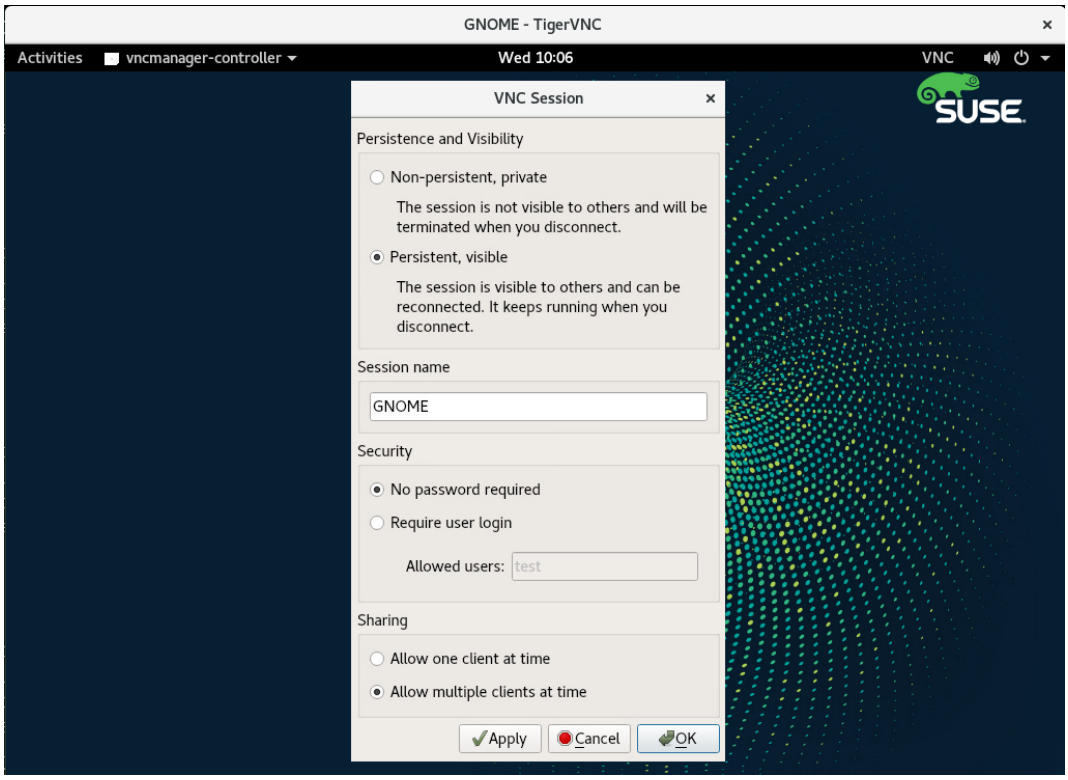


FIGURE 4.8: VNC SESSION SETTINGS

There are several settings that influence the VNC session's behavior:

Non-persistent, private

This is equivalent to a one-time session. It is not visible to others and will be terminated after you disconnect from it. Refer to *Section 4.3, "One-time VNC Sessions"* for more information.

Persistent, visible

The session is visible to other users and keeps running even after you disconnect from it.

Session name

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

No password required

The session will be freely accessible without having to log in under user credentials.

Require user login

You need to log in with a valid user name and password to access the session. Lists the valid user names in the *Allowed users* text box.

Allow one client at a time

Prevents multiple users from joining the session at the same time.

Allow multiple clients at a time

Allows multiple users to join the persistent session at the same time. Useful for remote presentations or trainings.

Confirm with *OK*.

4.4.2.2 Joining Persistent VNC Sessions

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

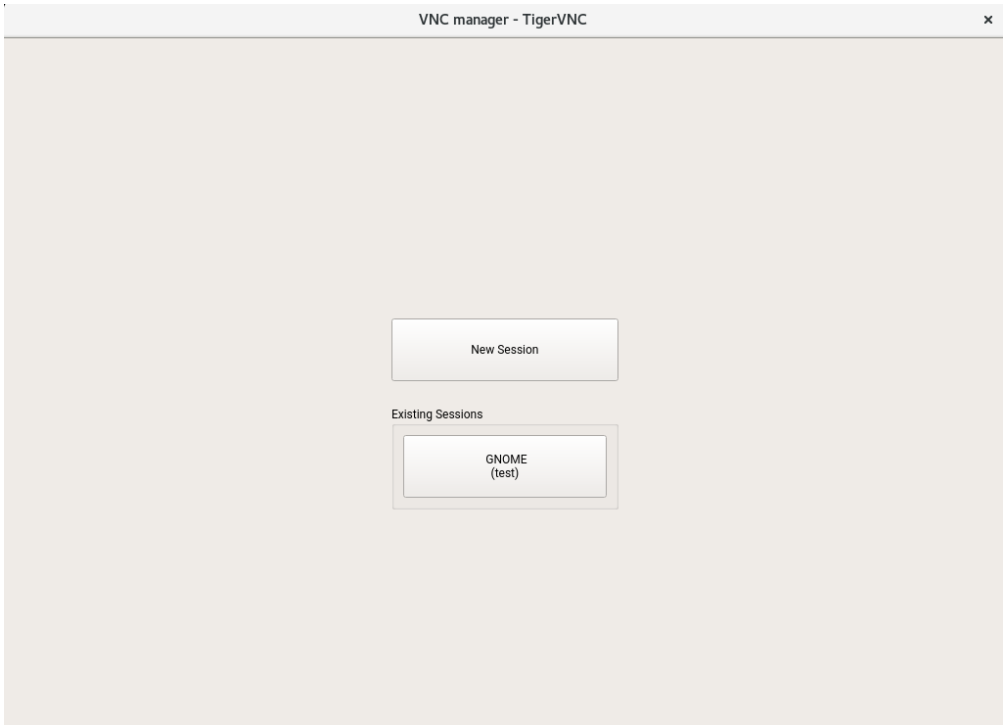


FIGURE 4.9: JOINING A PERSISTENT VNC SESSION

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

4.5 Encrypted VNC Communication

If the VNC server is set up properly, all communication between the VNC server and the client is encrypted. The authentication happens at the beginning of the session; the actual data transfer only begins afterward.

Whether for a one-time or a persistent VNC session, security options are configured via the `-securitytypes` parameter of the **`/usr/bin/Xvnc`** command located on the `server_args` line. The `-securitytypes` parameter selects both authentication method and encryption. It has the following options:

AUTHENTICATIONS

None, TLSNone, X509None

No authentication.

VncAuth, TLSVnc, X509Vnc

Authentication using custom password.

Plain, TLSPlain, X509Plain

Authentication using PAM to verify user's password.

ENCRYPTIONS

None, VncAuth, Plain

No encryption.

TLSNone, TLSVnc, TLSPlain

Anonymous TLS encryption. Everything is encrypted, but there is no verification of the remote host. So you are protected against passive attackers, but not against man-in-the-middle attackers.

X509None, X509Vnc, X509Plain

TLS encryption with certificate. If you use a self-signed certificate, you will be asked to verify it on the first connection. On subsequent connections you will be warned only if the certificate changed. So you are protected against everything except man-in-the-middle on the first connection (similar to typical SSH usage). If you use a certificate signed by a certificate authority matching the machine name, then you get full security (similar to typical HTTPS usage).

Tip: Path to Certificate and Key

With X509 based encryption, you need to specify the path to the X509 certificate and the key with `-X509Cert` and `-X509Key` options.

If you select multiple security types separated by comma, the first one supported and allowed by both client and server will be used. That way you can configure opportunistic encryption on the server. This is useful if you need to support VNC clients that do not support encryption.

On the client, you can also specify the allowed security types to prevent a downgrade attack if you are connecting to a server which you know has encryption enabled (although our vncviewer will warn you with the "Connection not encrypted!" message in that case).

5 Expert Partitioner

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

5.1 Using the Expert Partitioner

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

Warning: Repartitioning the Running System

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

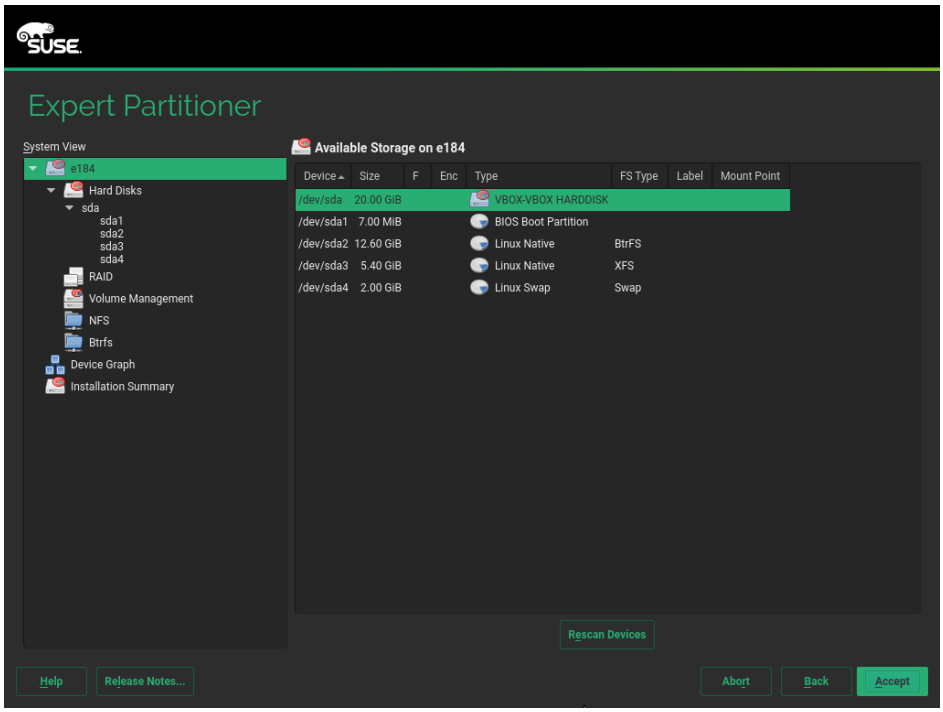


FIGURE 5.1: THE YAST PARTITIONER

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

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

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

5.1.1 Partition Tables

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

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

5.1.1.1 Master Boot Record

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

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

UEFI firmwares usually support booting from MBR in the legacy mode.

5.1.1.2 GPT Partition Table

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

Old BIOS firmwares do not support booting from GPT partitions.

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

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

Note: Partitions Created with Parted 3.1 or Earlier Mislabeled

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

To remove the flag, run:

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

5.1.2 Partitions

The YaST Partitioner can create and format partitions with several file systems. The default file system used by openSUSE Leap is `Btrfs`. For details, see *Section 5.1.2.2, "Btrfs Partitioning"*.

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

5.1.2.1 Creating a Partition

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

1. Click *Add* to create a new partition. When using *MBR*, specify to create a primary or extended partition. Within the extended partition, you can create several logical partitions. For details, see *Section 5.1.1, "Partition Tables"*.
2. Specify the size of the new partition. You can either choose to occupy all the free unpartitioned space, or enter a custom size.
3. Select the file system to use and a mount point. YaST suggests a mount point for each partition created. To use a different mount method, like mount by label, select *Fstab Options*. 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.

5.1.2.2 Btrfs Partitioning

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

Important: Btrfs on an Encrypted Root Partition

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

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

`/boot/grub2/i386-pc`, `/boot/grub2/x86_64-efi`, `/boot/grub2/powerpc-ieee1275`, `/boot/grub2/s390x-emu`
: A rollback of the boot loader configuration is not supported. The directories listed above are architecture-specific. The first two directories are present on AMD64/Intel 64 machines, the latter two on IBM POWER and on IBM Z, respectively.

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

`/opt`, `/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/cache`, `/var/crash`
: All directories containing temporary files and caches are excluded from snapshots.

`/usr/local`
: This directory is used when manually installing software. It is excluded to avoid uninstalling these installations on rollbacks.

`/var/lib/libvirt/images`
: The default location for virtual machine images managed with libvirt. Excluded to ensure virtual machine images are not replaced with older versions during a rollback. By default, this subvolume is created with the option `no copy on write`.

`/var/lib/mailman`, `/var/spool`
: Directories containing mails or mail queues are excluded to avoid a loss of mails after a rollback.

`/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/mariadb`, `/var/lib/mysql`, `/var/lib/pgqsl`
: These directories contain database data. By default, these subvolumes are created with the option `no copy on write`.

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

Tip: Size of Btrfs Partition

Since saved snapshots require more disk space, it is recommended to reserve enough space for Btrfs. The suggested size for a root Btrfs partition with default subvolumes is 20 GB.

5.1.2.3 Managing Btrfs Subvolumes using YaST

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

PROCEDURE 5.1: BTRFS SUBVOLUMES WITH YAST

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

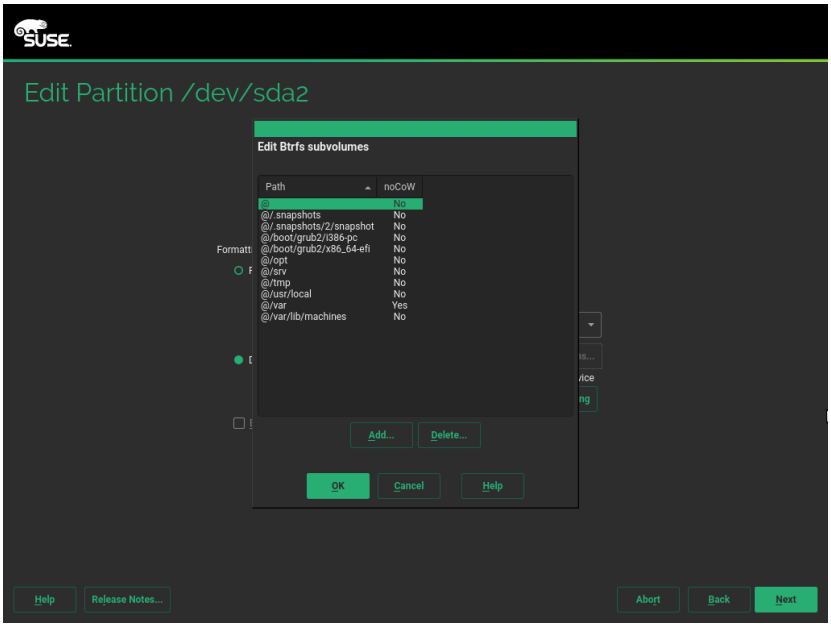


FIGURE 5.2: BTRFS SUBVOLUMES IN YAST PARTITIONER

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.

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

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

Warning: Changing the File System

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

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

Encrypt Device

If you activate the encryption, all data is written to the hard disk in encrypted form. This increases the security of sensitive data, but reduces the system speed, as the encryption takes some time to process. More information about the encryption of file systems is provided in *Book "Security 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 devices names, use the mount option *Device ID*, *UUID* or *LABEL*. In openSUSE Leap, persistent device names are enabled by default.

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

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

3. Select *Finish* to save the changes.

Note: Resize File Systems

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

5.1.4 Expert Options

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

Create New Partition Table

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

Warning: Creating a New Partition Table

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

Clone This Disk

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

5.1.5 Advanced Options

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

Configure iSCSI

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

Configure Multipath

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

5.1.6 More Partitioning Tips

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

5.1.6.1 Cylinder Numbers

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

5.1.6.2 Using swap

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

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

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

System with no swap

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

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

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

System with lots of swap (several GB)

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

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

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, to add a swap file with 128 MB swap at `/var/lib/swap/swapfile`, use the commands:

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

2. Initialize this swap file with the command

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

Note: Changed UUID for Swap Partitions When Formatting via `mkswap`

Do not reformat existing swap partitions with `mkswap` if possible. Reformatting with `mkswap` will change the UUID value of the swap partition. Either reformat via YaST (which will update `/etc/fstab`) or adjust `/etc/fstab` manually.

3. Activate the swap with the command

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

To disable this swap file, use the command

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

4. Check the current available swap spaces with the command

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

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

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

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

5.1.7 Partitioning and LVM

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

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

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

Warning: File System for Booting

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

5.2 LVM Configuration

This section explains specific steps to take when configuring LVM.

Warning: Back up Your Data

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

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

5.2.1 Create Physical Volume

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

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

5.2.2 Creating Volume Groups

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

1. Enter a name for the VG, for example, `system`.

2. Select the desired *Physical Extend Size*. This value defines the size of a physical block in the volume group. All the disk space in a volume group is handled in blocks of this size.

3. Add the prepared PVs to the VG by selecting the device and clicking *Add*. Selecting several devices is possible by holding `Ctrl` while selecting the devices.

4. Select *Finish* to make the VG available to further configuration steps.

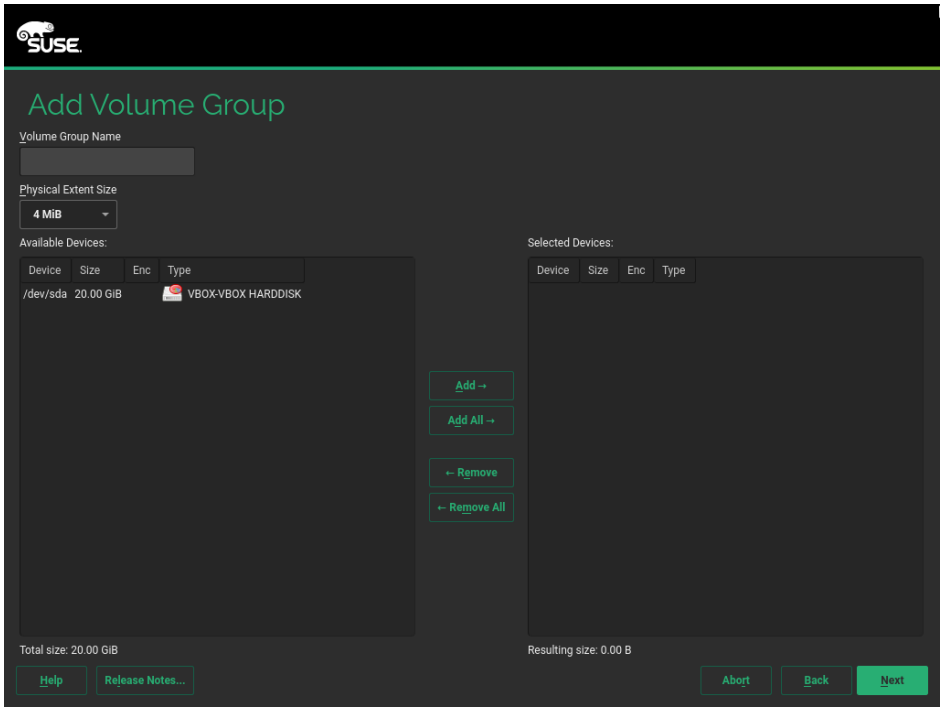


FIGURE 5.3: CREATING A VOLUME GROUP

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

5.2.3 Configuring Logical Volumes

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

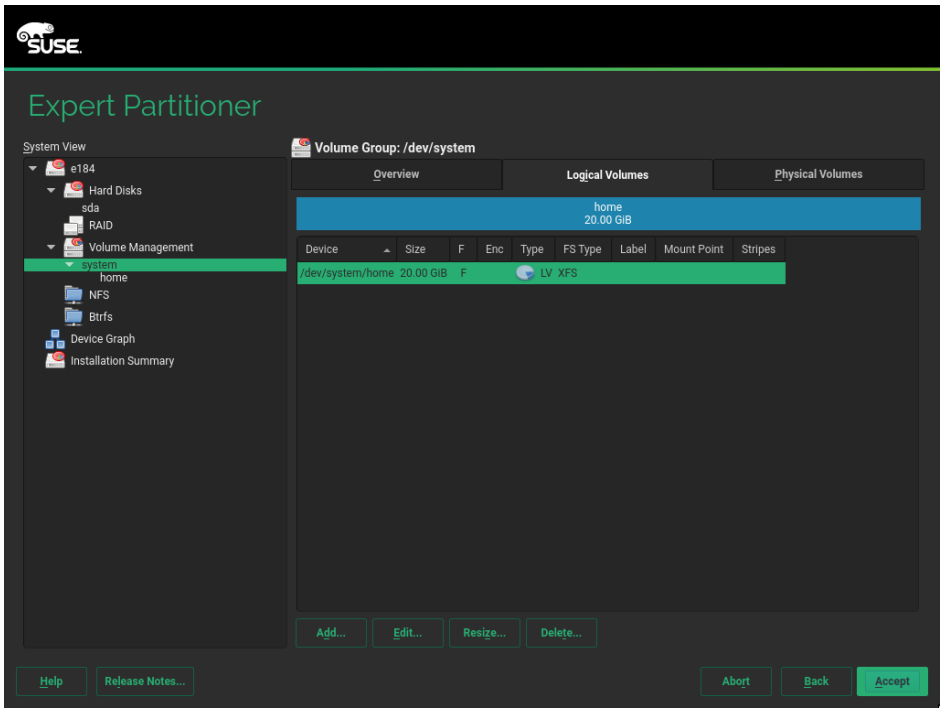


FIGURE 5.4: LOGICAL VOLUME MANAGEMENT

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

1. Enter the name of the LV. For a partition that should be mounted to `/home`, a name like `HOME` could be used.

2. Select the type of the LV. It can be either *Normal Volume*, *Thin Pool*, or *Thin Volume*. Note that you need to create a thin pool first, which can store individual thin volumes. The big advantage of thin provisioning is that the total sum of all thin volumes stored in a thin pool can exceed the size of the pool itself.

3. Select the size and the number of stripes of the LV. If you have only one PV, selecting more than one stripe is not useful.

4. Choose the file system to use on the LV and the mount point.

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

Warning: Striping

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

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

5.3 Soft RAID

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

5.3.1 Soft RAID Configuration

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

1. Select a hard disk from *Hard Disks*.
2. Change to the *Partitions* tab.
3. Click *Add* and enter the desired size of the raid partition on this disk.
4. Use *Do not Format the Partition* and change the *File System ID* to *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*.

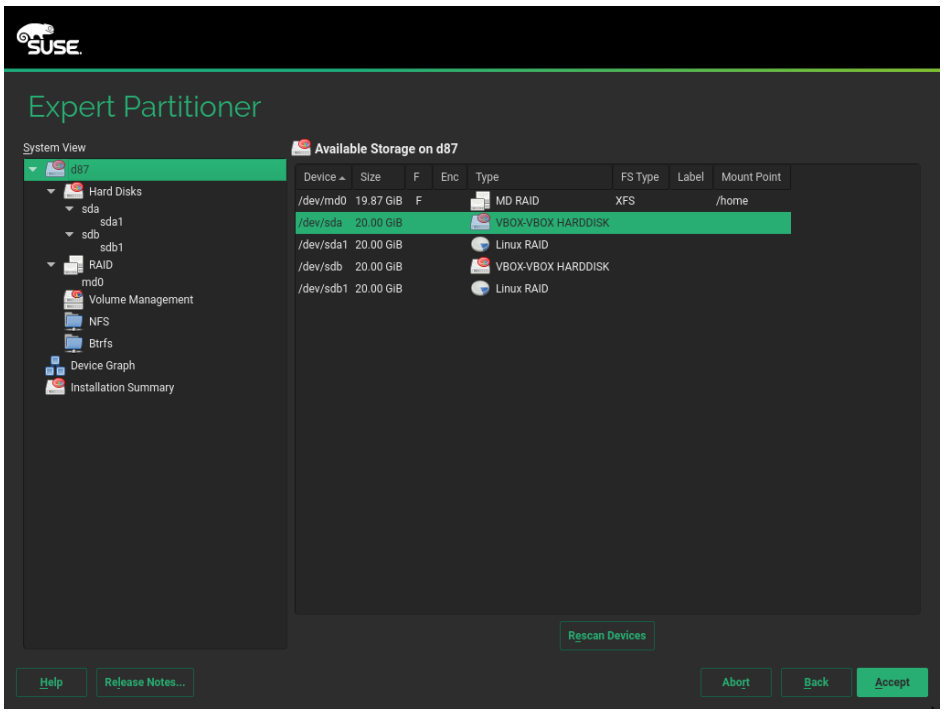


FIGURE 5.5: RAID PARTITIONS

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

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

5.3.2 Troubleshooting

Check the file `/proc/mdstat` to find out whether a RAID partition has been damaged. If Th 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 parameter.

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

Tip: Check Your Boot Loader Configuration Kernel

It is recommended to check your boot loader configuration after having installed another kernel to set the default boot entry of your choice. See *Section 12.3, "Configuring the Boot Loader with YaST"* for more information.

6.1 Enabling and Configuring Multiversion Support

Installing multiple versions of a software package (multiversion support) is enabled by default from openSUSE Leap. To verify this setting, proceed as follows:

1. Open `/etc/zypp/zypp.conf` with the editor of your choice as `root`.
2. Search for the string `multiversion`. If multiversion is enabled for all kernel packages capable of this feature, the following line appears uncommented:

   ```
   multiversion = provides:multiversion(kernel)
   ```

3. To restrict multiversion support to certain kernel flavors, add the package names as a comma-separated list to the `multiversion` option in `/etc/zypp/zypp.conf`—for example

   ```
   multiversion = kernel-default,kernel-default-base,kernel-source
   ```

4. Save your changes.

Warning: Kernel Module Packages (KMP)

Make sure that required vendor provided kernel modules (Kernel Module Packages) are also installed for the new updated kernel. The kernel update process will not warn about eventually missing kernel modules because package requirements are still fulfilled by the old kernel that is kept on the system.

6.1.1 Automatically Deleting Unused Kernels

When frequently testing new kernels with multiversion support enabled, the boot menu quickly becomes confusing. Since a `/boot` partition usually has limited space you also might run into trouble with `/boot` overflowing. While you can delete unused kernel versions manually with YaST or Zypper (as described below), you can also configure `libzypp` to automatically delete kernels no longer used. By default no kernels are deleted.

1. Open `/etc/zypp/zypp.conf` with the editor of your choice as `root`.
2. Search for the string `multiversion.kernels` and activate this option by uncommenting the line. This option takes a comma-separated list of the following values:

   `4.4.126-48`: 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,4.4.126-48`

Keep the latest kernel, the one currently running, and `4.4.126-48`.

Tip: Keep the Running Kernel

Unless you are using a special setup, always keep the kernel marked `running`.

If you do not keep the running kernel, it will be deleted when updating the kernel. In turn, this means that all of the running kernel's modules are also deleted and cannot be loaded anymore.

If you decide not to keep the running kernel, always reboot immediately after a kernel upgrade to avoid issues with modules.

6.1.2 Use Case: Deleting an Old Kernel after Reboot Only

You want to make sure that an old kernel will only be deleted after the system has rebooted successfully with the new kernel.

Change the following line in `/etc/zypp/zypp.conf`:

```
multiversion.kernels = latest,running
```

The previous parameters tell the system to keep the latest kernel and the running one only if they differ.

6.1.3 Use Case: Keeping Older Kernels as Fallback

You want to keep one or more kernel versions to have one or more “spare” kernels.

This can be useful if you need kernels for testing. If something goes wrong (for example, your machine does not boot), you still can use one or more kernel versions which are known to be good.

Change the following line in `/etc/zypp/zypp.conf`:

```
multiversion.kernels = latest,latest-1,latest-2,running
```

When you reboot your system after the installation of a new kernel, the system will keep three kernels: the current kernel (configured as `latest,running`) and its two immediate predecessors (configured as `latest-1` and `latest-2`).

6.1.4 Use Case: Keeping a Specific Kernel Version

You make regular system updates and install new kernel versions. However, you are also compiling your own kernel version and want to make sure that the system will keep them.

Change the following line in `/etc/zypp/zypp.conf`:

```
multiversion.kernels = latest,3.12.28-4.20,running
```

When you reboot your system after the installation of a new kernel, the system will keep two kernels: the new and running kernel (configured as `latest,running`) and your self-compiled kernel (configured as `3.12.28-4.20`).

6.2 Installing/Removing Multiple Kernel Versions with YaST

You can install or remove multiple kernels with YaST:

1. Start YaST and open the software manager via *Software* › *Software Management*.
2. List all packages capable of providing multiple versions by choosing *View* › *Package Groups* › *Multiversion Packages*.

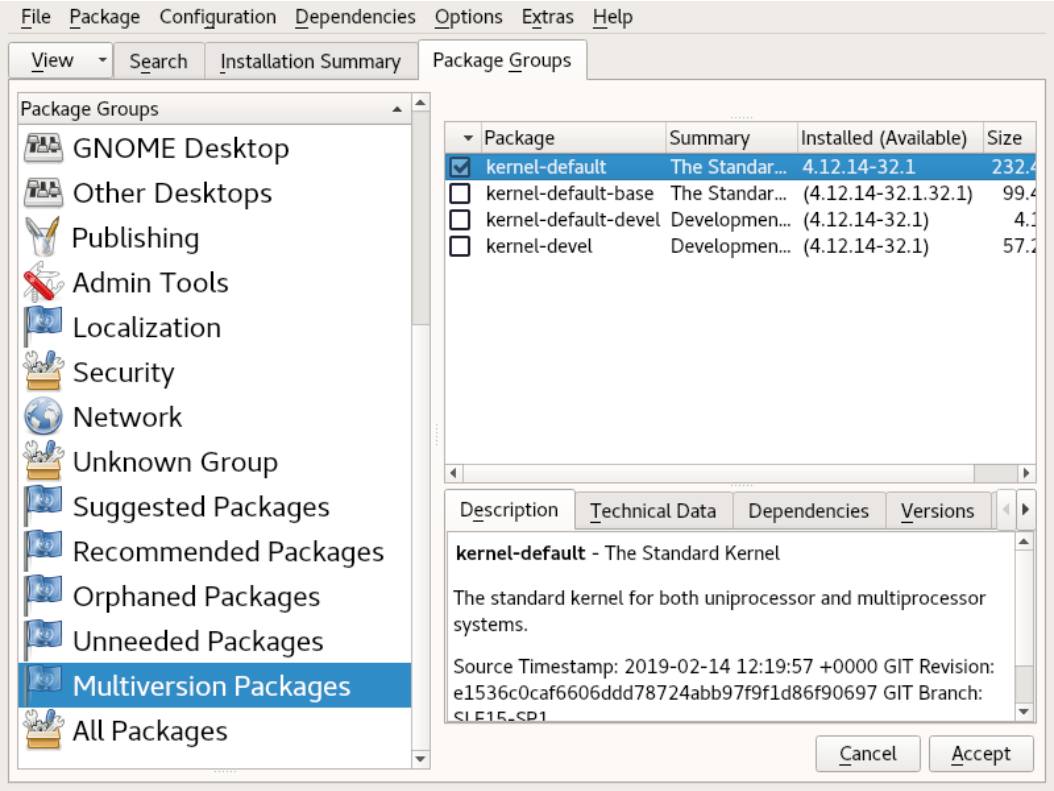


FIGURE 6.1: THE YAST SOFTWARE MANAGER: MULTIVERSION VIEW

3. Select a package and open its *Version* tab in the bottom pane on the left.

4. To install a package, click the check box next to it. A green check mark indicates it is selected for installation.
   To remove an already installed package (marked with a white check mark), click the check box next to it until a red `X` indicates it is selected for removal.

5. Click *Accept* to start the installation.

6.3 Installing/Removing Multiple Kernel Versions with Zypper

You can install or remove multiple kernels with **`zypper`**:

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

```
S | Name          | Type       | Version         | Arch   | Repository
```

```
--+---------------+------------+----------------+--------+------------------
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:

```
tux > sudo 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.

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

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

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

2. To refresh repositories, run:

```
tux > sudo zypper ref
```

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

```
tux > sudo zypper dist-upgrade --from kernel-repo
```

4. Reboot the machine.

Warning: Installing from `Kernel:HEAD` May Break the System

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

7 Graphical User Interface

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

7.1 X Window System

The X.org server is the de facto standard for implementing the X11 protocol. X is network-based, enabling applications started on one host to be displayed on another host connected over any kind of network (LAN or Internet).

Usually, the X Window System needs no configuration. The hardware is dynamically detected during X start-up. The use of `xorg.conf` is therefore deprecated. If you still need to specify custom options to change the way X behaves, you can still do so by modifying configuration files under `/etc/X11/xorg.conf.d/`.

In openSUSE Leap 15.1 Wayland is included as an alternative to the X.org server. It can be selected during the installation.

Install the package `xorg-docs` to get more in-depth information about X11. **`man 5 xorg.conf`** tells you more about the format of the manual configuration (if needed). More information on the X11 development can be found on the project's home page at http://www.x.org.

Drivers are found in `xf86-video-*` packages, for example `xf86-video-ati`. Many of the drivers delivered with these packages are described in detail in the related manual page. For example, if you use the `ati` driver, find more information about this driver in **`man 4 ati`**.

Information about third-party drivers is available in `/usr/share/doc/packages/<package_name>`. For example, the documentation of `x11-video-nvidiaG03` is available in `/usr/share/doc/packages/x11-video-nvidiaG04` after the package was installed.

7.2 Installing and Configuring Fonts

Fonts in Linux can be categorized into two parts:

Outline or Vector Fonts

Contains a mathematical description as drawing instructions about the shape of a glyph. As such, each glyph can be scaled to arbitrary sizes without loss of quality. Before such a font (or glyph) can be used, the mathematical descriptions need to be transformed into

a raster (grid). This process is called *font rasterization*. *Font hinting* (embedded inside the font) improves and optimizes the rendering result for a particular size. Rasterization and hinting is done with the FreeType library.
Common formats under Linux are PostScript Type 1 and Type 2, TrueType, and OpenType.

Bitmap or Raster Fonts

Consists of an array of pixels designed for a specific font size. Bitmap fonts are extremely fast and simple to render. However, compared to vector fonts, bitmap fonts cannot be scaled without losing quality. As such, these fonts are usually distributed in different sizes. These days, bitmap fonts are still used in the Linux console and sometimes in terminals.
Under Linux, Portable Compiled Format (PCF) or Glyph Bitmap Distribution Format (BDF) are the most common formats.

The appearance of these fonts can be influenced by two main aspects:

- choosing a suitable font family,
- rendering the font with an algorithm that achieves results comfortable for the receiver's eyes.

The last point is only relevant to vector fonts. Although the above two points are highly subjective, some defaults need to be created.

Linux font rendering systems consist of several libraries with different relations. The basic font rendering library is FreeType (http://www.freetype.org/), which converts font glyphs of supported formats into optimized bitmap glyphs. The rendering process is controlled by an algorithm and its parameters (which may be subject to patent issues).

Every program or library which uses FreeType should consult the Fontconfig (http://www.fontconfig.org/) library. This library gathers font configuration from users and from the system. When a user amends their Fontconfig setting, this change will result in Fontconfig-aware applications.

More sophisticated OpenType shaping needed for scripts such as Arabic, Han or Phags-Pa and other higher level text processing is done using Harfbuzz (http://www.harfbuzz.org/) or Pango (http://www.pango.org/).

7.2.1 Showing Installed Fonts

To get an overview about which fonts are installed on your system, ask the commands **rpm** or **fc-list**. Both will give you a good answer, but may return a different list depending on system and user configuration:

rpm

Invoke **rpm** to see which software packages containing fonts are installed on your system:

```
tux > rpm -qa '*fonts*'
```

Every font package should satisfy this expression. However, the command may return some false positives like `fonts-config` (which is neither a font nor does it contain fonts).

fc-list

Invoke **fc-list** to get an overview about what font families can be accessed, whether they are installed on the system or in your home:

```
tux > fc-list ':' family
```

Note: Command **fc-list**

The command **fc-list** is a wrapper to the Fontconfig library. It is possible to query a lot of interesting information from Fontconfig—or, to be more precise, from its cache. See **man 1 fc-list** for more details.

7.2.2 Viewing Fonts

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

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

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

7.2.3 Querying Fonts

To query which font is used when a pattern is given, use the **fc-match** command.

For example, if your pattern contains an already installed font, `fc-match` returns the file name, font family, and the style:

```
tux > fc-match 'Liberation Serif'
LiberationSerif-Regular.ttf: "Liberation Serif" "Regular"
```

If the desired font does not exist on your system, Fontconfig's matching rules take place and try to find the most similar fonts available. This means, your request is substituted:

```
tux > fc-match 'Foo Family'
DejaVuSans.ttf: "DejaVu Sans" "Book"
```

Fontconfig supports *aliases*: a name is substituted with another family name. A typical example are the generic names such as “sans-serif”, “serif”, and “monospace”. These alias names can be substituted by real family names or even a preference list of family names:

```
tux > for font in serif sans mono; do fc-match "$font" ; done
DejaVuSerif.ttf: "DejaVu Serif" "Book"
DejaVuSans.ttf: "DejaVu Sans" "Book"
DejaVuSansMono.ttf: "DejaVu Sans Mono" "Book"
```

The result may vary on your system, depending on which fonts are currently installed.

Note: Similarity Rules according to Fontconfig

Fontconfig *always* returns a real family (if at least one is installed) according to the given request, as similar as possible. “Similarity” depends on Fontconfig's internal metrics and on the user's or administrator's Fontconfig settings.

7.2.4 Installing Fonts

To install a new font there are these major methods:

1. Manually install the font files such as `*.ttf` or `*.otf` to a known font directory. If it needs to be system-wide, use the standard directory `/usr/share/fonts`. For installation in your home directory, use `~/.config/fonts`.

If you want to deviate from the standard directories, Fontconfig allows you to choose another one. Let Fontconfig know by using the `<dir>` element, see *Section 7.2.5.2, "Diving into Fontconfig XML"* for details.

2. Install fonts using **`zypper`**. Lots of fonts are already available as a package, be it on your SUSE distribution or in the M17N:fonts (http://download.opensuse.org/repositories/M17N:/fonts/) repository. Add the repository to your list using the following command. For example, to add a repository for openSUSE Leap 15:

```
tux > sudo zypper ar
     http://download.opensuse.org/repositories/M17N:/fonts/openSUSE_Leap_15.0/
```

To search for your *`FONT_FAMILY_NAME`* use this command:

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

7.2.5 Configuring the Appearance of Fonts

Depending on the rendering medium, and font size, the result may be unsatisfactory. For example, an average monitor these days has a resolution of 100dpi which makes pixels too big and glyphs look clunky.

There are several algorithms available to deal with low resolutions, such as anti-aliasing (grayscale smoothing), hinting (fitting to the grid), or subpixel rendering (tripling resolution in one direction). These algorithms can also differ from one font format to another.

Important: Patent Issues with Subpixel Rendering

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

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

7.2.5.1 Configuring Fonts via `sysconfig`

openSUSE Leap comes with a `sysconfig` layer above Fontconfig. This is a good starting point for experimenting with font configuration. To change the default settings, edit the configuration file `/etc/sysconfig/fonts-config`. (or use the YaST sysconfig module). After you have edited the file, run **`fonts-config`**:

```
tux > sudo /usr/sbin/fonts-config
```

Restart the application to make the effect visible. Keep in mind the following issues:

- A few applications do need not to be restarted. For example, Firefox re-reads Fontconfig configuration from time to time. Newly created or reloaded tabs get new font configurations later.
- The **`fonts-config`** script is called automatically after every package installation or removal (if not, it is a bug of the font software package).
- Every sysconfig variable can be temporarily overridden by the **`fonts-config`** command line option. See **`fonts-config --help`** for details.

There are several sysconfig variables which can be altered. See **`man 1 fonts-config`** or the help page of the YaST sysconfig module. The following variables are examples:

Usage of Rendering Algorithms

Consider `FORCE_HINTSTYLE`, `FORCE_AUTOHINT`, `FORCE_BW`, `FORCE_BW_MONOSPACE`, `USE_EMBEDDED_BITMAPS` and `EMBEDDED_BITMAP_LANGAGES`

Preference Lists of Generic Aliases

Use `PREFER_SANS_FAMILIES`, `PREFER_SERIF_FAMILIES`, `PREFER_MONO_FAMILIES` and `SEARCH_METRIC_COMPATIBLE`

The following list provides some configuration examples, sorted from the “most readable” fonts (more contrast) to “most beautiful” (more smoothed).

Bitmap Fonts

Prefer bitmap fonts via the `PREFER_*_FAMILIES` variables. Follow the example in the help section for these variables. Be aware that these fonts are rendered black and white, not smoothed and that bitmap fonts are available in several sizes only. Consider using

```
SEARCH_METRIC_COMPATIBLE="no"
```

to disable metric compatibility-driven family name substitutions.

Scalable Fonts Rendered Black and White

Scalable fonts rendered without antialiasing can result in a similar outcome to bitmap fonts, while maintaining font scalability. Use well hinted fonts like the Liberation families. Unfortunately, there is a lack of well hinted fonts though. Set the following variable to force this method:

```
FORCE_BW="yes"
```

Monospaced Fonts Rendered Black and White

Render monospaced fonts without antialiasing only, otherwise use default settings:

```
FORCE_BW_MONOSPACE="yes"
```

Default Settings

All fonts are rendered with antialiasing. Well hinted fonts will be rendered with the *byte code interpreter* (BCI) and the rest with autohinter (`hintstyle=hintslight`). Leave all relevant sysconfig variables to the default setting.

CFF Fonts

Use fonts in CFF format. They can be considered also more readable than the default TrueType fonts given the current improvements in FreeType2. Try them out by following the example of `PREFER_*_FAMILIES`. Possibly make them more dark and bold with:

```
SEARCH_METRIC_COMPATIBLE="no"
```

as they are rendered by `hintstyle=hintslight` by default. Also consider using:

```
SEARCH_METRIC_COMPATIBLE="no"
```

Autohinter Exclusively

Even for a well hinted font, use FreeType2's autohinter. That can lead to thicker, sometimes fuzzier letter shapes with lower contrast. Set the following variable to activate this:

```
FORCE_AUTOHINTER="yes"
```

Use `FORCE_HINTSTYLE` to control the level of hinting.

7.2.5.2 Diving into Fontconfig XML

Fontconfig's configuration format is the *eXtensible Markup Language* (XML). These few examples are not a complete reference, but a brief overview. Details and other inspiration can be found in **`man 5 fonts-conf`** or in `/etc/fonts/conf.d/`.

The central Fontconfig configuration file is `/etc/fonts/fonts.conf`, which—along other work—includes the whole `/etc/fonts/conf.d/` directory. To customize Fontconfig, there are two places where you can insert your changes:

FONTCONFIG CONFIGURATION FILES

1. **System-wide changes.** Edit the file `/etc/fonts/local.conf` (by default, it contains an empty `fontconfig` element).
2. **User-specific changes.** Edit the file `~/.config/fontconfig/fonts.conf`. Place Fontconfig configuration files in the `~/.config/fontconfig/conf.d/` directory.

User-specific changes overwrite any system-wide settings.

Note: Deprecated User Configuration File

The file `~/.fonts.conf` is marked as deprecated and should not be used anymore. Use `~/.config/fontconfig/fonts.conf` instead.

Every configuration file needs to have a `fontconfig` element. As such, the minimal file looks like this:

```
<?xml version="1.0"?>
   <!DOCTYPE fontconfig SYSTEM "fonts.dtd">
   <fontconfig>
   <!-- Insert your changes here -->
   </fontconfig>
```

If the default directories are not enough, insert the `dir` element with the respective directory:

```
<dir>/usr/share/fonts2</dir>
```

Fontconfig searches *recursively* for fonts.

Font-rendering algorithms can be chosen with following Fontconfig snippet (see *Example 7.1, "Specifying Rendering Algorithms"*):

EXAMPLE 7.1: SPECIFYING RENDERING ALGORITHMS

```
<match target="font">
 <test name="family">
  <string>FAMILY_NAME</string>
 </test>
 <edit name="antialias" mode="assign">
```

```
  <bool>true</bool>
 </edit>
 <edit name="hinting" mode="assign">
  <bool>true</bool>
 </edit>
 <edit name="autohint" mode="assign">
  <bool>false</bool>
 </edit>
 <edit name="hintstyle" mode="assign">
  <const>hintfull</const>
 </edit>
</match>
```

Various properties of fonts can be tested. For example, the `<test>` element can test for the font family (as shown in the example), size interval, spacing, font format, and others. When abandoning `<test>` completely, all `<edit>` elements will be applied to every font (global change).

EXAMPLE 7.2: ALIASES AND FAMILY NAME SUBSTITUTIONS

Rule 1

```
<alias>
 <family>Alegreya SC</family>
 <default>
  <family>serif</family>
 </default>
</alias>
```

Rule 2

```
<alias>
 <family>serif</family>
 <prefer>
  <family>Droid Serif</family>
 </prefer>
</alias>
```

Rule 3

```
<alias>
 <family>serif</family>
 <accept>
  <family>STIXGeneral</family>
 </accept>
</alias>
```

The rules from *Example 7.2, "Aliases and Family Name Substitutions"* create a *prioritized family list* (PFL). Depending on the element, different actions are performed:

`<default>` from *Rule 1*
: This rule adds a `serif` family name *at the end* of the PFL.

`<prefer>` from *Rule 2*
: This rule adds "Droid Serif" *just before* the first occurrence of `serif` in the PFL, whenever `Alegreya SC` is in PFL.

`<accept>` from *Rule 3*
: This rule adds a "STIXGeneral" family name *just after* the first occurrence of the `serif` family name in the PFL.

Putting this together, when snippets occur in the order *Rule 1* - *Rule 2* - *Rule 3* and the user requests "Alegreya SC", then the PFL is created as depicted in *Table 7.1, "Generating PFL from Fontconfig rules"*.

TABLE 7.1: GENERATING PFL FROM FONTCONFIG RULES

Order	Current PFL
Request	`Alegreya SC`
Rule 1	`Alegreya SC`, `serif`
Rule 2	`Alegreya SC`, `Droid Serif`, `serif`
Rule 3	`Alegreya SC`, `Droid Serif`, `serif`, `STIXGeneral`

In Fontconfig's metrics, the family name has the highest priority over other patterns, like style, size, etc. Fontconfig checks which family is currently installed on the system. If "Alegreya SC" is installed, then Fontconfig returns it. If not, it asks for "Droid Serif", etc.

Be careful. When the order of Fontconfig snippets is changed, Fontconfig can return different results, as depicted in *Table 7.2, "Results from Generating PFL from Fontconfig Rules with Changed Order"*.

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

Order	Current PFL	Note
Request	`Alegreya SC`	Same request performed.

Order	Current PFL	Note
Rule 2	`Alegreya SC`	`serif` not in PFL, nothing is substituted
Rule 3	`Alegreya SC`	`serif` not in PFL, nothing is substituted
Rule 1	`Alegreya SC`, `serif`	`Alegreya SC` present in PFL, substitution is performed

Note: Implication

Think of the `<default>` alias as a classification or inclusion of this group (if not installed). As the example shows, `<default>` should always precede the `<prefer>` and `<accept>` aliases of that group.

`<default>` classification is not limited to the generic aliases serif, sans-serif and monospace. See `/usr/share/fontconfig/conf.avail/30-metric-aliases.conf` for a complex example.

The following Fontconfig snippet in *Example 7.3, "Aliases and Family Name Substitutions"* creates a `serif` group. Every family in this group could substitute others when a former font is not installed.

EXAMPLE 7.3: ALIASES AND FAMILY NAME SUBSTITUTIONS

```
<alias>
 <family>Alegreya SC</family>
 <default>
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>Droid Serif</family>
 <default>
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>STIXGeneral</family>
```

```
 <default>
  <family>serif</family>
 </default>
</alias>
<alias>
 <family>serif</family>
 <accept>
  <family>Droid Serif</family>
  <family>STIXGeneral</family>
  <family>Alegreya SC</family>
 </accept>
</alias>
```

Priority is given by the order in the `<accept>` alias. Similarly, stronger `<prefer>` aliases can be used.

Example 7.2, "Aliases and Family Name Substitutions" is expanded by *Example 7.4, "Aliases and Family Names Substitutions"*.

EXAMPLE 7.4: ALIASES AND FAMILY NAMES SUBSTITUTIONS

Rule 4

```
<alias>
 <family>serif</family>
 <accept>
  <family>Liberation Serif</family>
 </accept>
</alias>
```

Rule 5

```
<alias>
 <family>serif</family>
 <prefer>
  <family>DejaVu Serif</family>
 </prefer>
</alias>
```

The expanded configuration from *Example 7.4, "Aliases and Family Names Substitutions"* would lead to the following PFL evolution:

TABLE 7.3: RESULTS FROM GENERATING PFL FROM FONTCONFIG RULES

Order	Current PFL
Request	`Alegreya SC`

Order	Current PFL
Rule 1	`Alegreya SC`, `serif`
Rule 2	`Alegreya SC`, `Droid Serif`, `serif`
Rule 3	`Alegreya SC`, `Droid Serif`, `serif`, `STIXGeneral`
Rule 4	`Alegreya SC`, `Droid Serif`, `serif`, `Liberation Serif`, `STIXGeneral`
Rule 5	`Alegreya SC`, `Droid Serif`, `DejaVu Serif`, `serif`, `Liberation Serif`, `STIXGeneral`

Note: Implications.

- In case multiple `<accept>` declarations for the same generic name exist, the declaration that is parsed last "wins". If possible, do not use `<accept>` **after** user (`/etc/fonts/conf.d/*-user.conf`) when creating a system-wide configuration.
- In case multiple `<prefer` declarations for the same generic name exist, the declaration that is parsed last "wins". If possible, do not use `<prefer>` **before** user in the system-wide configuration.
- Every `<prefer>` declaration overwrites `<accept>` declarations for the same generic name. If the administrator wants to allow the user to use `<accept>` and not only `<prefer>`, the administrator should not use `<prefer>` in the system-wide configuration. On the other hand, as users mostly use `<prefer>`, this should not have any detrimental effect. We also see the use of `<prefer>` in system-wide configurations.

7.3 GNOME Configuration for Administrators

7.3.1 The dconf System

Configuration of the GNOME desktop is managed with `dconf`. It is a hierarchically structured database or registry that allows users to modify their personal settings, and system administrators to set default or mandatory values for all users. `dconf` replaces the `gconf` system of GNOME 2.

Use **`dconf-editor`** to view the `dconf` options with a graphical user interface. Use **`dconf`** to access and modify configuration options with the command line.

The GNOME `Tweaks` tool provides an easy-to-use user interface for additional configuration options beyond the normal GNOME configuration. The tool can be started from the GNOME application menu or from the command line with **`gnome-tweak-tool`**.

7.3.2 System-wide Configuration

Global `dconf` configuration parameters can be set in the `/etc/dconf/db/` directory. This includes the configuration for GDM or locking certain configuration options for users.

Use the following procedure as an example to create a system-wide configuration:

1. Create a new directory that ends with a `.d` in `/etc/dconf/db/`. This directory can contain an arbitrary amount of text files with configuration options. For this example, create the file `/etc/dconf/db/network/00-proxy` with the following content:

   ```
   # This is a comment
   [system/proxy/http]
   host='10.0.0.1'
   enabled=true
   ```

2. Parse the new configuration directives into the dconf database format:

   ```
   tux > sudo dconf update
   ```

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

   ```
   system-db:network
   ```

The file `/etc/dconf/profiles/user` is a GNOME default that will be used. Other profiles can be defined in the environment variable `DCONF_PROFILE`.

4. Optional: To lock the proxy configuration for users, create the file `/etc/dconf/db/network/locks/proxy`. Then add a line to this file with the keys that may not be changed:

```
/system/proxy/http/host
/system/proxy/http/enabled
```

You can use the graphical **`dconf-editor`** to create a profile with one user and then use **`dconf dump /`** to list all configuration options. The configuration options can then be stored in a global profile.

A detailed description of the global configuration is available at https://wiki.gnome.org/Projects/dconf/SystemAdministrators.

7.3.3 More Information

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

II System

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

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

openSUSE Leap for the 64-bit platforms AMD64 and Intel 64 is designed so that existing 32-bit applications run in the 64-bit environment “out-of-the-box.” This support means that you can continue to use your preferred 32-bit applications without waiting for a corresponding 64-bit port to become available.

Note: No Support for Building 32-bit Applications

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

8.1 Runtime Support

Important: Conflicts Between Application Versions

If an application is available for both 32-bit and 64-bit environments, installing both versions may cause problems. In such cases, decide on one version to install to avoid potential runtime errors.

An exception to this rule is PAM (pluggable authentication modules). openSUSE Leap uses PAM in the authentication process as a layer that mediates between user and application. Always install both PAM versions on 64-bit operating systems that also run 32-bit applications.

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

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

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

If the data content of 32-bit subdirectories under `/lib` does not depend on word size, they are not moved. This scheme conforms to LSB (Linux Standards Base) and FHS (File System Hierarchy Standard).

8.2 Kernel Specifications

The 64-bit kernels for AMD64/Intel 64 offer both a 64-bit and a 32-bit kernel ABI (application binary interface). The latter is identical to the ABI for the corresponding 32-bit kernel. This means that communication between both 32-bit and 64-bit applications with 64-bit kernels are identical.

The 32-bit system call emulation for 64-bit kernels does not support all the APIs used by system programs. This depends on the platform. For this reason, few applications, like **`lspci`**, must be compiled.

A 64-bit kernel can only load 64-bit kernel modules. You must compile 64-bit modules specifically for 64-bit kernels. It is not possible to use 32-bit kernel modules with 64-bit kernels.

Tip: Kernel-loadable Modules

Some applications require separate kernel-loadable modules. If you intend to use a 32-bit application in a 64-bit system environment, contact the provider of the application and SUSE. Make sure that the 64-bit version of the kernel-loadable module and the 32-bit compiled version of the kernel API are available for this module.

9 Introduction to the Boot Process

Booting a Linux system involves different components and tasks. After a firmware and hardware initialization process, which depends on the machine's architecture, the kernel is started by means of the boot loader GRUB 2. After this point, the boot process is completely controlled by the operating system and handled by `systemd`. `systemd` provides a set of "targets" that boot configurations for everyday usage, maintenance or emergencies.

9.1 Terminology

This chapter uses terms that can be interpreted ambiguously. To understand how they are used here, read the definitions below:

`init`

Two different processes are commonly named "init":

- The `initramfs` process mounting the root file system
- The operating system process that starts all other processes that is executed from the real root file system

In both cases, the `systemd` program is taking care of this task. It is first executed from the `initramfs` to mount the root file system. Once that has succeeded, it is re-executed from the root file system as the initial process. To avoid confusing these two `systemd` processes, we refer to the first process as *init on initramfs* and to the second one as *systemd*.

`initrd/initramfs`

An `initrd` (initial RAM disk) is an image file containing a root file system image which is loaded by the kernel and mounted from `/dev/ram` as the temporary root file system. Mounting this file system requires a file system driver.

Beginning with kernel 2.6.13, the initrd has been replaced by the `initramfs` (initial RAM file system), which does not require a file system driver to be mounted. openSUSE Leap exclusively uses an `initramfs`. However, since the `initramfs` is stored as `/boot/initrd`, it is often called "initrd". In this chapter we exclusively use the name `initramfs`.

9.2 The Linux Boot Process

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

1. *Section 9.2.1, "The Initialization and Boot Loader Phase"*
2. *Section 9.2.2, "The Kernel Phase"*
3. *Section 9.2.3, "The init on initramfs Phase"*
4. *Section 9.2.4, "The systemd Phase"*

9.2.1 The Initialization and Boot Loader Phase

During the initialization phase the machine's hardware is set up and the devices are prepared. This process differs significantly between hardware architectures.

openSUSE Leap uses the boot loader GRUB 2 on all architectures. Depending on the architecture and firmware, starting the GRUB 2 boot loader can be a multi-step process. The purpose of the boot loader is to load the kernel and the initial, RAM-based file system (initramfs). For more information about GRUB 2, refer to *Chapter 12, The Boot Loader GRUB 2*.

9.2.1.1 Initialization and Boot Loader Phase on AArch64 and AMD64/Intel 64

After turning on the computer, the BIOS or the UEFI initializes the screen and keyboard, and tests the main memory. Up to this stage, the machine does not access any mass storage media. Subsequently, the information about the current date, time, and the most important peripherals are loaded from the CMOS values. When the boot media and its geometry are recognized, the system control passes from the BIOS/UEFI to the boot loader.

On a machine equipped with a traditional BIOS, only code from the first physical 512-byte data sector (the Master Boot Record, MBR) of the boot disk can be loaded. Only a minimal GRUB 2 fits into the MBR. Its sole purpose is to load a GRUB 2 core image containing file system drivers from the gap between the MBR and the first partition (MBR partition table) or from the BIOS boot partition (GPT partition table). This image contains file system drivers and therefore is able to access `/boot` located on the root file system. `/boot` contains additional modules for GRUB 2 core as well as the kernel and the initramfs image. Once it has access to this partition, GRUB 2 loads the kernel and the initramfs image into memory and hands control over to the kernel.

When booting a BIOS system from an encrypted file system that includes an encrypted `/boot` partition, you need to enter the password for decryption twice. It is first needed by GRUB 2 to decrypt `/boot` and then for `systemd` to mount the encrypted volumes.

On machines with UEFI the boot process is much simpler than on machines with a traditional BIOS. The firmware is able to read from a FAT formatted system partition of disks with a GPT partition table. This EFI system-partition (in the running system mounted as `/boot/efi`) holds enough space to host a fully-fledged GRUB 2 which is directly loaded and executed by the firmware.

If the BIOS/UEFI supports network booting, it is also possible to configure a boot server that provides the boot loader. The system can then be booted via PXE. The BIOS/UEFI acts as the boot loader. It gets the boot image from the boot server and starts the system. This is completely independent of local hard disks.

9.2.2 The Kernel Phase

When the boot loader has passed on system control, the boot process is the same on all architectures. The boot loader loads both the kernel and an initial RAM-based file system (`initramfs`) into memory and the kernel takes over.

After the kernel has set up memory management and has detected the CPU type and its features, it initializes the hardware and mounts the temporary root file system from the memory that was loaded with the `initramfs`.

9.2.2.1 The `initramfs` file

`initramfs` (initial RAM file system) is a small cpio archive that the kernel can load into a RAM disk. It is located at `/boot/initrd`. It can be created with a tool called **`dracut`**—refer to **`man 8 dracut`** for details.

The `initramfs` provides a minimal Linux environment that enables the execution of programs before the actual root file system is mounted. This minimal Linux environment is loaded into memory by BIOS or UEFI routines and does not have specific hardware requirements other than sufficient memory. The `initramfs` archive must always provide an executable named `init` that executes the `systemd` daemon on the root file system for the boot process to proceed.

Before the root file system can be mounted and the operating system can be started, the kernel needs the corresponding drivers to access the device on which the root file system is located. These drivers may include special drivers for certain kinds of hard disks or even network drivers to access a network file system. The needed modules for the root file system are loaded by `init` on `initramfs`. After the modules are loaded, `udev` provides the `initramfs` with the needed devices. Later in the boot process, after changing the root file system, it is necessary to regenerate the devices. This is done by the `systemd` unit `systemd-udev-trigger.service`.

9.2.2.1.1 Regenerating the initramfs

Because the `initramfs` contains drivers, it needs to be updated whenever a new version of one of its drivers is available. This is done automatically when installing the package containing the driver update. YaST or zypper will inform you about this by showing the output of the command that generates the `initramfs`. However, there are some occasions when you need to regenerate an `initramfs` manually:

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

Adding Drivers Because of Hardware Changes

If you need to change hardware (for example, hard disks), and this hardware requires different drivers to be in the kernel at boot time, you must update the `initramfs` file. Open or create `/etc/dracut.conf.d/10-`*`DRIVER`*`.conf` and add the following line (mind the leading whitespace):

```
force_drivers+=" DRIVER1"
```

Replace *`DRIVER1`* with the module name of the driver. If you need to add more than one driver, list them space-separated:

```
force_drivers+=" DRIVER1 DRIVER2"
```

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

Moving System Directories to a RAID or LVM

Whenever you move swap files, or system directories like `/usr` in a running system to a RAID or logical volume, you need to create an `initramfs` that contains support for software RAID or LVM drivers.

To do so, create the respective entries in `/etc/fstab` and mount the new entries (for example with **`mount -a`** and/or **`swapon -a`**).

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

Adding Disks to an LVM Group or Btrfs RAID Containing the Root File System

Whenever you add (or remove) a disk to a logical volume group or a Btrfs RAID containing the root file system, you need to create an `initramfs` that contains support for the enlarged volume. Follow the instructions at *Procedure 9.1, "Generate an initramfs"*.

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

Changing Kernel Variables

If you change the values of kernel variables via the **`sysctl`** interface by editing related files (`/etc/sysctl.conf` or `/etc/sysctl.d/*.conf`), the change will be lost on the next system reboot. Even if you load the values with **`sysctl --system`** at runtime, the changes are not saved into the `initramfs` file. You need to update it by proceeding as outlined in *Procedure 9.1, "Generate an initramfs"*.

PROCEDURE 9.1: GENERATE AN INITRAMFS

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

1. Generate a new `initramfs` file by running

```
dracut MY_INITRAMFS
```

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

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

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

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

9.2.3 The init on initramfs Phase

The temporary root file system mounted by the kernel from the `initramfs` contains the executable `systemd` (which is called `init` on `initramfs` in the following, also see *Section 9.1, "Terminology"*. This program performs all actions needed to mount the proper root file system. It provides kernel functionality for the needed file system and device drivers for mass storage controllers with `udev`.

The main purpose of `init` on `initramfs` is to prepare the mounting of and access to the real root file system. Depending on your system configuration, `init` on `initramfs` is responsible for the following tasks.

Loading Kernel Modules

Depending on your hardware configuration, special drivers may be needed to access the hardware components of your computer (the most important component being your hard disk). To access the final root file system, the kernel needs to load the proper file system drivers.

Providing Block Special Files

The kernel generates device events depending on loaded modules. `udev` handles these events and generates the required special block files on a RAM file system in `/dev`. Without those special files, the file system and other devices would not be accessible.

Managing RAID and LVM Setups

If you configured your system to hold the root file system under RAID or LVM, `init` on `initramfs` sets up LVM or RAID to enable access to the root file system later.

Managing the Network Configuration

If you configured your system to use a network-mounted root file system (mounted via NFS), `init` must make sure that the proper network drivers are loaded and that they are set up to allow access to the root file system.

If the file system resides on a network block device like iSCSI or SAN, the connection to the storage server is also set up by `init` on `initramfs`. openSUSE Leap supports booting from a secondary iSCSI target if the primary target is not available. .

Note: Handling of Mount Failures

If the root file system fails to mount from within the boot environment, it must be checked and repaired before the boot can continue. The file system checker will be automatically started for Ext3 and Ext4 file systems. The repair process is not automated for XFS and

Btrfs file systems, and the user is presented with information describing the options available to repair the file system. When the file system has been successfully repaired, exiting the boot environment will cause the system to retry mounting the root file system. If successful, the boot will continue normally.

9.2.3.1 The init on initramfs Phase in the Installation Process

When `init` on `initramfs` is called during the initial boot as part of the installation process, its tasks differ from those mentioned above. Note that the installation system also does not start `systemd` from `initramfs`—these tasks are performed by **`linuxrc`**.

Finding the Installation Medium

When starting the installation process, your machine loads an installation kernel and a special `init` containing the YaST installer. The YaST installer is running in a RAM file system and needs to have information about the location of the installation medium to access it for installing the operating system.

Initiating Hardware Recognition and Loading Appropriate Kernel Modules

As mentioned in *Section 9.2.2.1, "The `initramfs` file"*, the boot process starts with a minimum set of drivers that can be used with most hardware configurations. On AArch64, POWER, and AMD64/Intel 64 machines, **`linuxrc`** starts an initial hardware scanning process that determines the set of drivers suitable for your hardware configuration. On IBM IBM Z, a list of drivers and their parameters needs to be provided, for example via linuxrc or a parmfile. These drivers are used to generate a custom `initramfs` that is needed to boot the system. If the modules are not needed for boot but for coldplug, the modules can be loaded with `systemd`; for more information, see *Section 10.6.4, "Loading Kernel Modules"*.

Loading the Installation System

When the hardware is properly recognized, the appropriate drivers are loaded. The `udev` program creates the special device files and **`linuxrc`** starts the installation system with the YaST installer.

Starting YaST

Finally, **`linuxrc`** starts YaST, which starts the package installation and the system configuration.

9.2.4 The systemd Phase

After the “real” root file system has been found, it is checked for errors and mounted. If this is successful, the `initramfs` is cleaned and the `systemd` daemon on the root file system is executed. `systemd` is Linux's system and service manager. It is the parent process that is started as PID 1 and acts as an init system which brings up and maintains user space services. See *Chapter 10, The* `systemd` *Daemon* for details.

10 The `systemd` Daemon

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

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

10.1 The systemd Concept

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

10.1.1 What Is systemd

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

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

10.1.2 Unit File

A unit configuration file contains information about a service, a socket, a device, a mount point, an automount point, a swap file or partition, a start-up target, a watched file system path, a timer controlled and supervised by systemd, a temporary system state snapshot, a resource management slice or a group of externally created processes. “Unit file” is a generic term used by systemd for the following:

- **Service.** Information about a process (for example running a daemon); file ends with .service
- **Targets.** Used for grouping units and as synchronization points during start-up; file ends with .target
- **Sockets.** Information about an IPC or network socket or a file system FIFO, for socket-based activation (like `inetd`); file ends with .socket
- **Path.** Used to trigger other units (for example running a service when files change); file ends with .path
- **Timer.** Information about a timer controlled, for timer-based activation; file ends with .timer
- **Mount point.** Usually auto-generated by the fstab generator; file ends with .mount
- **Automount point.** Information about a file system automount point; file ends with .automount
- **Swap.** Information about a swap device or file for memory paging; file ends with .swap
- **Device.** Information about a device unit as exposed in the sysfs/udev(7) device tree; file ends with .device
- **Scope / Slice.** A concept for hierarchically managing resources of a group of processes; file ends with .scope/.slice

For more information about systemd.unit see http://www.freedesktop.org/software/systemd/man/systemd.unit.html

10.2 Basic Usage

The System V init system uses several commands to handle services—the init scripts, **`insserv`**, **`telinit`** and others. systemd makes it easier to manage services, since there is only one command to memorize for the majority of service-handling tasks: **`systemctl`**. It uses the "command plus subcommand" notation like **`git`** or **`zypper`**:

```
systemctl GENERAL OPTIONS SUBCOMMAND SUBCOMMAND OPTIONS
```

See **`man 1 systemctl`** for a complete manual.

Tip: Terminal Output and Bash Completion

If the output goes to a terminal (and not to a pipe or a file, for example) systemd commands send long output to a pager by default. Use the `--no-pager` option to turn off paging mode.

systemd also supports bash-completion, allowing you to enter the first letters of a subcommand and then press →| to automatically complete it. This feature is only available in the `bash` shell and requires the installation of the package `bash-completion`.

10.2.1 Managing Services in a Running System

Subcommands for managing services are the same as for managing a service with System V init (**`start`**, **`stop`**, ...). The general syntax for service management commands is as follows:

systemd

```
systemctl reload|restart|start|status|stop|... MY_SERVICE(S)
```

System V init

```
rcMY_SERVICE(S) reload|restart|start|status|stop|...
```

systemd allows you to manage several services in one go. Instead of executing init scripts one after the other as with System V init, execute a command like the following:

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

To list all services available on the system:

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

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

TABLE 10.1: SERVICE MANAGEMENT COMMANDS

Task	systemd Command	System V init Command
Starting.	`start`	`start`
Stopping.	`stop`	`stop`
Restarting. Shuts down services and starts them afterward. If a service is not yet running it will be started.	`restart`	`restart`
Restarting conditionally. Restarts services if they are currently running. Does nothing for services that are not running.	`try-restart`	`try-restart`
Reloading. Tells services to reload their configuration files without interrupting operation. Use case: Tell Apache to reload a modified `httpd.conf` configuration file. Note that not all services support reloading.	`reload`	`reload`
Reloading or restarting. Reloads services if reloading is supported, otherwise restarts them. If a service is not yet running it will be started.	`reload-or-restart`	`n/a`
Reloading or restarting conditionally. Reloads services if reloading is supported, otherwise restarts them if currently running. Does nothing for services that are not running.	`reload-or-try-restart`	`n/a`

Task	systemd Command	System V init Command
Getting detailed status information. Lists information about the status of services. The `systemd` command shows details such as 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.	`status`	`status`
Getting short status information. Shows whether services are active or not.	`is-active`	`status`

10.2.2 Permanently Enabling/Disabling Services

The service management commands mentioned in the previous section let you manipulate services for the current session. systemd also lets you permanently enable or disable services, so they are automatically started when requested or are always unavailable. You can either do this by using YaST, or on the command line.

10.2.2.1 Enabling/Disabling Services on the Command Line

The following table lists enabling and disabling commands for systemd and System V init:

Important: Service Start

When enabling a service on the command line, it is not started automatically. It is scheduled to be started with the next system start-up or runlevel/target change. To immediately start a service after having enabled it, explicitly run **`systemctl start`** *`MY_SERVICE`* or **`rc`** *`MY_SERVICE`* **`start`**.

TABLE 10.2: COMMANDS FOR ENABLING AND DISABLING SERVICES

Task	`systemd` Command	System V init Command
Enabling.	**`systemctl enable`** *`MY_SERVICE(S)`*	**`insserv`** *`MY_SERVICE(S)`*, **`chkconfig -a`** *`MY_SERVICE(S)`*
Disabling.	**`systemctl disable`** *`MY_SERVICE(S)`***`.service`**	**`insserv -r`** *`MY_SERVICE(S)`*, **`chkconfig -d`** *`MY_SERVICE(S)`*
Checking. Shows whether a service is enabled or not.	**`systemctl is-enabled`** *`MY_SERVICE`*	**`chkconfig`** *`MY_SERVICE`*
Re-enabling. Similar to restarting a service, this command first disables and then enables a service. Useful to re-enable a service with its defaults.	**`systemctl reenable`** *`MY_SERVICE`*	n/a
Masking. After "disabling" a service, it can still be started manually. To completely disable a service, you need to mask it. Use with care.	**`systemctl mask`** *`MY_SERVICE`*	n/a
Unmasking. A service that has been masked can only be used again after it has been unmasked.	**`systemctl unmask`** *`MY_SERVICE`*	n/a

10.3 System Start and Target Management

The entire process of starting the system and shutting it down is maintained by systemd. From this point of view, the kernel can be considered a background process to maintain all other processes and adjust CPU time and hardware access according to requests from other programs.

10.3.1 Targets Compared to Runlevels

With System V init the system was booted into a so-called "Runlevel". A runlevel defines how the system is started and what services are available in the running system. Runlevels are numbered; the most commonly known ones are `0` (shutting down the system), `3` (multiuser with network) and `5` (multiuser with network and display manager).

systemd introduces a new concept by using so-called "target units". However, it remains fully compatible with the runlevel concept. Target units are named rather than numbered and serve specific purposes. For example, the targets `local-fs.target` and `swap.target` mount local file systems and swap spaces.

The target `graphical.target` provides a multiuser system with network and display manager capabilities and is equivalent to runlevel 5. Complex targets, such as `graphical.target` act as "meta" targets by combining a subset of other targets. Since systemd makes it easy to create custom targets by combining existing targets, it offers great flexibility.

The following list shows the most important systemd target units. For a full list refer to **`man 7 systemd.special`**.

SELECTED SYSTEMD TARGET UNITS

`default.target`

The target that is booted by default. Not a "real" target, but rather a symbolic link to another target like `graphic.target`. Can be permanently changed via YaST (see *Section 10.4, "Managing Services with YaST"*). To change it for a session, use the kernel parameter `systemd.unit=`*`MY_TARGET.target`* at the boot prompt.

`emergency.target`

Starts an emergency shell on the console. Only use it at the boot prompt as `systemd.unit=emergency.target`.

`graphical.target`

Starts a system with network, multiuser support and a display manager.

`halt.target`
: Shuts down the system.

`mail-transfer-agent.target`
: Starts all services necessary for sending and receiving mails.

`multi-user.target`
: Starts a multiuser system with network.

`reboot.target`
: Reboots the system.

`rescue.target`
: Starts a single-user system without network.

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

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

TABLE 10.3: SYSTEM V RUNLEVELS AND `systemd` TARGET UNITS

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

System V runlevel	`systemd` target	Purpose
6	`runlevel6.target`, `reboot.target`,	System reboot

Important: systemd Ignores `/etc/inittab`

The runlevels in a System V init system are configured in `/etc/inittab`. systemd does *not* use this configuration. Refer to *Section 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/ runlevel	**`systemctl isolate`** *MY_TARGET*.target	**`telinit`** *X*
Change to the default target/ runlevel	**`systemctl default`**	n/a
Get the current target/runlevel	**`systemctl list-units --type=target`** With systemd there is usually more than one active target. The command lists all currently active targets.	**`who -r`** or **`runlevel`**
persistently change the default runlevel	Use the Services Manager or run the following command: **`ln -sf /usr/lib/systemd/system/`** *MY_TARGET*.target /etc/systemd/system/ default.target	Use the Services Manager or change the line **`id:`** *X*:initdefault: in `/etc/inittab`

Task	systemd Command	System V init Command
Change the default runlevel for the current boot process	Enter the following option at the boot prompt **systemd.unit=** *MY_TARGET*.target	Enter the desired runlevel number at the boot prompt.
Show a target's/ runlevel's dependencies	**systemctl show -p "Requires"** *MY_TARGET*.target **systemctl show -p "Wants"** *MY_TARGET*.target “Requires” lists the hard dependencies (the ones that must be resolved), whereas “Wants” lists the soft dependencies (the ones that get resolved if possible).	n/a

10.3.2 Debugging System Start-Up

systemd offers the means to analyze the system start-up process. You can review the list of all services and their status (rather than having to parse `/varlog/`). systemd also allows you to scan the start-up procedure to find out how much time each service start-up consumes.

10.3.2.1 Review Start-Up of Services

To review the complete list of services that have been started since booting the system, enter the command **systemctl**. It lists all active services like shown below (shortened). To get more information on a specific service, use **systemctl status** *MY_SERVICE*.

EXAMPLE 10.1: LIST ACTIVE SERVICES

```
root # systemctl
UNIT                        LOAD   ACTIVE SUB       JOB DESCRIPTION
[...]
iscsi.service               loaded active exited    Login and scanning of iSC+
kmod-static-nodes.service   loaded active exited    Create list of required s+
libvirtd.service            loaded active running   Virtualization daemon
nscd.service                loaded active running   Name Service Cache Daemon
chronyd.service             loaded active running   NTP Server Daemon
polkit.service              loaded active running   Authorization Manager
```

```
postfix.service             loaded active running   Postfix Mail Transport Ag+
rc-local.service            loaded active exited    /etc/init.d/boot.local Co+
rsyslog.service             loaded active running   System Logging Service
[...]
LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

161 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

To restrict the output to services that failed to start, use the `--failed` option:

EXAMPLE 10.2: LIST FAILED SERVICES

```
root # systemctl --failed
UNIT                    LOAD   ACTIVE SUB    JOB DESCRIPTION
apache2.service         loaded failed failed     apache
NetworkManager.service loaded failed failed     Network Manager
plymouth-start.service loaded failed failed     Show Plymouth Boot Screen

[...]
```

10.3.2.2 Debug Start-Up Time

To debug system start-up time, systemd offers the **`systemd-analyze`** command. It shows the total start-up time, a list of services ordered by start-up time and can also generate an SVG graphic showing the time services took to start in relation to the other services.

Listing the System Start-Up Time

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

Listing the Services Start-Up Time

```
root # systemd-analyze blame
  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
  1080ms chronyd.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
```

10.3.2.3 Review the Complete Start-Up Process

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

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

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

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

10.3.3 System V Compatibility

systemd is compatible with System V, allowing you to still use existing System V init scripts. However, there is at least one known issue where a System V init script does not work with systemd out of the box: starting a service as a different user via **`su`** or **`sudo`** in init scripts will result in a failure of the script, producing an "Access denied" error.

When changing the user with **`su`** or **`sudo`**, a PAM session is started. This session will be terminated after the init script is finished. As a consequence, the service that has been started by the init script will also be terminated. To work around this error, proceed as follows:

1. Create a service file wrapper with the same name as the init script plus the file name extension `.service`:

   ```
   [Unit]
   Description=DESCRIPTION
   After=network.target

   [Service]
   User=USER
   Type=forking ❶
   PIDFile=PATH TO PID FILE ❶
   ExecStart=PATH TO INIT SCRIPT start
   ExecStop=PATH TO INIT SCRIPT stop
   ExecStopPost=/usr/bin/rm -f PATH TO PID FILE ❶

   [Install]
   WantedBy=multi-user.target ❷
   ```

   Replace all values written in *`UPPERCASE LETTERS`* with appropriate values.

   ❶ Optional—only use if the init script starts a daemon.

   ❷ `multi-user.target` also starts the init script when booting into `graphical.target`. If it should only be started when booting into the display manager, user `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*.

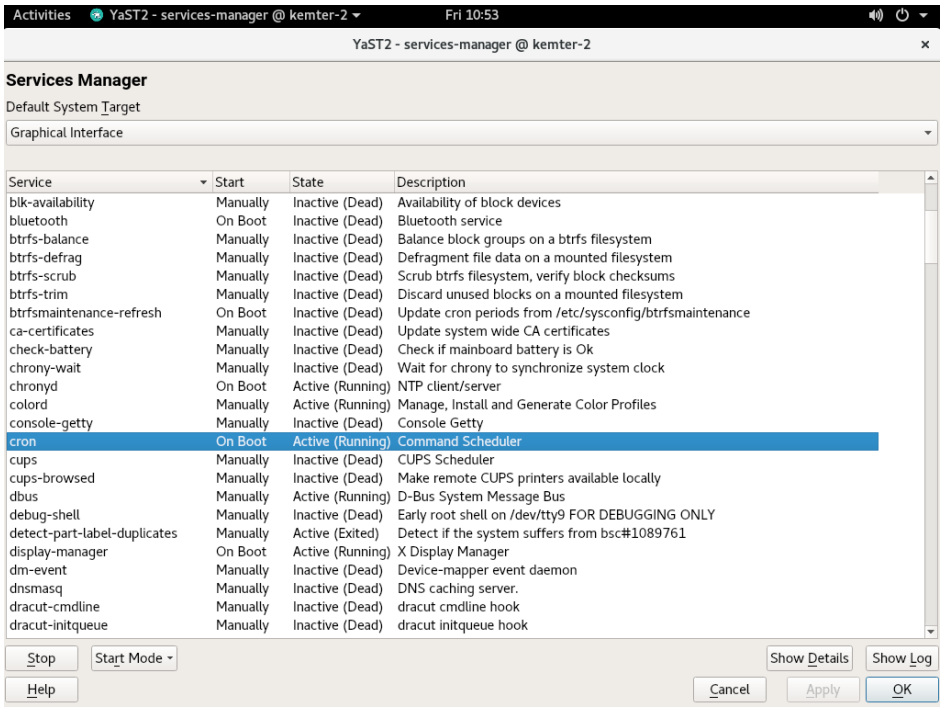


FIGURE 10.1: SERVICES MANAGER

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` customization.

Warning: Avoiding Overwritten Customization

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 Unit Files

The systemd unit 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.

10.5.1.1 Converting `xinetd` Services to `systemd`

Since the release of openSUSE Leap 15, the `xinetd` infrastructure has been removed. This section outlines how to convert existing custom `xinetd` service files to `systemd` sockets.

For each `xinetd` service file, you need at least two `systemd` unit files: the socket file (`*.socket`) and an associated service file (`*.service`). The socket file tells `systemd` which socket to create, and the service file tells `systemd` which executable to start.

Consider the following example `xinetd` service file:

```
root # cat /etc/xinetd.d/example
service example
{
  socket_type = stream
  protocol = tcp
  port = 10085
  wait = no
  user = user
  group = users
  groups = yes
  server = /usr/libexec/example/exampled
  server_args = -auth=bsdtcp exampledump
  disable = no
}
```

To convert it to `systemd`, you need the following two matching files:

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

[Install]
WantedBy=sockets.target
```

```
root # cat /usr/lib/systemd/system/example.service
[Unit]
Description=example

[Service]
ExecStart=/usr/libexec/example/exampled -auth=bsdtcp exampledump
```

```
User=user
Group=users
StandardInput=socket
```

For a complete list of the `systemd` 'socket' and 'service' file options, refer to the systemd.socket and systemd.service manual pages (`man 5 systemd.socket`, `man 5 systemd.service`).

10.5.2 Creating “Drop-in” Files

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

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

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

10.5.3 Creating Custom Targets

On System V init SUSE systems, runlevel 4 is unused to allow administrators to create their own runlevel configuration. systemd allows you to create any number of custom targets. It is suggested to start by adapting an existing target such as `graphical.target`.

1. Copy the configuration file `/usr/lib/systemd/system/graphical.target` to `/etc/systemd/system/MY_TARGET.target` and adjust it according to your needs.
2. The configuration file copied in the previous step already covers the required (“hard”) dependencies for the target. To also cover the wanted (“soft”) dependencies, create a directory `/etc/systemd/system/MY_TARGET.target.wants`.
3. For each wanted service, create a symbolic link from `/usr/lib/systemd/system` into `/etc/systemd/system/MY_TARGET.target.wants`.

4. When you have finished setting up the target, reload the systemd configuration to make the new target available:

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

10.6 Advanced Usage

The following sections cover advanced topics for system administrators. For even more advanced systemd documentation, refer to Lennart Pöttering's series about systemd for administrators at http://0pointer.de/blog/projects.

10.6.1 Cleaning Temporary Directories

`systemd` supports cleaning temporary directories regularly. The configuration from the previous system version is automatically migrated and active. `tmpfiles.d`—which is responsible for managing temporary files—reads its configuration from `/etc/tmpfiles.d/*.conf`, `/run/tmpfiles.d/*.conf`, and `/usr/lib/tmpfiles.d/*.conf` files. Configuration placed in `/etc/tmpfiles.d/*.conf` overrides related configurations from the other two directories (`/usr/lib/tmpfiles.d/*.conf` is where packages store their configuration files).

The configuration format is one line per path containing action and path, and optionally mode, ownership, age and argument fields, depending on the action. The following example unlinks the X11 lock files:

```
Type Path               Mode UID  GID  Age Argument
r    /tmp/.X[0-9]*-lock
```

To get the status the tmpfile timer:

```
tux > sudo systemctl status systemd-tmpfiles-clean.timer
systemd-tmpfiles-clean.timer - Daily Cleanup of Temporary Directories
 Loaded: loaded (/usr/lib/systemd/system/systemd-tmpfiles-clean.timer; static)
 Active: active (waiting) since Tue 2018-04-09 15:30:36 CEST; 1 weeks 6 days ago
   Docs: man:tmpfiles.d(5)
         man:systemd-tmpfiles(8)

Apr 09 15:30:36 jupiter systemd[1]: Starting Daily Cleanup of Temporary Directories.
Apr 09 15:30:36 jupiter systemd[1]: Started Daily Cleanup of Temporary Directories.
```

For more information on temporary files handling, see **`man 5 tmpfiles.d`**.

10.6.2 System Log

Section 10.6.8, "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 **`journalctl`** to display the complete log messages starting with the oldest entries. Refer to **`man 1 journalctl`** for options such as applying filters or changing the output format.

10.6.3 Snapshots

You can save the current state of `systemd` to a named snapshot and later revert to it with the **`isolate`** subcommand. This is useful when testing services or custom targets, because it allows you to return to a defined state at any time. A snapshot is only available in the current session and will automatically be deleted on reboot. A snapshot name must end in `.snapshot`.

Create a Snapshot

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

Delete a Snapshot

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

View a Snapshot

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

Activate a Snapshot

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

10.6.4 Loading Kernel Modules

With `systemd`, kernel modules can automatically be loaded at boot time via a configuration file in `/etc/modules-load.d`. The file should be named *`MODULE`*.conf and have the following content:

```
# load module MODULE at boot time
MODULE
```

In case a package installs a configuration file for loading a kernel module, the file gets installed to `/usr/lib/modules-load.d`. If two configuration files with the same name exist, the one in `/etc/modules-load.d` tales precedence.

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

10.6.5 Performing Actions before Loading a Service

With System V init actions that need to be performed before loading a service, needed to be specified in `/etc/init.d/before.local` . This procedure is no longer supported with systemd. If you need to do actions before starting services, do the following:

Loading Kernel Modules

Create a drop-in file in `/etc/modules-load.d` directory (see **`man modules-load.d`** for the syntax)

Creating Files or Directories, Cleaning-up Directories, Changing Ownership

Create a drop-in file in `/etc/tmpfiles.d` (see **`man tmpfiles.d`** for the syntax)

Other Tasks

Create a system service file, for example `/etc/systemd/system/before.service`, from the following template:

```
[Unit]
Before=NAME OF THE SERVICE YOU WANT THIS SERVICE TO BE STARTED BEFORE
[Service]
Type=oneshot
RemainAfterExit=true
ExecStart=YOUR_COMMAND
# beware, executable is run directly, not through a shell, check the man pages
# systemd.service and systemd.unit for full syntax
[Install]
# target in which to start the service
WantedBy=multi-user.target
#WantedBy=graphical.target
```

When the service file is created, you should run the following commands (as `root`):

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

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

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

10.6.6 Kernel Control Groups (cgroups)

On a traditional System V init system it is not always possible to clearly assign a process to the service that spawned it. Some services, such as Apache, spawn a lot of third-party processes (for example CGI or Java processes), which themselves spawn more processes. This makes a clear assignment difficult or even impossible. Additionally, a service may not terminate correctly, leaving some children alive.

systemd solves this problem by placing each service into its own cgroup. cgroups are a kernel feature that allows aggregating processes and all their children into hierarchical organized groups. systemd names each cgroup after its service. Since a non-privileged process is not allowed to “leave” its cgroup, this provides an effective way to label all processes spawned by a service with the name of the service.

To list all processes belonging to a service, use the command **`systemd-cgls`**. The result will look like the following (shortened) example:

EXAMPLE 10.3: LIST ALL PROCESSES BELONGING TO A SERVICE

```
root # systemd-cgls --no-pager
├─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 20
├─user.slice
| └─user-1000.slice
|   ├─session-102.scope
|   | ├─12426 gdm-session-worker [pam/gdm-password]
|   | ├─15831 gdm-session-worker [pam/gdm-password]
|   | ├─15839 gdm-session-worker [pam/gdm-password]
|   | ├─15858 /usr/lib/gnome-terminal-server

[...]

└─system.slice
  ├─systemd-hostnamed.service
  | └─17616 /usr/lib/systemd/systemd-hostnamed
  ├─cron.service
  | └─1689 /usr/sbin/cron -n
  ├─postfix.service
  | ├─ 1676 /usr/lib/postfix/master -w
  | ├─ 1679 qmgr -l -t fifo -u
  | └─15590 pickup -l -t fifo -u
  ├─sshd.service
  | └─1436 /usr/sbin/sshd -D

[...]
```

See *Book "System Analysis and Tuning Guide", Chapter 9 "Kernel Control Groups"* for more information about cgroups.

10.6.7 Terminating Services (Sending Signals)

As explained in *Section 10.6.6, "Kernel Control Groups (cgroups)"*, it is not always possible to assign a process to its parent service process in a System V init system. This makes it difficult to terminate a service and all of its children. Child processes that have not been terminated will remain as zombie processes.

systemd's concept of confining each service into a cgroup makes it possible to clearly identify all child processes of a service and therefore allows you to send a signal to each of these processes. Use **`systemctl kill`** to send signals to services. For a list of available signals refer to **`man 7 signals`**.

Sending `SIGTERM` to a Service

`SIGTERM` is the default signal that is sent.

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

Sending *`SIGNAL`* to a Service

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

```
tux > sudo systemctl kill -s SIGNAL MY_SERVICE
```

Selecting Processes

By default the **`kill`** command sends the signal to `all` processes of the specified cgroup. You can restrict it to the `control` or the `main` process. The latter is for example useful to force a service to reload its configuration by sending `SIGHUP`:

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

Warning: Terminating or Restarting the D-Bus Service Is Not Supported

The D-Bus service is the message bus for communication between systemd clients and the systemd manager that is running as pid 1. Even though `dbus` is a stand-alone daemon, it is an integral part of the initialization infrastructure.

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

Therefore, terminating or restarting `dbus` is neither recommended nor supported.

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 Apr 2018 16:52:26 +0200; 29s ago
   Process: 3088 ExecStart=/usr/sbin/start_apache2 -D SYSTEMD -k start (code=exited,
 status=1/FAILURE)
   CGroup: name=systemd:/system/apache2.service

Apr 04 16:52:26 g144 start_apache2[3088]: httpd2-prefork: Syntax error on line
205 of /etc/apache2/httpd.conf: Syntax error on li...alHost>
```

Show Last *`N`* Service Messages

The default behavior of the **`status`** subcommand is to display the last ten messages a service issued. To change the number of messages to show, use the `--lines=`*`N`* parameter:

```
tux > sudo systemctl status chronyd
tux > sudo systemctl --lines=20 status chronyd
```

Show Service Messages in Append Mode

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

```
tux > sudo systemctl --follow status chronyd
```

Messages Output Format

The `--output=`*`MODE`* parameter allows you to change the output format of service messages. The most important modes available are:

`short`
: The default format. Shows the log messages with a human readable time stamp.

`verbose`
: Full output with all fields.

`cat`
: Terse output without time stamps.

10.7 More Information

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

Homepage

http://www.freedesktop.org/wiki/Software/systemd

systemd for Administrators

Lennart Pöttering, one of the systemd authors, has written a series of blog entries (13 at the time of writing this chapter). Find them at http://0pointer.de/blog/projects.

11 `journalctl`: Query the `systemd` Journal

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

The journal itself is a system service managed by `systemd`. Its full name is `systemd-journald.service`. It collects and stores logging data by maintaining structured indexed journals based on logging information received from the kernel, user processes, standard input, and system service errors. The `systemd-journald` service is on by default:

```
tux > sudo systemctl status systemd-journald
systemd-journald.service - Journal Service
   Loaded: loaded (/usr/lib/systemd/system/systemd-journald.service; static)
   Active: active (running) since Mon 2014-05-26 08:36:59 EDT; 3 days ago
     Docs: man:systemd-journald.service(8)
           man:journald.conf(5)
 Main PID: 413 (systemd-journal)
   Status: "Processing requests..."
   CGroup: /system.slice/systemd-journald.service
           └─413 /usr/lib/systemd/systemd-journald
[...]
```

11.1 Making the Journal Persistent

The journal stores log data in `/run/log/journal/` by default. Because the `/run/` directory is volatile by nature, log data is lost at reboot. To make the log data persistent, the directory `/var/log/journal/` must exist with correct ownership and permissions so the systemd-journald service can store its data. `systemd` will create the directory for you—and switch to persistent logging—if you do the following:

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

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

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

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

```
[...]
```

3. Save the file and restart systemd-journald:

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

11.2 `journalctl` Useful Switches

This section introduces several common useful options to enhance the default `journalctl` behavior. All switches are described in the `journalctl` manual page, `man 1 journalctl`.

Tip: Messages Related to a Specific Executable

To show all journal messages related to a specific executable, specify the full path to the executable:

```
tux > sudo journalctl /usr/lib/systemd/systemd
```

-f

Shows only the most recent journal messages, and prints new log entries as they are added to the journal.

-e

Prints the messages and jumps to the end of the journal, so that the latest entries are visible within the pager.

-r

Prints the messages of the journal in reverse order, so that the latest entries are listed first.

-k

Shows only kernel messages. This is equivalent to the field match `_TRANSPORT=kernel` (see *Section 11.3.3, "Filtering Based on Fields"*).

-u

Shows only messages for the specified `systemd` unit. This is equivalent to the field match `_SYSTEMD_UNIT=UNIT` (see *Section 11.3.3, "Filtering Based on Fields"*).

```
tux > sudo journalctl -u apache2
[...]
Jun 03 10:07:11 pinkiepie systemd[1]: Starting The Apache Webserver...
```

```
Jun 03 10:07:12 pinkiepie systemd[1]: Started The Apache Webserver.
```

11.3 Filtering the Journal Output

When called without switches, **journalctl** shows the full content of the journal, the oldest entries listed first. The output can be filtered by specific switches and fields.

11.3.1 Filtering Based on a Boot Number

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

```
tux > sudo journalctl --list-boots
-1 097ed2cd99124a2391d2cffab1b566f0 Mon 2014-05-26 08:36:56 EDT—Fri 2014-05-30 05:33:44
 EDT
 0 156019a44a774a0bb0148a92df4af81b Fri 2014-05-30 05:34:09 EDT—Fri 2014-05-30 06:15:01
 EDT
```

The first column lists the boot offset: `0` for the current boot, `-1` for the previous one, `-2` for the one prior to that, etc. The second column contains the boot ID followed by the limiting time stamps of the specific boot.

Show all messages from the current boot:

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

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

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

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

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

11.3.2 Filtering Based on Time Interval

You can filter the output of **journalctl** by specifying the starting and/or ending date. The date specification should be of the format "2014-06-30 9:17:16". If the time part is omitted, midnight is assumed. If seconds are omitted, ":00" is assumed. If the date part is omitted, the current day is assumed. Instead of numeric expression, you can specify the keywords "yesterday", "today",

or "tomorrow". They refer to midnight of the day before the current day, of the current day, or of the day after the current day. If you specify "now", it refers to the current time. You can also specify relative times prefixed with `-` or `+`, referring to times before or after the current time.

Show only new messages since now, and update the output continuously:

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

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

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

11.3.3 Filtering Based on Fields

You can filter the output of the journal by specific fields. The syntax of a field to be matched is `FIELD_NAME=MATCHED_VALUE`, such as `_SYSTEMD_UNIT=httpd.service`. You can specify multiple matches in a single query to filter the output messages even more. See **`man 7 systemd.journal-fields`** for a list of default fields.

Show messages produced by a specific process ID:

```
tux > sudo journalctl _PID=1039
```

Show messages belonging to a specific user ID:

```
# journalctl _UID=1000
```

Show messages from the kernel ring buffer (the same as **`dmesg`** produces):

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

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

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

Show messages produced by a specified service only:

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

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

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

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

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

You can use the '+' separator to combine two expressions in a logical 'OR'. The following example shows all messages from the Avahi service process with the process ID 1480 together with all messages from the D-Bus service:

```
tux > sudo journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=1480 +
 _SYSTEMD_UNIT=dbus.service
```

11.4 Investigating `systemd` Errors

This section introduces a simple example to illustrate how to find and fix the error reported by `systemd` during **`apache2`** start-up.

1. Try to start the apache2 service:

   ```
   # systemctl start apache2
   Job for apache2.service failed. See 'systemctl status apache2' and 'journalctl -xn'
    for details.
   ```

2. Let us see what the service's status says:

   ```
   tux > sudo systemctl status apache2
   apache2.service - The Apache Webserver
      Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
      Active: failed (Result: exit-code) since Tue 2014-06-03 11:08:13 CEST; 7min ago
     Process: 11026 ExecStop=/usr/sbin/start_apache2 -D SYSTEMD -DFOREGROUND \
              -k graceful-stop (code=exited, status=1/FAILURE)
   ```

   The ID of the process causing the failure is 11026.

3. Show the verbose version of messages related to process ID 11026:

   ```
   tux > sudo journalctl -o verbose _PID=11026
   [...]
   MESSAGE=AH00526: Syntax error on line 6 of /etc/apache2/default-server.conf:
   [...]
   MESSAGE=Invalid command 'DocumenttRoot', perhaps misspelled or defined by a module
   [...]
   ```

4. Fix the typo inside `/etc/apache2/default-server.conf`, start the apache2 service, and print its status:

   ```
   tux > sudo systemctl start apache2 && systemctl status apache2
   apache2.service - The Apache Webserver
   ```

```
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
   Active: active (running) since Tue 2014-06-03 11:26:24 CEST; 4ms ago
  Process: 11026 ExecStop=/usr/sbin/start_apache2 -D SYSTEMD -DFOREGROUND
           -k graceful-stop (code=exited, status=1/FAILURE)
 Main PID: 11263 (httpd2-prefork)
   Status: "Processing requests..."
   CGroup: /system.slice/apache2.service
           ├─11263 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11280 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11281 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11282 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           ├─11283 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
           └─11285 /usr/sbin/httpd2-prefork -f /etc/apache2/httpd.conf -D [...]
```

11.5 Journald Configuration

The behavior of the systemd-journald service can be adjusted by modifying `/etc/systemd/journald.conf`. This section introduces only basic option settings. For a complete file description, see **`man 5 journald.conf`**. Note that you need to restart the journal for the changes to take effect with

```
tux > sudo systemctl restart systemd-journald
```

11.5.1 Changing the Journal Size Limit

If the journal log data is saved to a persistent location (see *Section 11.1, "Making the Journal Persistent"*), it uses up to 10% of the file system the `/var/log/journal` resides on. For example, if `/var/log/journal` is located on a 30 GB `/var` partition, the journal may use up to 3 GB of the disk space. To change this limit, change (and uncomment) the `SystemMaxUse` option:

```
SystemMaxUse=50M
```

11.5.2 Forwarding the Journal to `/dev/ttyX`

You can forward the journal to a terminal device to inform you about system messages on a preferred terminal screen, for example `/dev/tty12`. Change the following journald options to

```
ForwardToConsole=yes
TTYPath=/dev/tty12
```

11.5.3 Forwarding the Journal to Syslog Facility

Journald is backward compatible with traditional syslog implementations such as `rsyslog`. Make sure the following is valid:

- rsyslog is installed.

```
tux > sudo rpm -q rsyslog
rsyslog-7.4.8-2.16.x86_64
```

- rsyslog service is enabled.

```
tux > sudo systemctl is-enabled rsyslog
enabled
```

- Forwarding to syslog is enabled in `/etc/systemd/journald.conf`.

```
ForwardToSyslog=yes
```

11.6 Using YaST to Filter the `systemd` Journal

For an easy way of filtering the systemd journal (without dealing with the journalctl syntax), you can use the YaST journal module. After installing it with **`sudo zypper in yast2-journal`**, start it from YaST by selecting *System* › *Systemd Journal*. Alternatively, start it from command line by entering **`sudo yast2 journal`**.

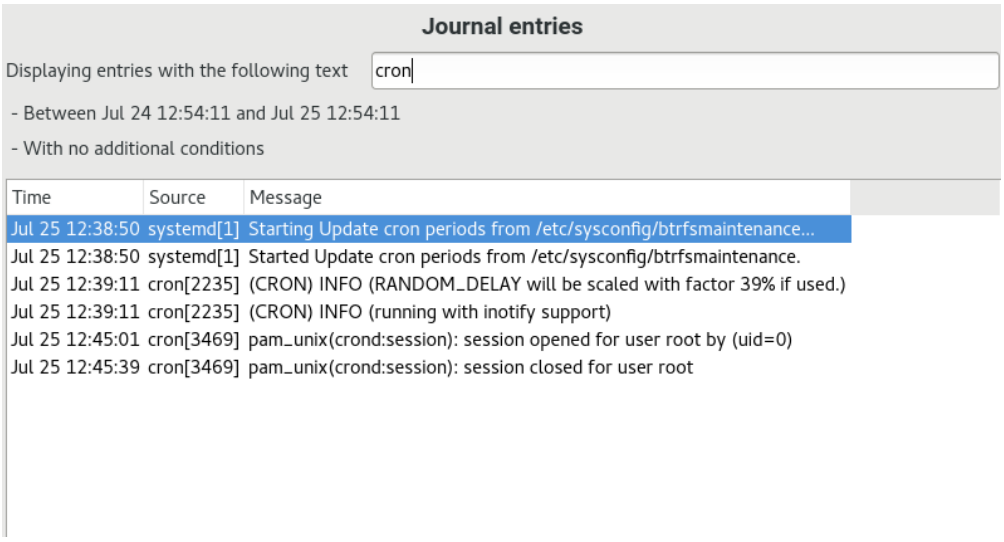


FIGURE 11.1: YAST SYSTEMD JOURNAL

The module displays the log entries in a table. The search box on top allows you to search for entries that contain certain characters, similar to using **grep**. To filter the entries by date and time, unit, file, or priority, click *Change filters* and set the respective options.

11.7 Viewing Logs in GNOME

You can view the journal with *GNOME Logs*. Start it from the application menu. To view system log messages, it needs to be run as root, for example with **xdg-su gnome-logs**. This command can be executed when pressing Alt–F2.

12 The Boot Loader GRUB 2

This chapter describes how to configure GRUB 2, the boot loader used in openSUSE® Leap. It is the successor to the traditional GRUB boot loader—now called “GRUB Legacy”. A YaST module is available for configuring the most important settings. The boot procedure as a whole is outlined in *Chapter 9, Introduction to the Boot Process*. For details on Secure Boot support for UEFI machines, see *Chapter 14, UEFI (Unified Extensible Firmware Interface)*.

12.1 Main Differences between GRUB Legacy and GRUB 2

- The configuration is stored in different files.
- More file systems are supported (for example, Btrfs).
- Can directly read files stored on LVM or RAID devices.
- The user interface can be translated and altered with themes.
- Includes a mechanism for loading modules to support additional features, such as file systems, etc.
- Automatically searches for and generates boot entries for other kernels and operating systems, such as Windows.
- Includes a minimal Bash-like console.

12.2 Configuration File Structure

The configuration of GRUB 2 is based on the following files:

`/boot/grub2/grub.cfg`

This file contains the configuration of the GRUB 2 menu items. It replaces `menu.lst` used in GRUB Legacy. `grub.cfg` should not be edited—it is automatically generated by the command **`grub2-mkconfig -o /boot/grub2/grub.cfg`**.

`/boot/grub2/custom.cfg`

This optional file is directly sourced by `grub.cfg` at boot time and can be used to add custom items to the boot menu. Starting with openSUSE Leap Leap 42.2 these entries will also be parsed when using **`grub-once`**.

`/etc/default/grub`

This file controls the user settings of GRUB 2 and usually includes additional environmental settings such as backgrounds and themes.

Scripts under `/etc/grub.d/`

The scripts in this directory are read during execution of the command **`grub2-mkconfig -o /boot/grub2/grub.cfg`**. Their instructions are integrated into the main configuration file `/boot/grub/grub.cfg`.

`/etc/sysconfig/bootloader`

This configuration file holds some basic settings like the boot loader type and whether to enable UEFI Secure Boot support.

`/boot/grub2/x86_64-efi`,

These configuration files contain architecture-specific options.

GRUB 2 can be controlled in various ways. Boot entries from an existing configuration can be selected from the graphical menu (splash screen). The configuration is loaded from the file `/boot/grub2/grub.cfg` which is compiled from other configuration files (see below). All GRUB 2 configuration files are considered system files, and you need `root` privileges to edit them.

Note: Activating Configuration Changes

After having manually edited GRUB 2 configuration files, you need to run **`grub2-mkconfig -o /boot/grub2/grub.cfg`** to activate the changes. However, this is not necessary when changing the configuration with YaST, because YaST will automatically run this command.

12.2.1 The File `/boot/grub2/grub.cfg`

The graphical splash screen with the boot menu is based on the GRUB 2 configuration file `/boot/grub2/grub.cfg`, which contains information about all partitions or operating systems that can be booted by the menu.

Every time the system is booted, GRUB 2 loads the menu file directly from the file system. For this reason, GRUB 2 does not need to be re-installed after changes to the configuration file. `grub.cfg` is automatically rebuilt with kernel installations or removals.

`grub.cfg` is compiled from the file `/etc/default/grub` and scripts found in the `/etc/grub.d/` directory when running the command **`grub2-mkconfig -o /boot/grub2/grub.cfg`**. Therefore you should never edit the file manually. Instead, edit the related source files or use the YaST *Boot Loader* module to modify the configuration as described in *Section 12.3, "Configuring the Boot Loader with YaST"*.

12.2.2 The File `/etc/default/grub`

More general options of GRUB 2 belong here, such as the time the menu is displayed, or the default OS to boot. To list all available options, see the output of the following command:

```
tux > grep "export GRUB_DEFAULT" -A50 /usr/sbin/grub2-mkconfig | grep GRUB_
```

In addition to already defined variables, the user may introduce their own variables, and use them later in the scripts found in the `/etc/grub.d` directory.

After having edited `/etc/default/grub`, update the main configuration file with **`grub2-mkconfig -o /boot/grub2/grub.cfg`**.

Note: Scope

All options set in this file are general options that affect all boot entries. Specific options for Xen kernels or the Xen hypervisor can be set via the GRUB_*_XEN_* configuration options. See below for details.

`GRUB_DEFAULT`

Sets the boot menu entry that is booted by default. Its value can be a numeric value, the complete name of a menu entry, or "saved".

`GRUB_DEFAULT=2` boots the third (counted from zero) boot menu entry.

`GRUB_DEFAULT="2>0"` boots the first submenu entry of the third top-level menu entry.

`GRUB_DEFAULT="Example boot menu entry"` boots the menu entry with the title "Example boot menu entry".

`GRUB_DEFAULT=saved` boots the entry specified by the **`grub2-once`** or **`grub2-set-default`** commands. While **`grub2-reboot`** sets the default boot entry for the next reboot only, **`grub2-set-default`** sets the default boot entry until changed. **`grub2-editenv list`** lists the next boot entry.

`GRUB_HIDDEN_TIMEOUT`

Waits the specified number of seconds for the user to press a key. During the period no menu is shown unless the user presses a key. If no key is pressed during the time specified, the control is passed to `GRUB_TIMEOUT`. `GRUB_HIDDEN_TIMEOUT=0` first checks whether `Shift` is pressed and shows the boot menu if yes, otherwise immediately boots the default menu entry. This is the default when only one bootable OS is identified by GRUB 2.

`GRUB_HIDDEN_TIMEOUT_QUIET`

If `false` is specified, a countdown timer is displayed on a blank screen when the `GRUB_HIDDEN_TIMEOUT` feature is active.

`GRUB_TIMEOUT`

Time period in seconds the boot menu is displayed before automatically booting the default boot entry. If you press a key, the timeout is cancelled and GRUB 2 waits for you to make the selection manually. `GRUB_TIMEOUT=-1` will cause the menu to be displayed until you select the boot entry manually.

`GRUB_CMDLINE_LINUX`

Entries on this line are added at the end of the boot entries for normal and recovery mode. Use it to add kernel parameters to the boot entry.

`GRUB_CMDLINE_LINUX_DEFAULT`

Same as `GRUB_CMDLINE_LINUX` but the entries are appended in the normal mode only.

`GRUB_CMDLINE_LINUX_RECOVERY`

Same as `GRUB_CMDLINE_LINUX` but the entries are appended in the recovery mode only.

`GRUB_CMDLINE_LINUX_XEN_REPLACE`

This entry will completely replace the `GRUB_CMDLINE_LINUX` parameters for all Xen boot entries.

`GRUB_CMDLINE_LINUX_XEN_REPLACE_DEFAULT`

Same as `GRUB_CMDLINE_LINUX_XEN_REPLACE` but it will only replace parameters of `GRUB_CMDLINE_LINUX_DEFAULT`.

`GRUB_CMDLINE_XEN`

This entry specifies the kernel parameters for the Xen guest kernel only—the operation principle is the same as for `GRUB_CMDLINE_LINUX`.

`GRUB_CMDLINE_XEN_DEFAULT`

Same as `GRUB_CMDLINE_XEN`—the operation principle is the same as for `GRUB_CMDLINE_LINUX_DEFAULT`.

`GRUB_TERMINAL`

Enables and specifies an input/output terminal device. Can be `console` (PC BIOS and EFI consoles), `serial` (serial terminal), `ofconsole` (Open Firmware console), or the default `gfxterm` (graphics-mode output). It is also possible to enable more than one device by quoting the required options, for example `GRUB_TERMINAL="console serial"`.

`GRUB_GFXMODE`

The resolution used for the `gfxterm` graphical terminal. Note that you can only use modes supported by your graphics card (VBE). The default is 'auto', which tries to select a preferred resolution. You can display the screen resolutions available to GRUB 2 by typing **`videoinfo`** in the GRUB 2 command line. The command line is accessed by typing C when the GRUB 2 boot menu screen is displayed.

You can also specify a color depth by appending it to the resolution setting, for example `GRUB_GFXMODE=1280x1024x24`.

`GRUB_BACKGROUND`

Set a background image for the `gfxterm` graphical terminal. The image must be a file readable by GRUB 2 at boot time, and it must end with the `.png`, `.tga`, `.jpg`, or `.jpeg` suffix. If necessary, the image will be scaled to fit the screen.

`GRUB_DISABLE_OS_PROBER`

If this option is set to `true`, automatic searching for other operating systems is disabled. Only the kernel images in `/boot/` and the options from your own scripts in `/etc/grub.d/` are detected.

`SUSE_BTRFS_SNAPSHOT_BOOTING`

If this option is set to `true`, GRUB 2 can boot directly into Snapper snapshots. For more information, see *Section 3.3, "System Rollback by Booting from Snapshots"*.

For a complete list of options, see the GNU GRUB manual (http://www.gnu.org/software/grub/manual/grub.html#Simple-configuration).

12.2.3 Scripts in `/etc/grub.d`

The scripts in this directory are read during execution of the command **`grub2-mkconfig -o /boot/grub2/grub.cfg`**. Their instructions are incorporated into `/boot/grub2/grub.cfg`. The order of menu items in `grub.cfg` is determined by the order in which the files in this directory are run. Files with a leading numeral are executed first, beginning with the lowest number. `00_header` is run before `10_linux`, which would run before `40_custom`. If files with alphabetic names are present, they are executed after the numerically-named files. Only executable files generate output to `grub.cfg` during execution of **`grub2-mkconfig`**. By default all files in the `/etc/grub.d` directory are executable.

Tip: Persistent Custom Content in `grub.cfg`

Because `/boot/grub2/grub.cfg` is recompiled each time **`grub2-mkconfig`** is run, any custom content is lost. If you want to insert your lines directly into `/boot/grub2/grub.cfg` without losing them after **`grub2-mkconfig`** is run, insert them between

```
### BEGIN /etc/grub.d/90_persistent ###
```

and

```
### END /etc/grub.d/90_persistent ###
```

The `90_persistent` script ensures that such content will be preserved.

A list of the most important scripts follows:

`00_header`

Sets environmental variables such as system file locations, display settings, themes, and previously saved entries. It also imports preferences stored in the `/etc/default/grub`. Normally you do not need to make changes to this file.

`10_linux`

Identifies Linux kernels on the root device and creates relevant menu entries. This includes the associated recovery mode option if enabled. Only the latest kernel is displayed on the main menu page, with additional kernels included in a submenu.

`30_os-prober`

This script uses **os-prober** to search for Linux and other operating systems and places the results in the GRUB 2 menu. There are sections to identify specific other operating systems, such as Windows or macOS.

`40_custom`

This file provides a simple way to include custom boot entries into `grub.cfg`. Make sure that you do not change the `exec tail -n +3 $0` part at the beginning.

The processing sequence is set by the preceding numbers with the lowest number being executed first. If scripts are preceded by the same number the alphabetical order of the complete name decides the order.

Tip: `/boot/grub2/custom.cfg`

If you create `/boot/grub2/custom.cfg` and fill it with content, it will be automatically included into `/boot/grub2/grub.cfg` just after 40_custom at boot time.

12.2.4 Mapping between BIOS Drives and Linux Devices

In GRUB Legacy, the `device.map` configuration file was used to derive Linux device names from BIOS drive numbers. The mapping between BIOS drives and Linux devices cannot always be guessed correctly. For example, GRUB Legacy would get a wrong order if the boot sequence of IDE and SCSI drives is exchanged in the BIOS configuration.

GRUB 2 avoids this problem by using device ID strings (UUIDs) or file system labels when generating `grub.cfg`. GRUB 2 utilities create a temporary device map on the fly, which is usually sufficient, particularly in the case of single-disk systems.

However, if you need to override the GRUB 2's automatic device mapping mechanism, create your custom mapping file `/boot/grub2/device.map`. The following example changes the mapping to make `DISK 3` the boot disk. Note that GRUB 2 partition numbers start with `1` and not with `0` as in GRUB Legacy.

```
(hd1)  /dev/disk-by-id/DISK3 ID
(hd2)  /dev/disk-by-id/DISK1 ID
(hd3)  /dev/disk-by-id/DISK2 ID
```

12.2.5 Editing Menu Entries during the Boot Procedure

Being able to directly edit menu entries is useful when the system does not boot anymore because of a faulty configuration. It can also be used to test new settings without altering the system configuration.

1. In the graphical boot menu, select the entry you want to edit with the arrow keys.
2. Press E to open the text-based editor.
3. Use the arrow keys to move to the line you want to edit.

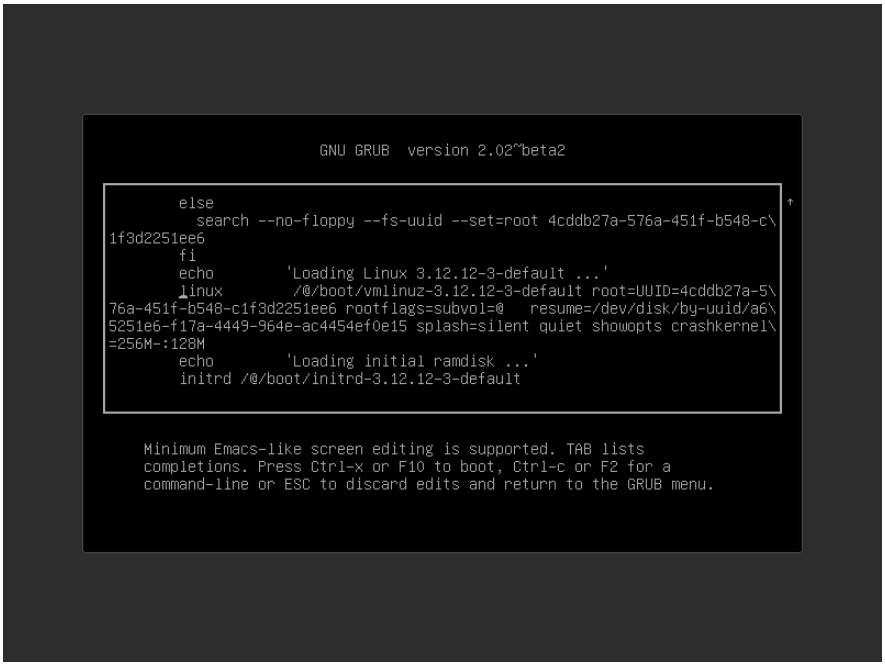


FIGURE 12.1: GRUB 2 BOOT EDITOR

Now you have two options:

a. Add space-separated parameters to the end of the line starting with `linux` or `linuxefi` to edit the kernel parameters. A complete list of parameters is available at http://en.opensuse.org/Linuxrc.

b. Or edit the general options to change for example the kernel version. The →| key suggests all possible completions.

4. Press F10 to boot the system with the changes you made or press Esc to discard your edits and return to the GRUB 2 menu.

Changes made this way only apply to the current boot process and are not saved permanently.

Important: Keyboard Layout During the Boot Procedure

The US keyboard layout is the only one available when booting. See *Book "Start-Up", Chapter 4 "Troubleshooting", Section 4.3 "Booting from Installation Media Fails", US Keyboard Layout*.

Note: Boot Loader on the Installation Media

The Boot Loader of the installation media on systems with a traditional BIOS is still GRUB Legacy. To add boot parameters, select an entry and start typing. Additions you make to the installation boot entry will be permanently saved in the installed system.

12.2.6 Setting a Boot Password

Even before the operating system is booted, GRUB 2 enables access to file systems. Users without root permissions can access files in your Linux system to which they have no access after the system is booted. To block this kind of access or to prevent users from booting certain menu entries, set a boot password.

Important: Booting Requires Password

If set, the boot password is required on every boot, which means the system does not boot automatically.

Proceed as follows to set a boot password. Alternatively use YaST (*Protect Boot Loader with Password*).

1. Encrypt the password using **`grub2-mkpasswd-pbkdf2:`**

```
tux > sudo grub2-mkpasswd-pbkdf2
Password: ****
Reenter password: ****
PBKDF2 hash of your password is grub.pbkdf2.sha512.10000.9CA4611006FE96BC77A...
```

2. Paste the resulting string into the file `/etc/grub.d/40_custom` together with the **`set superusers`** command.

```
set superusers="root"
password_pbkdf2 root grub.pbkdf2.sha512.10000.9CA4611006FE96BC77A...
```

3. To import the changes into the main configuration file, run:

```
tux > sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

After you reboot, you will be prompted for a user name and a password when trying to boot a menu entry. Enter `root` and the password you typed during the **`grub2-mkpasswd-pbkdf2`** command. If the credentials are correct, the system will boot the selected boot entry.

For more information, see https://www.gnu.org/software/grub/manual/grub.html#Security.

12.3 Configuring the Boot Loader with YaST

The easiest way to configure general options of the boot loader in your openSUSE Leap system is to use the YaST module. In the *YaST Control Center*, select *System* › *Boot Loader*. The module shows the current boot loader configuration of your system and allows you to make changes.

Use the *Boot Code Options* tab to view and change settings related to type, location and advanced loader settings. You can choose whether to use GRUB 2 in standard or EFI mode.

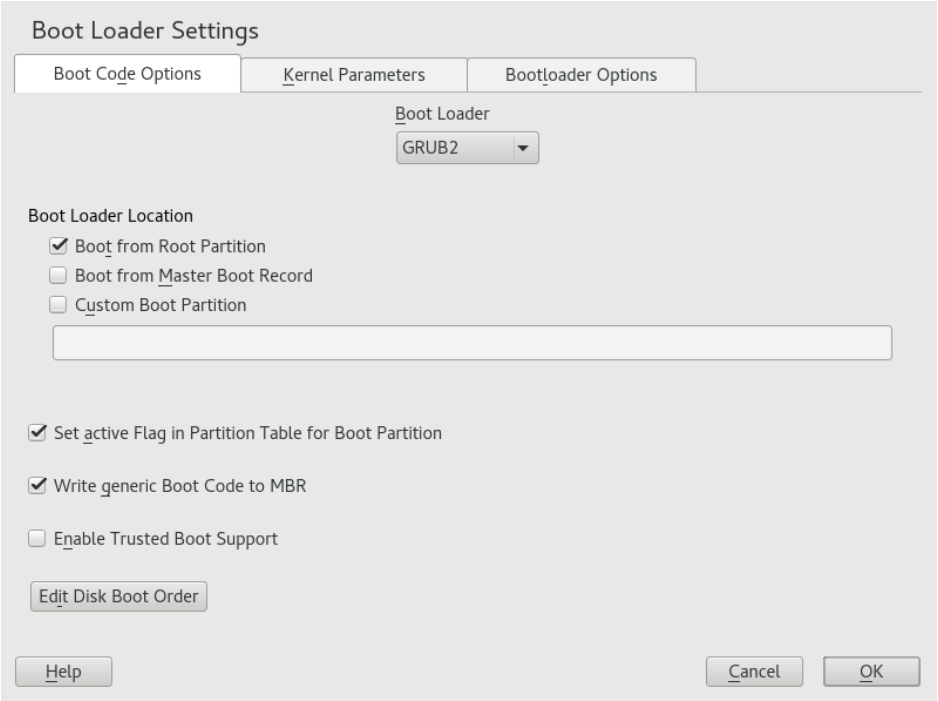


FIGURE 12.2: BOOT CODE OPTIONS

Important: EFI Systems require GRUB2-EFI

If you have an EFI system you can only install GRUB2-EFI, otherwise your system is no longer bootable.

Important: Reinstalling the Boot Loader

To reinstall the boot loader, make sure to change a setting in YaST and then change it back. For example, to reinstall GRUB2-EFI, select *GRUB2* first and then immediately switch back to *GRUB2-EFI*.

Otherwise, the boot loader may only be partially reinstalled.

Note: Custom Boot Loader

To use a boot loader other than the ones listed, select *Do Not Install Any Boot Loader*. Read the documentation of your boot loader carefully before choosing this option.

12.3.1 Boot Loader Location and Boot Code Options

The default location of the boot loader depends on the partition setup and is either the Master Boot Record (MBR) or the boot sector of the `/` partition. To modify the location of the boot loader, follow these steps:

PROCEDURE 12.1: CHANGING THE BOOT LOADER LOCATION

1. Select the *Boot Code Options* tab and then choose one of the following options for *Boot Loader Location*:

   Boot from Master Boot Record

   This installs the boot loader in the MBR of the disk containing the directory `/boot`. Usually this will be the disk mounted to `/`, but if `/boot` is mounted to a separate partition on a different disk, the MBR of that disk will be used.

   Boot from Root Partition

   This installs the boot loader in the boot sector of the `/` partition.

   Custom Boot Partition

   Use this option to specify the location of the boot loader manually.

2. Click *OK* to apply your changes.

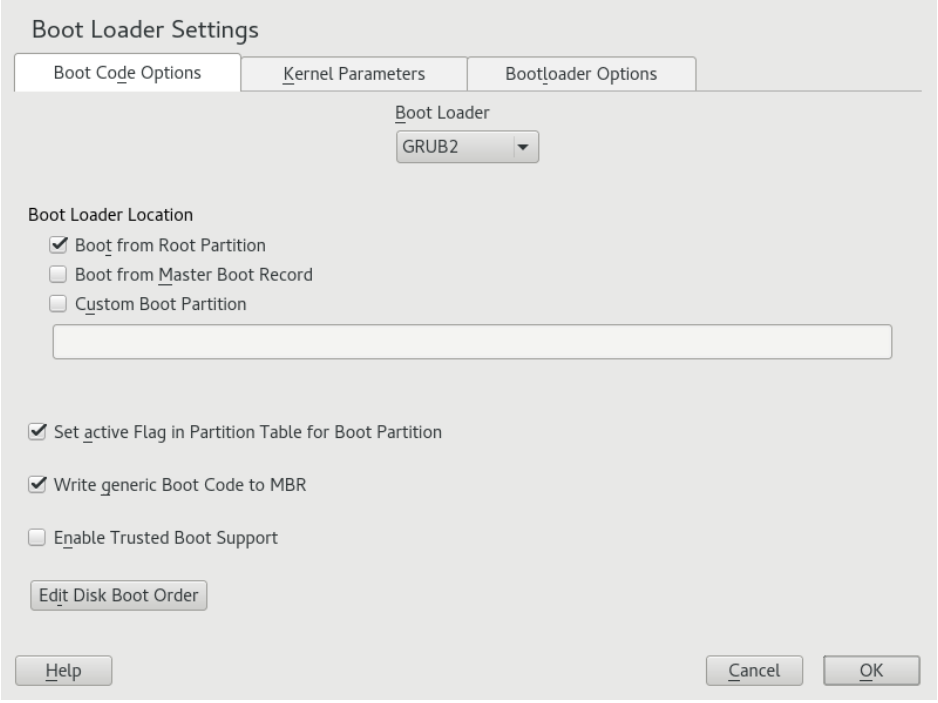


FIGURE 12.3: CODE OPTIONS

The *Boot Code Options* tab includes the following additional options:

Set Active Flag in Partition Table for Boot Partition

Activates the partition that contains the `/boot` directory. For POWER systems it activates the PReP partition. Use this option on systems with old BIOS and/or legacy operating systems because they may fail to boot from a non-active partition. It is safe to leave this option active.

Write Generic Boot Code to MBR

If MBR contains a custom 'non-GRUB' code, this option replaces it with a generic, operating system independent code. If you deactivate this option, the system may become unbootable.

Enable Trusted Boot Support

Starts TrustedGRUB2, which supports trusted computing functionality (Trusted Platform Module (TPM)). For more information refer to https://github.com/Sirrix-AG/TrustedGRUB2.

12.3.2 Adjusting the Disk Order

If your computer has more than one hard disk, you can specify the boot sequence of the disks. The first disk in the list is where GRUB 2 will be installed in the case of booting from MBR. It is the disk where openSUSE Leap is installed by default. The rest of the list is a hint for GRUB 2's device mapper (see *Section 12.2.4, "Mapping between BIOS Drives and Linux Devices"*).

Warning: Unbootable System

The default value is usually valid for almost all deployments. If you change the boot order of disks wrongly, the system may become unbootable on the next reboot. For example, if the first disk in the list is not part of the BIOS boot order, and the other disks in the list have empty MBRs.

PROCEDURE 12.2: SETTING THE DISK ORDER

1. Open the *Boot Code Options* tab.
2. Click *Edit Disk Boot Order*.

3. If more than one disk is listed, select a disk and click *Up* or *Down* to reorder the displayed disks.

4. Click *OK* two times to save the changes.

12.3.3 Configuring Advanced Options

Advanced boot parameters can be configured via the *Boot Loader Options* tab.

12.3.3.1 *Boot Loader Options* Tab

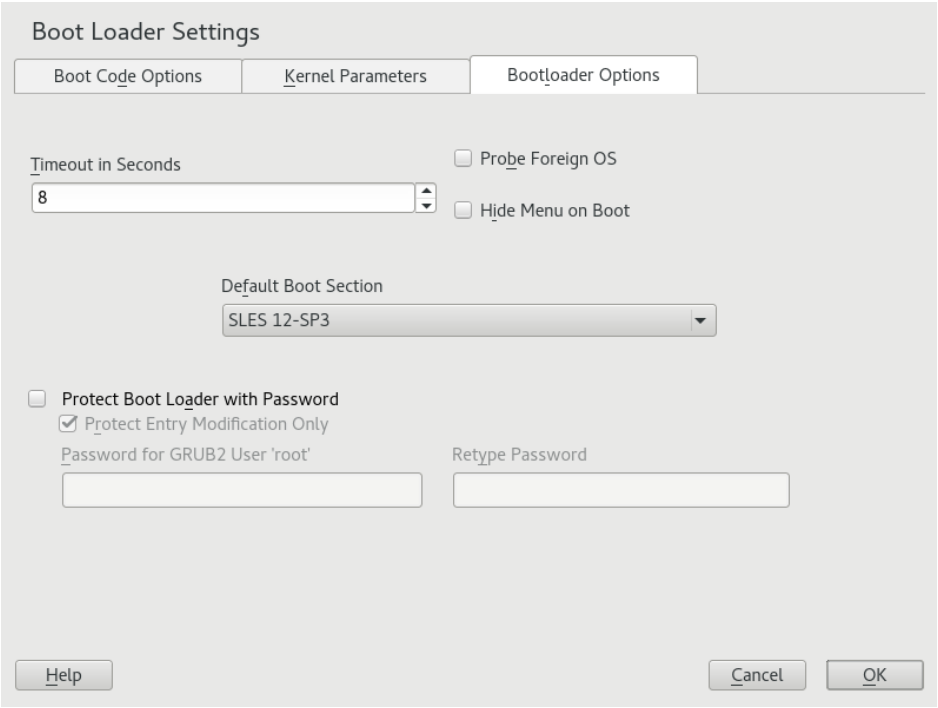


FIGURE 12.4: BOOT LOADER OPTIONS

Boot Loader Time-Out

Change the value of *Time-Out in Seconds* by typing in a new value and clicking the appropriate arrow key with your mouse.

Probe Foreign OS

When selected, the boot loader searches for other systems like Windows or other Linux installations.

Hide Menu on Boot

Hides the boot menu and boots the default entry.

Adjusting the Default Boot Entry

Select the desired entry from the “Default Boot Section” list. Note that the “>” sign in the boot entry name delimits the boot section and its subsection.

Protect Boot Loader with Password

Protects the boot loader and the system with an additional password. For more information, see *Section 12.2.6, “Setting a Boot Password”*.

12.3.3.2 *Kernel Parameters* Tab

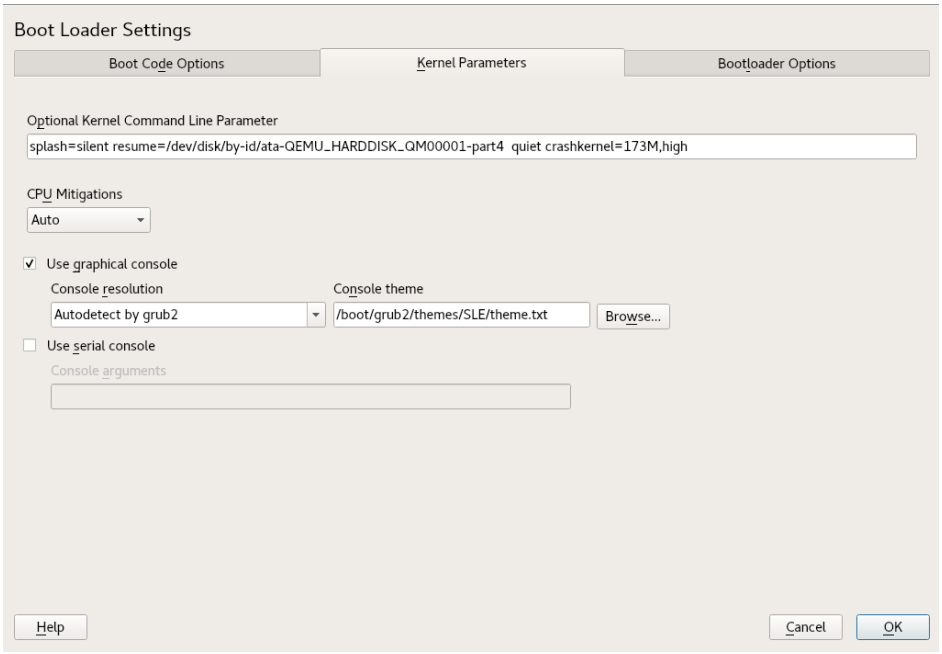


FIGURE 12.5: KERNEL PARAMETERS

Optional Kernel Command Line Parameter

Specify optional kernel parameters here to enable/disable system features, add drivers, etc.

CPU Mitigations

SUSE has released one or more kernel boot command line parameters for all software mitigations that have been deployed to prevent CPU side-channel attacks. Some of those may result in performance loss. Choose one the following options to strike a balance between security and performance, depending on your setting:

Auto. Enables all mitigations required for your CPU model, but does not protect against cross-CPU thread attacks. This setting may impact performance to some degree, depending on the workload.

Auto + No SMT. Provides the full set of available security mitigations. Enables all mitigations required for your CPU model. In addition, it disables Simultaneous Multithreading (SMT) to avoid side-channel attacks across multiple CPU threads. This setting may further impact performance, depending on the workload.

Off. Disables all mitigations. Side-channel attacks against your CPU are possible, depending on the CPU model. This setting has no impact on performance.

Manual. Does not set any mitigation level. Specify your CPU mitigations manually by using the kernel command line options.

Use Graphical Console

When checked, the boot menu appears on a graphical splash screen rather than in a text mode. The resolution of the boot screen is set automatically by default, but you can manually set it via *Console resolution*. The graphical theme definition file can be specified with the *Console theme* file-chooser. Only change this if you want to apply your own, custom-made theme.

Use Serial Console

If your machine is controlled via a serial console, activate this option and specify which COM port to use at which speed. See **`info grub`** or http://www.gnu.org/software/grub/manual/grub.html#Serial-terminal

12.4 Helpful GRUB 2 Commands

`grub2-mkconfig`

Generates a new `/boot/grub2/grub.cfg` based on `/etc/default/grub` and the scripts from `/etc/grub.d/`.

EXAMPLE 12.1: USAGE OF GRUB2-MKCONFIG

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

Tip: Syntax Check

Running **grub2-mkconfig** without any parameters prints the configuration to STDOUT where it can be reviewed. Use **grub2-script-check** after `/boot/grub2/grub.cfg` has been written to check its syntax.

Important: **grub2-mkconfig** Cannot Repair UEFI Secure Boot Tables

If you are using UEFI Secure Boot and your system is not reaching GRUB 2 correctly anymore, you may need to additionally reinstall Shim and regenerate the UEFI boot table. To do so, use:

```
root # shim-install --config-file=/boot/grub2/grub.cfg
```

grub2-mkrescue

Creates a bootable rescue image of your installed GRUB 2 configuration.

EXAMPLE 12.2: USAGE OF GRUB2-MKRESCUE

```
grub2-mkrescue -o save_path/name.iso iso
```

grub2-script-check

Checks the given file for syntax errors.

EXAMPLE 12.3: USAGE OF GRUB2-SCRIPT-CHECK

```
grub2-script-check /boot/grub2/grub.cfg
```

grub2-once

Set the default boot entry for the next boot only. To get the list of available boot entries use the `--list` option.

EXAMPLE 12.4: USAGE OF GRUB2-ONCE

```
grub2-once number_of_the_boot_entry
```

Tip: `grub2-once` Help

Call the program without any option to get a full list of all possible options.

12.5 More Information

Extensive information about GRUB 2 is available at http://www.gnu.org/software/grub/. Also refer to the `grub` info page.

13 Basic Networking

Linux offers the necessary networking tools and features for integration into all types of network structures. Network access using a network card can be configured with YaST. Manual configuration is also possible. In this chapter only the fundamental mechanisms and the relevant network configuration files are covered.

Linux and other Unix operating systems use the TCP/IP protocol. It is not a single network protocol, but a family of network protocols that offer various services. The protocols listed in *Several Protocols in the TCP/IP Protocol Family*, are provided for exchanging data between two machines via TCP/IP. Networks combined by TCP/IP, comprising a worldwide network, are also called "the Internet."

RFC stands for *Request for Comments*. RFCs are documents that describe various Internet protocols and implementation procedures for the operating system and its applications. The RFC documents describe the setup of Internet protocols. For more information about RFCs, see http://www.ietf.org/rfc.html.

SEVERAL PROTOCOLS IN THE TCP/IP PROTOCOL FAMILY

TCP

Transmission Control Protocol: a connection-oriented secure protocol. The data to transmit is first sent by the application as a stream of data and converted into the appropriate format by the operating system. The data arrives at the respective application on the destination host in the original data stream format it was initially sent. TCP determines whether any data has been lost or jumbled during the transmission. TCP is implemented wherever the data sequence matters.

UDP

User Datagram Protocol: a connectionless, insecure protocol. The data to transmit is sent in the form of packets generated by the application. The order in which the data arrives at the recipient is not guaranteed and data loss is possible. UDP is suitable for record-oriented applications. It features a smaller latency period than TCP.

ICMP

Internet Control Message Protocol: This is not a protocol for the end user, but a special control protocol that issues error reports and can control the behavior of machines participating in TCP/IP data transfer. In addition, it provides a special echo mode that can be viewed using the program ping.

IGMP

Internet Group Management Protocol: This protocol controls machine behavior when implementing IP multicast.

As shown in *Figure 13.1, "Simplified Layer Model for TCP/IP"*, data exchange takes place in different layers. The actual network layer is the insecure data transfer via IP (Internet protocol). On top of IP, TCP (transmission control protocol) guarantees, to a certain extent, security of the data transfer. The IP layer is supported by the underlying hardware-dependent protocol, such as Ethernet.

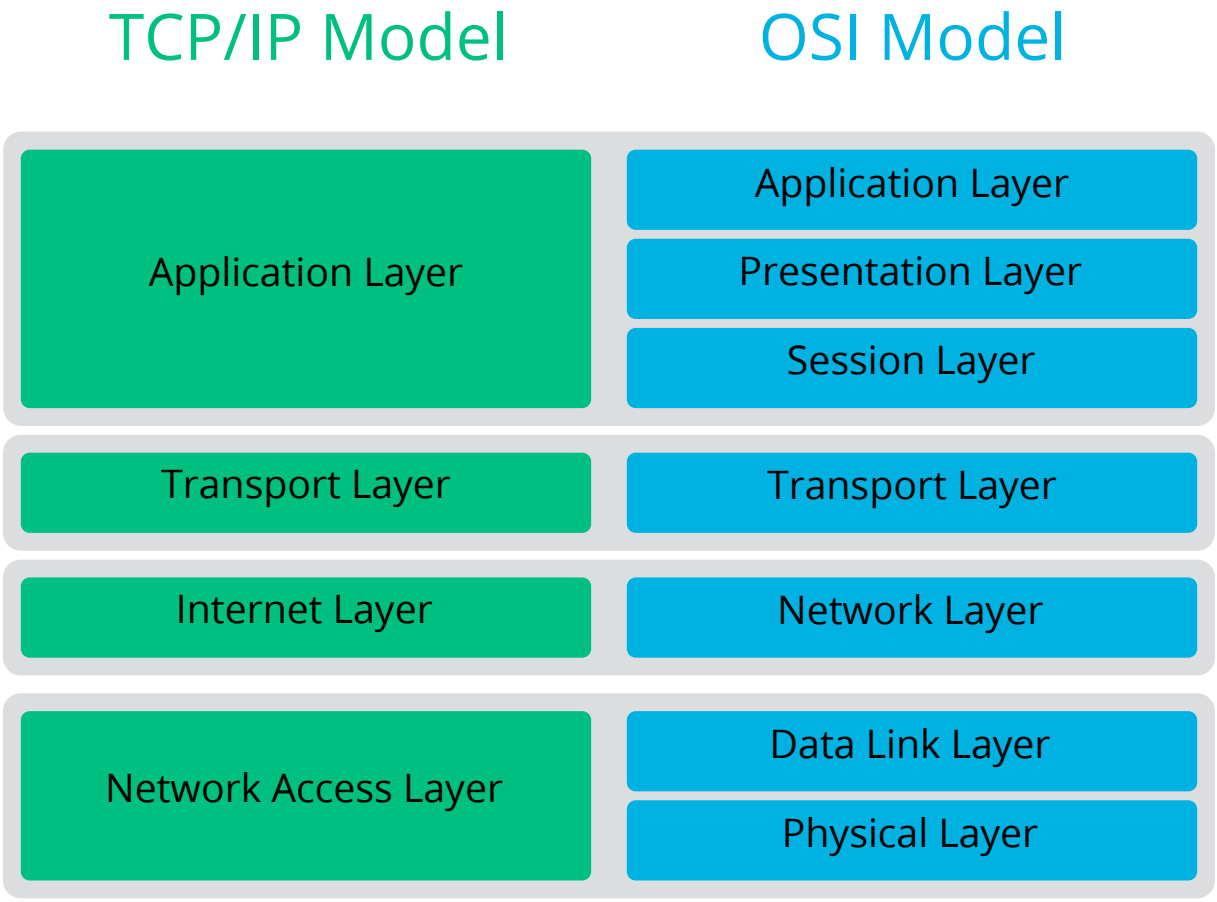


FIGURE 13.1: SIMPLIFIED LAYER MODEL FOR TCP/IP

The diagram provides one or two examples for each layer. The layers are ordered according to *abstraction levels*. The lowest layer is very close to the hardware. The uppermost layer, however, is almost a complete abstraction from the hardware. Every layer has its own special function. The special functions of each layer are mostly implicit in their description. The data link and physical layers represent the physical network used, such as Ethernet.

Almost all hardware protocols work on a packet-oriented basis. The data to transmit is collected into *packets* (it cannot be sent all at once). The maximum size of a TCP/IP packet is approximately 64 KB. Packets are normally quite smaller, as the network hardware can be a limiting factor. The maximum size of a data packet on an Ethernet is about fifteen hundred bytes. The size of a TCP/IP packet is limited to this amount when the data is sent over an Ethernet. If more data is transferred, more data packets need to be sent by the operating system.

For the layers to serve their designated functions, additional information regarding each layer must be saved in the data packet. This takes place in the *header* of the packet. Every layer attaches a small block of data, called the protocol header, to the front of each emerging packet. A sample TCP/IP data packet traveling over an Ethernet cable is illustrated in *Figure 13.2, "TCP/IP Ethernet Packet"*. The proof sum is located at the end of the packet, not at the beginning. This simplifies things for the network hardware.

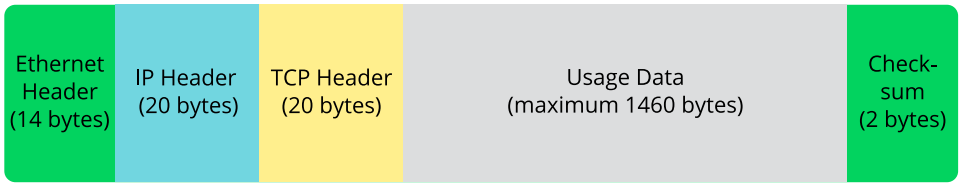


FIGURE 13.2: TCP/IP ETHERNET PACKET

When an application sends data over the network, the data passes through each layer, all implemented in the Linux kernel except the physical layer. Each layer is responsible for preparing the data so it can be passed to the next layer. The lowest layer is ultimately responsible for sending the data. The entire procedure is reversed when data is received. Like the layers of an onion, in each layer the protocol headers are removed from the transported data. Finally, the transport layer is responsible for making the data available for use by the applications at the destination. In this manner, one layer only communicates with the layer directly above or below it. For applications, it is irrelevant whether data is transmitted via a 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
`10.0.0.0/255.0.0.0`	`10.x.x.x`
`172.16.0.0/255.240.0.0`	`172.16.x.x – 172.31.x.x`
`192.168.0.0/255.255.0.0`	`192.168.x.x`

13.2 IPv6—The Next Generation Internet

Because of the emergence of the World Wide Web (WWW), the Internet has experienced explosive growth, with an increasing number of computers communicating via TCP/IP in the past fifteen years. Since Tim Berners-Lee at CERN (http://public.web.cern.ch) invented the WWW in 1990, the number of Internet hosts has grown from a few thousand to about a hundred million.

As mentioned, an IPv4 address consists of only 32 bits. Also, quite a few IP addresses are lost —they cannot be used because of the way in which networks are organized. The number of addresses available in your subnet is two to the power of the number of bits, minus two. A subnet has, for example, 2, 6, or 14 addresses available. To connect 128 hosts to the Internet, for example, you need a subnet with 256 IP addresses, from which only 254 are usable, because two IP addresses are needed for the structure of the subnet itself: the broadcast and the base network address.

Under the current IPv4 protocol, DHCP or NAT (network address translation) are the typical mechanisms used to circumvent the potential address shortage. Combined with the convention to keep private and public address spaces separate, these methods can certainly mitigate the shortage. The problem with them lies in their configuration, which is a chore to set up and a burden to maintain. To set up a host in an IPv4 network, you need several address items, such as the host's own IP address, the subnetmask, the gateway address and maybe a name server address. All these items need to be known and cannot be derived from somewhere else.

With IPv6, both the address shortage and the complicated configuration should be a thing of the past. The following sections tell more about the improvements and benefits brought by IPv6 and about the transition from the old protocol to the new one.

13.2.1 Advantages

The most important and most visible improvement brought by the new protocol is the enormous expansion of the available address space. An IPv6 address is made up of 128 bit values instead of the traditional 32 bits. This provides for as many as several quadrillion IP addresses.

However, IPv6 addresses are not only different from their predecessors with regard to their length. They also have a different internal structure that may contain more specific information about the systems and the networks to which they belong. More details about this are found in *Section 13.2.2, "Address Types and Structure"*.

The following is a list of other advantages of the new protocol:

Autoconfiguration

IPv6 makes the network "plug and play" capable, which means that a newly set up system integrates into the (local) network without any manual configuration. The new host uses its automatic configuration mechanism to derive its own address from the information made available by the neighboring routers, relying on a protocol called the *neighbor discovery* (ND) protocol. This method does not require any intervention on the administrator's part and there is no need to maintain a central server for address allocation—an additional advantage over IPv4, where automatic address allocation requires a DHCP server.

Nevertheless if a router is connected to a switch, the router should send periodic advertisements with flags telling the hosts of a network how they should interact with each other. For more information, see RFC 2462 and the `radvd.conf(5)` man page, and RFC 3315.

Mobility

IPv6 makes it possible to assign several addresses to one network interface at the same time. This allows users to access several networks easily, something that could be compared with the international roaming services offered by mobile phone companies. When you take your mobile phone abroad, the phone automatically logs in to a foreign service when it enters the corresponding area, so you can be reached under the same number everywhere and can place an outgoing call, as you would in your home area.

Secure Communication

With IPv4, network security is an add-on function. IPv6 includes IPsec as one of its core features, allowing systems to communicate over a secure tunnel to avoid eavesdropping by outsiders on the Internet.

Backward Compatibility

Realistically, it would be impossible to switch the entire Internet from IPv4 to IPv6 at one time. Therefore, it is crucial that both protocols can coexist not only on the Internet, but also on one system. This is ensured by compatible addresses (IPv4 addresses can easily be translated into IPv6 addresses) and by using several tunnels. See *Section 13.2.3, "Coexistence of IPv4 and IPv6"*. Also, systems can rely on a *dual stack IP* technique to support both protocols at the same time, meaning that they have two network stacks that are completely separate, such that there is no interference between the two protocol versions.

Custom Tailored Services through Multicasting

With IPv4, some services, such as SMB, need to broadcast their packets to all hosts in the local network. IPv6 allows a much more fine-grained approach by enabling servers to address hosts through *multicasting*, that is by addressing several hosts as parts of a group. This is different from addressing all hosts through *broadcasting* or each host individually through *unicasting*. Which hosts are addressed as a group may depend on the concrete application. There are some predefined groups to address all name servers (the *all name servers multicast group*), for example, or all routers (the *all routers multicast group*).

13.2.2 Address Types and Structure

As mentioned, the current IP protocol has two major limitations: there is an increasing shortage of IP addresses, and configuring the network and maintaining the routing tables is becoming a more complex and burdensome task. IPv6 solves the first problem by expanding the address space to 128 bits. The second one is mitigated by introducing a hierarchical address structure combined with sophisticated techniques to allocate network addresses, and *multihoming* (the ability to assign several addresses to one device, giving access to several networks).

When dealing with IPv6, it is useful to know about three different types of addresses:

Unicast

Addresses of this type are associated with exactly one network interface. Packets with such an address are delivered to only one destination. Accordingly, unicast addresses are used to transfer packets to individual hosts on the local network or the Internet.

Multicast

Addresses of this type relate to a group of network interfaces. Packets with such an address are delivered to all destinations that belong to the group. Multicast addresses are mainly used by certain network services to communicate with certain groups of hosts in a well-directed manner.

Anycast

Addresses of this type are related to a group of interfaces. Packets with such an address are delivered to the member of the group that is closest to the sender, according to the principles of the underlying routing protocol. Anycast addresses are used to make it easier for hosts to find out about servers offering certain services in the given network area. All servers of the same type have the same anycast address. Whenever a host requests a service, it receives a reply from the server with the closest location, as determined by the routing protocol. If this server should fail for some reason, the protocol automatically selects the second closest server, then the third one, and so forth.

An IPv6 address is made up of eight four-digit fields, each representing 16 bits, written in hexadecimal notation. They are separated by colons (`:`). Any leading zero bytes within a given field may be dropped, but zeros within the field or at its end may not. Another convention is that more than four consecutive zero bytes may be collapsed into a double colon. However, only one such `::` is allowed per address. This kind of shorthand notation is shown in *Example 13.3, "Sample IPv6 Address"*, where all three lines represent the same address.

EXAMPLE 13.3: SAMPLE IPV6 ADDRESS

```
fe80 : 0000 : 0000 : 0000 : 0000 : 10 : 1000 : 1a4
fe80 :    0 :    0 :    0 :    0 : 10 : 1000 : 1a4
fe80 :                            : 10 : 1000 : 1a4
```

Each part of an IPv6 address has a defined function. The first bytes form the prefix and specify the type of address. The center part is the network portion of the address, but it may be unused. The end of the address forms the host part. With IPv6, the netmask is defined by indicating the length of the prefix after a slash at the end of the address. An address, as shown in *Example 13.4, "IPv6 Address Specifying the Prefix Length"*, contains the information that the first 64 bits form the network part of the address and the last 64 form its host part. In other words, the `64` means that the netmask is filled with 64 1-bit values from the left. As with IPv4, the IP address is combined with AND with the values from the netmask to determine whether the host is located in the same subnet or in another one.

EXAMPLE 13.4: IPV6 ADDRESS SPECIFYING THE PREFIX LENGTH

```
fe80::10:1000:1a4/64
```

IPv6 knows about several predefined types of prefixes. Some are shown in *Various IPv6 Prefixes*.

VARIOUS IPV6 PREFIXES

`00`

IPv4 addresses and IPv4 over IPv6 compatibility addresses. These are used to maintain compatibility with IPv4. Their use still requires a router able to translate IPv6 packets into IPv4 packets. Several special addresses, such as the one for the loopback device, have this prefix as well.

`2` or `3` as the first digit

Aggregatable global unicast addresses. As is the case with IPv4, an interface can be assigned to form part of a certain subnet. Currently, there are the following address spaces: `2001::/16` (production quality address space) and `2002::/16` (6to4 address space).

`fe80::/10`

Link-local addresses. Addresses with this prefix should not be routed and should therefore only be reachable from within the same subnet.

`fec0::/10`

Site-local addresses. These may be routed, but only within the network of the organization to which they belong. In effect, they are the IPv6 equivalent of the current private network address space, such as `10.x.x.x`.

`ff`

These are multicast addresses.

A unicast address consists of three basic components:

Public Topology

The first part (which also contains one of the prefixes mentioned above) is used to route packets through the public Internet. It includes information about the company or institution that provides the Internet access.

Site Topology

The second part contains routing information about the subnet to which to deliver the packet.

Interface ID

The third part identifies the interface to which to deliver the packet. This also allows for the MAC to form part of the address. Given that the MAC is a globally unique, fixed identifier coded into the device by the hardware maker, the configuration procedure is substantially

simplified. In fact, the first 64 address bits are consolidated to form the `EUI-64` token, with the last 48 bits taken from the MAC, and the remaining 24 bits containing special information about the token type. This also makes it possible to assign an `EUI-64` token to interfaces that do not have a MAC, such as those based on PPP.

On top of this basic structure, IPv6 distinguishes between five different types of unicast addresses:

`::` (unspecified)
: This address is used by the host as its source address when the interface is initialized for the first time (at which point, the address cannot yet be determined by other means).

`::1` (loopback)
: The address of the loopback device.

IPv4 Compatible Addresses
: The IPv6 address is formed by the IPv4 address and a prefix consisting of 96 zero bits. This type of compatibility address is used for tunneling (see *Section 13.2.3, "Coexistence of IPv4 and IPv6"*) to allow IPv4 and IPv6 hosts to communicate with others operating in a pure IPv4 environment.

IPv4 Addresses Mapped to IPv6
: This type of address specifies a pure IPv4 address in IPv6 notation.

Local Addresses
: There are two address types for local use:

  link-local
  : This type of address can only be used in the local subnet. Packets with a source or target address of this type should not be routed to the Internet or other subnets. These addresses contain a special prefix (`fe80::/10`) and the interface ID of the network card, with the middle part consisting of zero bytes. Addresses of this type are used during automatic configuration to communicate with other hosts belonging to the same subnet.

  site-local
  : Packets with this type of address may be routed to other subnets, but not to the wider Internet—they must remain inside the organization's own network. Such addresses are used for intranets and are an equivalent of the private address space defined by IPv4. They contain a special prefix (`fec0::/10`), the interface ID, and a 16 bit field specifying the subnet ID. Again, the rest is filled with zero bytes.

As a completely new feature introduced with IPv6, each network interface normally gets several IP addresses, with the advantage that several networks can be accessed through the same interface. One of these networks can be configured completely automatically using the MAC and a known prefix with the result that all hosts on the local network can be reached when IPv6 is enabled (using the link-local address). With the MAC forming part of it, any IP address used in the world is unique. The only variable parts of the address are those specifying the *site topology* and the *public topology*, depending on the actual network in which the host is currently operating.

For a host to go back and forth between different networks, it needs at least two addresses. One of them, the *home address*, not only contains the interface ID but also an identifier of the home network to which it normally belongs (and the corresponding prefix). The home address is a static address and, as such, it does not normally change. Still, all packets destined to the mobile host can be delivered to it, regardless of whether it operates in the home network or somewhere outside. This is made possible by the completely new features introduced with IPv6, such as *stateless autoconfiguration* and *neighbor discovery*. In addition to its home address, a mobile host gets one or more additional addresses that belong to the foreign networks where it is roaming. These are called *care-of* addresses. The home network has a facility that forwards any packets destined to the host when it is roaming outside. In an IPv6 environment, this task is performed by the *home agent*, which takes all packets destined to the home address and relays them through a tunnel. On the other hand, those packets destined to the care-of address are directly transferred to the mobile host without any special detours.

13.2.3 Coexistence of IPv4 and IPv6

The migration of all hosts connected to the Internet from IPv4 to IPv6 is a gradual process. Both protocols will coexist for some time to come. The coexistence on one system is guaranteed where there is a *dual stack* implementation of both protocols. That still leaves the question of how an IPv6 enabled host should communicate with an IPv4 host and how IPv6 packets should be transported by the current networks, which are predominantly IPv4-based. The best solutions offer tunneling and compatibility addresses (see *Section 13.2.2, "Address Types and Structure"*).

IPv6 hosts that are more or less isolated in the (worldwide) IPv4 network can communicate through tunnels: IPv6 packets are encapsulated as IPv4 packets to move them across an IPv4 network. Such a connection between two IPv4 hosts is called a *tunnel*. To achieve this, packets must include the IPv6 destination address (or the corresponding prefix) and the IPv4 address of the remote host at the receiving end of the tunnel. A basic tunnel can be configured manually according to an agreement between the hosts' administrators. This is also called *static tunneling*.

However, the configuration and maintenance of static tunnels is often too labor-intensive to use them for daily communication needs. Therefore, IPv6 provides for three different methods of *dynamic tunneling*:

6over4

IPv6 packets are automatically encapsulated as IPv4 packets and sent over an IPv4 network capable of multicasting. IPv6 is tricked into seeing the whole network (Internet) as a huge local area network (LAN). This makes it possible to determine the receiving end of the IPv4 tunnel automatically. However, this method does not scale very well and is also hampered because IP multicasting is far from widespread on the Internet. Therefore, it only provides a solution for smaller corporate or institutional networks where multicasting can be enabled. The specifications for this method are laid down in RFC 2529.

6to4

With this method, IPv4 addresses are automatically generated from IPv6 addresses, enabling isolated IPv6 hosts to communicate over an IPv4 network. However, several problems have been reported regarding the communication between those isolated IPv6 hosts and the Internet. The method is described in RFC 3056.

IPv6 Tunnel Broker

This method relies on special servers that provide dedicated tunnels for IPv6 hosts. It is described in RFC 3053.

13.2.4 Configuring IPv6

To configure IPv6, you normally do not need to make any changes on the individual workstations. IPv6 is enabled by default. To disable or enable IPv6 on an installed system, use the YaST *Network Settings* module. On the *Global Options* tab, select or deselect the *Enable IPv6* option as necessary. To enable it temporarily until the next reboot, enter `modprobe -i ipv6` as `root`. It is impossible to unload the IPv6 module after it has been loaded.

Because of the autoconfiguration concept of IPv6, the network card is assigned an address in the *link-local* network. Normally, no routing table management takes place on a workstation. The network routers can be queried by the workstation, using the *router advertisement protocol*, for what prefix and gateways should be implemented. The radvd program can be used to set up an IPv6 router. This program informs the workstations which prefix to use for the IPv6 addresses and which routers. Alternatively, use zebra/quagga for automatic configuration of both addresses and routing.

For information about how to set up various types of tunnels using the `/etc/sysconfig/network` files, see the man page of `ifcfg-tunnel` (`man ifcfg-tunnel`).

13.2.5 For More Information

The above overview does not cover the topic of IPv6 comprehensively. For a more in-depth look at the new protocol, refer to the following online documentation and books:

http://www.ipv6.org/
: The starting point for everything about IPv6.

http://www.ipv6day.org
: All information needed to start your own IPv6 network.

http://www.ipv6-to-standard.org/
: The list of IPv6-enabled products.

http://www.bieringer.de/linux/IPv6/
: Here, find the Linux IPv6-HOWTO and many links related to the topic.

RFC 2460
: The fundamental RFC about IPv6.

IPv6 Essentials
: A book describing all the important aspects of the topic is *IPv6 Essentials* by Silvia Hagen (ISBN 0-596-00125-8).

13.3 Name Resolution

DNS assists in assigning an IP address to one or more names and assigning a name to an IP address. In Linux, this conversion is usually carried out by a special type of software known as bind. The machine that takes care of this conversion is called a *name server*. The names make up a hierarchical system in which each name component is separated by a period. The name hierarchy is, however, independent of the IP address hierarchy described above.

Consider a complete name, such as `jupiter.example.com`, written in the format `hostname.domain`. A full name, called a *fully qualified domain name* (FQDN), consists of a host name and a domain name (`example.com`). The latter also includes the *top level domain* or TLD (`com`).

TLD assignment has become quite confusing for historical reasons. Traditionally, three-letter domain names are used in the USA. In the rest of the world, the two-letter ISO national codes are the standard. In addition to that, longer TLDs were introduced in 2000 that represent certain spheres of activity (for example, `.info`, `.name`, `.museum`).

In the early days of the Internet (before 1990), the file `/etc/hosts` was used to store the names of all the machines represented over the Internet. This quickly proved to be impractical in the face of the rapidly growing number of computers connected to the Internet. For this reason, a decentralized database was developed to store the host names in a widely distributed manner. This database, similar to the name server, does not have the data pertaining to all hosts in the Internet readily available, but can dispatch requests to other name servers.

The top of the hierarchy is occupied by *root name servers*. These root name servers manage the top level domains and are run by the Network Information Center (NIC). Each root name server knows about the name servers responsible for a given top level domain. Information about top level domain NICs is available at http://www.internic.net.

DNS can do more than resolve host names. The name server also knows which host is receiving e-mails for an entire domain—the *mail exchanger (MX)*.

For your machine to resolve an IP address, it must know about at least one name server and its IP address. Easily specify such a name server using YaST. The configuration of name server access with openSUSE® Leap is described in *Section 13.4.1.4, "Configuring Host Name and DNS"*. Setting up your own name server is described in *Chapter 19, The Domain Name System*.

The protocol `whois` is closely related to DNS. With this program, quickly find out who is responsible for a given domain.

Note: MDNS and .local Domain Names

The `.local` top level domain is treated as link-local domain by the resolver. DNS requests are send as multicast DNS requests instead of normal DNS requests. If you already use the `.local` domain in your name server configuration, you must switch this option off in `/etc/host.conf`. For more information, see the `host.conf` manual page.

To switch off MDNS during installation, use `nomdns=1` as a boot parameter.

For more information on multicast DNS, see http://www.multicastdns.org.

13.4 Configuring a Network Connection with YaST

There are many supported networking types on Linux. Most of them use different device names and the configuration files are spread over several locations in the file system. For a detailed overview of the aspects of manual network configuration, see *Section 13.6, "Configuring a Network Connection Manually"*.

All network interfaces with link up (with a network cable connected) are automatically configured. Additional hardware can be configured any time on the installed system. The following sections describe the network configuration for all types of network connections supported by openSUSE Leap.

13.4.1 Configuring the Network Card with YaST

To configure your Ethernet or Wi-Fi/Bluetooth card in YaST, select *System* › *Network Settings*. After starting the module, YaST displays the *Network Settings* dialog with four tabs: *Global Options*, *Overview*, *Hostname/DNS* and *Routing*.

The *Global Options* tab allows you to set general networking options such as the network setup method, IPv6, and general DHCP options. For more information, see *Section 13.4.1.1, "Configuring Global Networking Options"*.

The *Overview* tab contains information about installed network interfaces and configurations. Any properly detected network card is listed with its name. You can manually configure new cards, remove or change their configuration in this dialog. To manually configure a card that was not automatically detected, see *Section 13.4.1.3, "Configuring an Undetected Network Card"*. To change the configuration of an already configured card, see *Section 13.4.1.2, "Changing the Configuration of a Network Card"*.

The *Hostname/DNS* tab allows to set the host name of the machine and name the servers to be used. For more information, see *Section 13.4.1.4, "Configuring Host Name and DNS"*.

The *Routing* tab is used for the configuration of routing. See *Section 13.4.1.5, "Configuring Routing"* for more information.

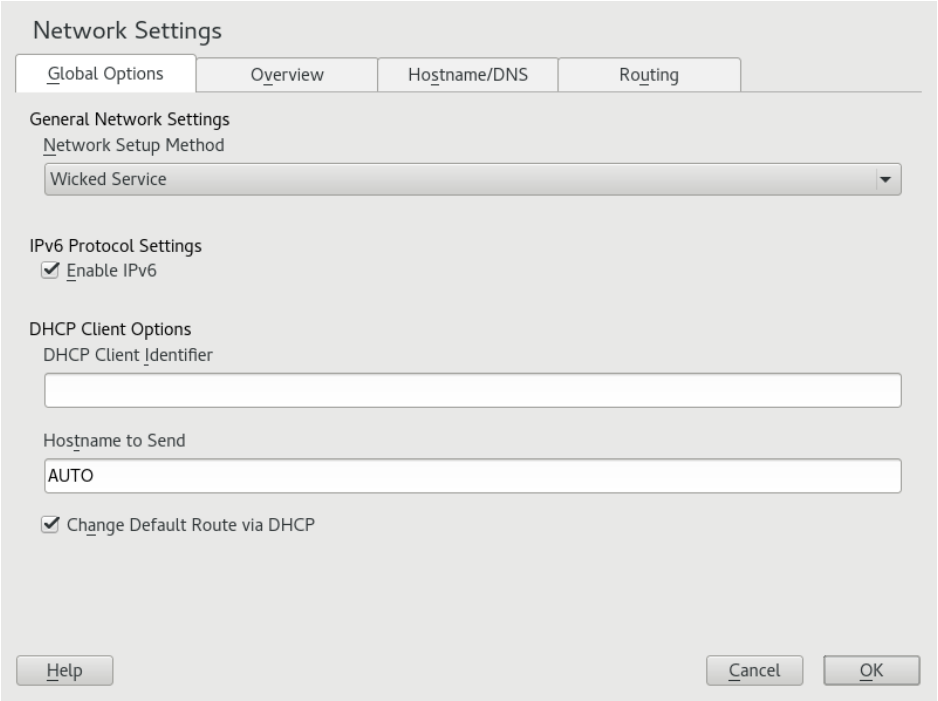


FIGURE 13.3: CONFIGURING NETWORK SETTINGS

13.4.1.1 Configuring Global Networking Options

The *Global Options* tab of the YaST *Network Settings* module allows you to set important global networking options, such as the use of NetworkManager, IPv6 and DHCP client options. These settings are applicable for all network interfaces.

In the *Network Setup Method* choose the way network connections are managed. If you want a NetworkManager desktop applet to manage connections for all interfaces, choose *NetworkManager Service*. NetworkManager is well suited for switching between multiple wired and wireless networks. If you do not run a desktop environment, or if your computer is a Xen server, virtual system, or provides network services such as DHCP or DNS in your network, use the *Wicked Service* method. If NetworkManager is used, **`nm-applet`** should be used to configure network options and the *Overview*, *Hostname/DNS* and *Routing* tabs of the *Network Settings* module are disabled. For more information on NetworkManager, see *Chapter 28, Using NetworkManager*.

In the *IPv6 Protocol Settings* choose whether to use the IPv6 protocol. It is possible to use IPv6 together with IPv4. By default, IPv6 is enabled. However, in networks not using IPv6 protocol, response times can be faster with IPv6 protocol disabled. To disable IPv6, deactivate *Enable IPv6*. If IPv6 is disabled, the kernel no longer loads the IPv6 module automatically. This setting will be applied after reboot.

In the *DHCP Client Options* configure options for the DHCP client. The *DHCP Client Identifier* must be different for each DHCP client on a single network. If left empty, it defaults to the hardware address of the network interface. However, if you are running several virtual machines using the same network interface and, therefore, the same hardware address, specify a unique free-form identifier here.

The *Hostname to Send* specifies a string used for the host name option field when the DHCP client sends messages to DHCP server. Some DHCP servers update name server zones (forward and reverse records) according to this host name (Dynamic DNS). Also, some DHCP servers require the *Hostname to Send* option field to contain a specific string in the DHCP messages from clients. Leave `AUTO` to send the current host name (that is the one defined in `/etc/HOSTNAME`). Make the option field empty for not sending any host name.

If you do not want to change the default route according to the information from DHCP, deactivate *Change Default Route via DHCP*.

13.4.1.2 Changing the Configuration of a Network Card

To change the configuration of a network card, select a card from the list of the detected cards in *Network Settings* › *Overview* in YaST and click *Edit*. The *Network Card Setup* dialog appears in which to adjust the card configuration using the *General*, *Address* and *Hardware* tabs.

13.4.1.2.1 Configuring IP Addresses

You can set the IP address of the network card or the way its IP address is determined in the *Address* tab of the *Network Card Setup* dialog. Both IPv4 and IPv6 addresses are supported. The network card can have *No IP Address* (which is useful for bonding devices), a *Statically Assigned IP Address* (IPv4 or IPv6) or a *Dynamic Address* assigned via *DHCP* or *Zeroconf* or both.

If using *Dynamic Address*, select whether to use *DHCP Version 4 Only* (for IPv4), *DHCP Version 6 Only* (for IPv6) or *DHCP Both Version 4 and 6*.

If possible, the first network card with link that is available during the installation is automatically configured to use automatic address setup via DHCP.

DHCP should also be used if you are using a DSL line but with no static IP assigned by the ISP (Internet Service Provider). If you decide to use DHCP, configure the details in *DHCP Client Options* in the *Global Options* tab of the *Network Settings* dialog of the YaST network card configuration module. If you have a virtual host setup where different hosts communicate through the same interface, an *DHCP Client Identifier* is necessary to distinguish them.

DHCP is a good choice for client configuration but it is not ideal for server configuration. To set a static IP address, proceed as follows:

1. Select a card from the list of detected cards in the *Overview* tab of the YaST network card configuration module and click *Edit*.
2. In the *Address* tab, choose *Statically Assigned IP Address*.
3. Enter the *IP Address*. Both IPv4 and IPv6 addresses can be used. Enter the network mask in *Subnet Mask*. If the IPv6 address is used, use *Subnet Mask* for prefix length in format `/64`. Optionally, you can enter a fully qualified *Hostname* for this address, which will be written to the `/etc/hosts` configuration file.
4. Click *Next*.
5. To activate the configuration, click *OK*.

Note: Interface Activation and Link Detection

During activation of a network interface, **`wicked`** checks for a carrier and only applies the IP configuration when a link has been detected. If you need to apply the configuration regardless of the link status (for example, when you want to test a service listening to a certain address), you can skip link detection by adding the variable `LINK_REQUIRED=no` to the configuration file of the interface in `/etc/sysconfig/network/ifcfg`.

Additionally, you can use the variable `LINK_READY_WAIT=5` to specify the timeout for waiting for a link in seconds.

For more information about the `ifcfg-*` configuration files, refer to *Section 13.6.2.5, "`/etc/sysconfig/network/ifcfg-*`"* and **`man 5 ifcfg`**.

If you use the static address, the name servers and default gateway are not configured automatically. To configure name servers, proceed as described in *Section 13.4.1.4, "Configuring Host Name and DNS"*. To configure a gateway, proceed as described in *Section 13.4.1.5, "Configuring Routing"*.

13.4.1.2.2 Configuring Multiple Addresses

One network device can have multiple IP addresses.

Note: Aliases Are a Compatibility Feature

These so-called aliases or labels, respectively, work with IPv4 only. With IPv6 they will be ignored. Using `iproute2` network interfaces can have one or more addresses.

Using YaST to set additional addresses for your network card, proceed as follows:

1. Select a card from the list of detected cards in the *Overview* tab of the YaST *Network Settings* dialog and click *Edit*.
2. In the *Address* › *Additional Addresses* tab, click *Add*.
3. Enter *IPv4 Address Label*, *IP Address*, and *Netmask*. Do not include the interface name in the alias name.
4. To activate the configuration, confirm the settings.

13.4.1.2.3 Changing the Device Name and Udev Rules

It is possible to change the device name of the network card when it is used. It is also possible to determine whether the network card should be identified by udev via its hardware (MAC) address or via the bus ID. The latter option is preferable in large servers to simplify hotplugging of cards. To set these options with YaST, proceed as follows:

1. Select a card from the list of detected cards in the *Overview* tab of the YaST *Network Settings* dialog and click *Edit*.
2. Go to the *Hardware* tab. The current device name is shown in *Udev Rules*. Click *Change*.
3. Select whether udev should identify the card by its *MAC Address* or *Bus ID*. The current MAC address and bus ID of the card are shown in the dialog.
4. To change the device name, check the *Change Device Name* option and edit the name.
5. To activate the configuration, confirm the settings.

13.4.1.2.4 Changing Network Card Kernel Driver

For some network cards, several kernel drivers may be available. If the card is already configured, YaST allows you to select a kernel driver to be used from a list of available suitable drivers. It is also possible to specify options for the kernel driver. To set these options with YaST, proceed as follows:

1. Select a card from the list of detected cards in the *Overview* tab of the YaST Network Settings module and click *Edit*.
2. Go to the *Hardware* tab.
3. Select the kernel driver to be used in *Module Name*. Enter any options for the selected driver in *Options* in the form **=** = *VALUE*. If more options are used, they should be space-separated.
4. To activate the configuration, confirm the settings.

13.4.1.2.5 Activating the Network Device

If you use the method with **wicked**, you can configure your device to either start during boot, on cable connection, on card detection, manually, or never. To change device start-up, proceed as follows:

1. In YaST select a card from the list of detected cards in *System* › *Network Settings* and click *Edit*.
2. In the *General* tab, select the desired entry from *Device Activation*.
   Choose *At Boot Time* to start the device during the system boot. With *On Cable Connection*, the interface is watched for any existing physical connection. With *On Hotplug*, the interface is set when available. It is similar to the *At Boot Time* option, and only differs in that no error occurs if the interface is not present at boot time. Choose *Manually* to control the interface manually with **ifup**. Choose *Never* to not start the device. The *On NFSroot* is similar to *At Boot Time*, but the interface does not shut down with the **systemctl stop network** command; the `network` service also cares about the `wicked` service if **wicked** is active. Use this if you use an NFS or iSCSI root file system.
3. To activate the configuration, confirm the settings.

Tip: NFS as a Root File System

On (diskless) systems where the root partition is mounted via network as an NFS share, you need to be careful when configuring the network device with which the NFS share is accessible.

When shutting down or rebooting the system, the default processing order is to turn off network connections, then unmount the root partition. With NFS root, this order causes problems as the root partition cannot be cleanly unmounted as the network connection to the NFS share is already not activated. To prevent the system from deactivating the relevant network device, open the network device configuration tab as described in *Section 13.4.1.2.5, "Activating the Network Device"* and choose *On NFSroot* in the *Device Activation* pane.

13.4.1.2.6 Setting Up Maximum Transfer Unit Size

You can set a maximum transmission unit (MTU) for the interface. MTU refers to the largest allowed packet size in bytes. A higher MTU brings higher bandwidth efficiency. However, large packets can block up a slow interface for some time, increasing the lag for further packets.

1. In YaST select a card from the list of detected cards in *System* › *Network Settings* and click *Edit*.
2. In the *General* tab, select the desired entry from the *Set MTU* list.
3. To activate the configuration, confirm the settings.

13.4.1.2.7 PCIe Multifunction Devices

Multifunction devices that support LAN, iSCSI, and FCoE are supported. The YaST FCoE client (`yast2 fcoe-client`) shows the private flags in additional columns to allow the user to select the device meant for FCoE. The YaST network module (`yast2 lan`) excludes "storage only devices" for network configuration.

13.4.1.2.8 Infiniband Configuration for IP-over-InfiniBand (IPoIB)

1. In YaST select the InfiniBand device in *System* › *Network Settings* and click *Edit*.

2. In the *General* tab, select one of the *IP-over-InfiniBand* (IPoIB) modes: *connected* (default) or *datagram*.

3. To activate the configuration, confirm the settings.

For more information about InfiniBand, see `/usr/src/linux/Documentation/infiniband/ipoib.txt`.

13.4.1.2.9 Configuring the Firewall

Without having to perform the detailed firewall setup as described in *Book "Security Guide", Chapter 16 "Masquerading and Firewalls", Section 16.4 "*`firewalld`*"*, you can determine the basic firewall configuration for your device as part of the device setup. Proceed as follows:

1. Open the YaST *System › Network Settings* module. In the *Overview* tab, select a card from the list of detected cards and click *Edit*.

2. Enter the *General* tab of the *Network Settings* dialog.

3. Determine the *Firewall Zone* to which your interface should be assigned. The following options are available:

   Firewall Disabled

   This option is available only if the firewall is disabled and the firewall does not run. Only use this option if your machine is part of a greater network that is protected by an outer firewall.

   Automatically Assign Zone

   This option is available only if the firewall is enabled. The firewall is running and the interface is automatically assigned to a firewall zone. The zone which contains the keyword `any` or the external zone will be used for such an interface.

   Internal Zone (Unprotected)

   The firewall is running, but does not enforce any rules to protect this interface. Use this option if your machine is part of a greater network that is protected by an outer firewall. It is also useful for the interfaces connected to the internal network, when the machine has more network interfaces.

   Demilitarized Zone

A demilitarized zone is an additional line of defense in front of an internal network and the (hostile) Internet. Hosts assigned to this zone can be reached from the internal network and from the Internet, but cannot access the internal network.

External Zone

The firewall is running on this interface and fully protects it against other—presumably hostile—network traffic. This is the default option.

4. To activate the configuration, confirm the settings.

13.4.1.3 Configuring an Undetected Network Card

If a network card is not detected correctly, the card is not included in the list of detected cards. If you are sure that your system includes a driver for your card, you can configure it manually. You can also configure special network device types, such as bridge, bond, TUN or TAP. To configure an undetected network card (or a special device) proceed as follows:

1. In the *System* › *Network Settings* › *Overview* dialog in YaST click *Add*.

2. In the *Hardware* dialog, set the *Device Type* of the interface from the available options and *Configuration Name*. If the network card is a USB device, activate the respective check box and exit this dialog with *Next*. Otherwise, you can define the kernel *Module Name* to be used for the card and its *Options*, if necessary.
   In *Ethtool Options*, you can set **`ethtool`** options used by **`ifup`** for the interface. For information about available options, see the **`ethtool`** manual page.
   If the option string starts with a `-` (for example, `-K INTERFACE_NAME rx on`), the second word in the string is replaced with the current interface name. Otherwise (for example, `autoneg off speed 10`) **`ifup`** adds `-s INTERFACE_NAME` to the beginning.

3. Click *Next*.

4. Configure any needed options, such as the IP address, device activation or firewall zone for the interface in the *General*, *Address*, and *Hardware* tabs. For more information about the configuration options, see *Section 13.4.1.2, “Changing the Configuration of a Network Card”*.

5. If you selected *Wireless* as the device type of the interface, configure the wireless connection in the next dialog.

6. To activate the new network configuration, confirm the settings.

13.4.1.4 Configuring Host Name and DNS

If you did not change the network configuration during installation and the Ethernet card was already available, a host name was automatically generated for your computer and DHCP was activated. The same applies to the name service information your host needs to integrate into a network environment. If DHCP is used for network address setup, the list of domain name servers is automatically filled with the appropriate data. If a static setup is preferred, set these values manually.

To change the name of your computer and adjust the name server search list, proceed as follows:

1. Go to the *Network Settings* › *Hostname/DNS* tab in the *System* module in YaST.

2. Enter the *Hostname* 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 `/run/netconfig/resolv.conf` file) is modified.
   If the *Use Default Policy* option is selected, the configuration is handled by the **`netconfig`** script which merges the data defined statically (with YaST or in the configuration files) with data obtained dynamically (from the DHCP client or NetworkManager). This default policy is usually sufficient.
   If the *Only Manually* option is selected, **`netconfig`** is not allowed to modify the `/run/netconfig/resolv.conf` file. However, this file can be edited manually.
   If the *Custom Policy* option is selected, a *Custom Policy Rule* string defining the merge policy should be specified. The string consists of a comma-separated list of interface names to be considered a valid source of settings. Except for complete interface names, basic wild cards

to match multiple interfaces are allowed, as well. For example, `eth* ppp?` will first target all eth and then all ppp0-ppp9 interfaces. There are two special policy values that indicate how to apply the static settings defined in the `/etc/sysconfig/network/config` file:

`STATIC`
: The static settings need to be merged together with the dynamic settings.

`STATIC_FALLBACK`
: The static settings are used only when no dynamic configuration is available.

For more information, see the man page of **`netconfig`**(8) (**`man 8 netconfig`**).

4. Enter the *Name Servers* and fill in the *Domain Search* list. Name servers must be specified by IP addresses, such as 192.168.1.116, not by host names. Names specified in the *Domain Search* tab are domain names used for resolving host names without a specified domain. If more than one *Domain Search* is used, separate domains with commas or white space.
5. To activate the configuration, confirm the settings.

It is also possible to edit the host name using YaST from the command line. The changes made by YaST take effect immediately (which is not the case when editing the `/etc/HOSTNAME` file manually). To change the host name, use the following command:

```
root # yast dns edit hostname=HOSTNAME
```

To change the name servers, use the following commands:

```
root # yast dns edit nameserver1=192.168.1.116
root # yast dns edit nameserver2=192.168.1.117
root # yast dns edit nameserver3=192.168.1.118
```

13.4.1.5 Configuring Routing

To make your machine communicate with other machines and other networks, routing information must be given to make network traffic take the correct path. If DHCP is used, this information is automatically provided. If a static setup is used, this data must be added manually.

1. In YaST go to *Network Settings* › *Routing*.
2. Enter the IP address of the *Default Gateway* (IPv4 and IPv6 if necessary). The default gateway matches every possible destination, but if a routing table entry exists that matches the required address, this will be used instead of the default route via the Default Gateway.

3. More entries can be entered in the *Routing Table*. Enter the *Destination* network IP address, *Gateway* IP address and the *Netmask*. Select the *Device* through which the traffic to the defined network will be routed (the minus sign stands for any device). To omit any of these values, use the minus sign `-`. To enter a default gateway into the table, use `default` in the *Destination* field.

   ### Note: Route Prioritization

   If more default routes are used, it is possible to specify the metric option to determine which route has a higher priority. To specify the metric option, enter `-metric` *`NUMBER`* in *Options*. The route with the highest metric is used as default. If the network device is disconnected, its route will be removed and the next one will be used. However, the current kernel does not use metric in static routing, only routing daemons like `multipathd` do.

4. If the system is a router, enable *IPv4 Forwarding* and *IPv6 Forwarding* in the *Network Settings* as needed.

5. To activate the configuration, confirm the settings.

13.5 NetworkManager

NetworkManager is the ideal solution for laptops and other portable computers. With NetworkManager, you do not need to worry about configuring network interfaces and switching between networks when you are moving.

13.5.1 NetworkManager and `wicked`

However, NetworkManager is not a suitable solution for all cases, so you can still choose between the **`wicked`** controlled method for managing network connections and NetworkManager. If you want to manage your network connection with NetworkManager, enable NetworkManager in the YaST Network Settings module as described in *Section 28.2, "Enabling or Disabling NetworkManager"* and configure your network connections with NetworkManager. For a list of use cases and a detailed description of how to configure and use NetworkManager, refer to *Chapter 28, Using NetworkManager*.

Some differences between wicked and NetworkManager:

`root` Privileges

If you use NetworkManager for network setup, you can easily switch, stop or start your network connection at any time from within your desktop environment using an applet. NetworkManager also makes it possible to change and configure wireless card connections without requiring `root` privileges. For this reason, NetworkManager is the ideal solution for a mobile workstation.

`wicked` also provides some ways to switch, stop or start the connection with or without user intervention, like user-managed devices. However, this always requires `root` privileges to change or configure a network device. This is often a problem for mobile computing, where it is not possible to preconfigure all the connection possibilities.

Types of Network Connections

Both **`wicked`** and NetworkManager can handle network connections with a wireless network (with WEP, WPA-PSK, and WPA-Enterprise access) and wired networks using DHCP and static configuration. They also support connection through dial-up and VPN. With NetworkManager you can also connect a mobile broadband (3G) modem or set up a DSL connection, which is not possible with the traditional configuration.

NetworkManager tries to keep your computer connected at all times using the best connection available. If the network cable is accidentally disconnected, it tries to reconnect. It can find the network with the best signal strength from the list of your wireless connections and automatically use it to connect. To get the same functionality with **`wicked`**, more configuration effort is required.

13.5.2 NetworkManager Functionality and Configuration Files

The individual network connection settings created with NetworkManager are stored in configuration profiles. The *system* connections configured with either NetworkManager or YaST are saved in `/etc/NetworkManager/system-connections/*` or in `/etc/sysconfig/network/ifcfg-*`. For GNOME, all user-defined connections are stored in GConf.

In case no profile is configured, NetworkManager automatically creates one and names it `Auto $INTERFACE-NAME`. That is made in an attempt to work without any configuration for as many cases as (securely) possible. If the automatically created profiles do not suit your needs, use the network connection configuration dialogs provided by GNOME to modify them as desired. For more information, see *Section 28.3, “Configuring Network Connections”*.

13.5.3 Controlling and Locking Down NetworkManager Features

On centrally administered machines, certain NetworkManager features can be controlled or disabled with PolKit, for example if a user is allowed to modify administrator defined connections or if a user is allowed to define their own network configurations. To view or change the respective NetworkManager policies, start the graphical *Authorizations* tool for PolKit. In the tree on the left side, find them below the *network-manager-settings* entry. For an introduction to PolKit and details on how to use it, refer to *Book "*Security 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. These scripts interact with each other in a way that is not well defined. This leads to unpredictable issues, obscure constraints and conventions, etc. Several layers of special hacks for a variety of different scenarios increase the maintenance burden. Address configuration protocols are being used that are implemented via daemons like dhcpcd, which interact rather poorly with the rest of the infrastructure. Funky interface naming schemes that require heavy udev support are introduced to achieve persistent identification of interfaces.

The idea of wicked is to decompose the problem in several ways. None of them is entirely novel, but trying to put ideas from different projects together is hopefully going to create a better solution overall.

One approach is to use a client/server model. This allows wicked to define standardized facilities for things like address configuration that are well integrated with the overall framework. For example, using a specific address configuration, the administrator may request that an interface

should be configured via DHCP or IPv4 zeroconf. In this case, the address configuration service simply obtains the lease from its server and passes it on to the wicked server process that installs the requested addresses and routes.

The other approach to decomposing the problem is to enforce the layering aspect. For any type of network interface, it is possible to define a dbus service that configures the network interface's device layer—a VLAN, a bridge, a bonding, or a paravirtualized device. Common functionality, such as address configuration, is implemented by joint services that are layered on top of these device specific services without having to implement them specifically.

The wicked framework implements these two aspects by using a variety of dbus services, which get attached to a network interface depending on its type. Here is a rough overview of the current object hierarchy in wicked.

Each network interface is represented via a child object of `/org/opensuse/Network/Interfaces`. The name of the child object is given by its ifindex. For example, the loopback interface, which usually gets ifindex 1, is `/org/opensuse/Network/Interfaces/1`, the first Ethernet interface registered is `/org/opensuse/Network/Interfaces/2`.

Each network interface has a “class” associated with it, which is used to select the dbus interfaces it supports. By default, each network interface is of class `netif`, and `wickedd` will automatically attach all interfaces compatible with this class. In the current implementation, this includes the following interfaces:

org.opensuse.Network.Interface
: Generic network interface functions, such as taking the link up or down, assigning an MTU, etc.

org.opensuse.Network.Addrconf.ipv4.dhcp,
org.opensuse.Network.Addrconf.ipv6.dhcp,
org.opensuse.Network.Addrconf.ipv4.auto
: Address configuration services for DHCP, IPv4 zeroconf, etc.

Beyond this, network interfaces may require or offer special configuration mechanisms. For an Ethernet device, for example, you should be able to control the link speed, offloading of checksumming, etc. To achieve this, Ethernet devices have a class of their own, called `netif-ethernet`, which is a subclass of `netif`. As a consequence, the dbus interfaces assigned to an Ethernet interface include all the services listed above, plus the `org.opensuse.Network.Ethernet` service available only to objects belonging to the `netif-ethernet` class.

Similarly, there exist classes for interface types like bridges, VLANs, bonds, or infinibands.

How do you interact with an interface like VLAN (which is really a virtual network interface that sits on top of an Ethernet device) that needs to be created first? For this, wicked defines factory interfaces, such as `org.opensuse.Network.VLAN.Factory`. Such a factory interface offers a single function that lets you create an interface of the requested type. These factory interfaces are attached to the `/org/opensuse/Network/Interfaces` list node.

13.6.1.1 `wicked` Architecture and Features

The `wicked` service comprises several parts as depicted in *Figure 13.4, "`wicked` architecture"*.

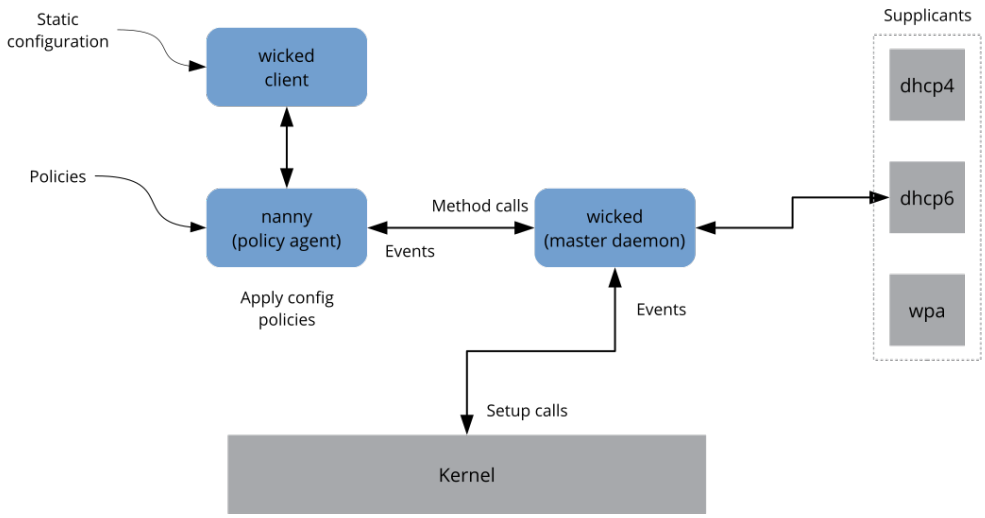


FIGURE 13.4: `wicked` ARCHITECTURE

`wicked` currently supports the following:

- Configuration file back-ends to parse SUSE style `/etc/sysconfig/network` files.
- An internal configuration back-end to represent network interface configuration in XML.
- Bring up and shutdown of "normal" network interfaces such as Ethernet or InfiniBand, VLAN, bridge, bonds, tun, tap, dummy, macvlan, macvtap, hsi, qeth, iucv, and wireless (currently limited to one wpa-psk/eap network) devices.
- A built-in DHCPv4 client and a built-in DHCPv6 client.

- The nanny daemon (enabled by default) helps to automatically bring up configured interfaces when the device is available (interface hotplugging) and set up the IP configuration when a link (carrier) is detected. See *Section 13.6.1.3, "Nanny"* for more information.
- `wicked` was implemented as a group of DBus services that are integrated with systemd. So the usual **`systemctl`** commands will apply to `wicked`.

13.6.1.2 Using `wicked`

On openSUSE Leap, `wicked` runs by default on desktop or server hardware. On mobile hardware NetworkManager runs by default. If you want to check what is currently enabled and whether it is running, call:

```
systemctl status network
```

If `wicked` is enabled, you will see something along these lines:

```
wicked.service - wicked managed network interfaces
    Loaded: loaded (/usr/lib/systemd/system/wicked.service; enabled)
    ...
```

In case something different is running (for example, NetworkManager) and you want to switch to `wicked`, first stop what is running and then enable `wicked`:

```
systemctl is-active network && \
systemctl stop      network
systemctl enable --force wicked
```

This enables the wicked services, creates the `network.service` to `wicked.service` alias link, and starts the network at the next boot.

Starting the server process:

```
systemctl start wickedd
```

This starts **`wickedd`** (the main server) and associated supplicants:

```
/usr/lib/wicked/bin/wickedd-auto4 --systemd --foreground
/usr/lib/wicked/bin/wickedd-dhcp4 --systemd --foreground
/usr/lib/wicked/bin/wickedd-dhcp6 --systemd --foreground
/usr/sbin/wickedd --systemd --foreground
/usr/sbin/wickedd-nanny --systemd --foreground
```

Then bringing up the network:

```
systemctl start wicked
```

Alternatively use the `network.service` alias:

```
systemctl start network
```

These commands are using the default or system configuration sources as defined in `/etc/wicked/client.xml`.

To enable debugging, set `WICKED_DEBUG` in `/etc/sysconfig/network/config`, for example:

```
WICKED_DEBUG="all"
```

Or, to omit some:

```
WICKED_DEBUG="all,-dbus,-objectmodel,-xpath,-xml"
```

Use the client utility to display interface information for all interfaces or the interface specified with *`IFNAME`*:

```
wicked show all
wicked show IFNAME
```

In XML output:

```
wicked show-xml all
wicked show-xml IFNAME
```

Bringing up one interface:

```
wicked ifup eth0
wicked ifup wlan0
...
```

Because there is no configuration source specified, the wicked client checks its default sources of configuration defined in `/etc/wicked/client.xml`:

1. `firmware:` iSCSI Boot Firmware Table (iBFT)
2. `compat:` `ifcfg` files—implemented for compatibility

Whatever `wicked` gets from those sources for a given interface is applied. The intended order of importance is `firmware`, then `compat`—this may be changed in the future.

For more information, see the **wicked** man page.

13.6.1.3 Nanny

Nanny is an event and policy driven daemon that is responsible for asynchronous or unsolicited scenarios such as hotplugging devices. Thus the nanny daemon helps with starting or restarting delayed or temporarily gone devices. Nanny monitors device and link changes, and integrates new devices defined by the current policy set. Nanny continues to set up even if **ifup** already exited because of specified timeout constraints.

By default, the nanny daemon is active on the system. It is enabled in the `/etc/wicked/common.xml` configuration file:

```
<config>
  ...
  <use-nanny>true</use-nanny>
</config>
```

This setting causes ifup and ifreload to apply a policy with the effective configuration to the nanny daemon; then, nanny configures `wickedd` and thus ensures hotplug support. It waits in the background for events or changes (such as new devices or carrier on).

13.6.1.4 Bringing Up Multiple Interfaces

For bonds and bridges, it may make sense to define the entire device topology in one file (ifcfg-bondX), and bring it up in one go. wicked then can bring up the whole configuration if you specify the top level interface names (of the bridge or bond):

```
wicked ifup br0
```

This command automatically sets up the bridge and its dependencies in the appropriate order without the need to list the dependencies (ports, etc.) separately.

To bring up multiple interfaces in one command:

```
wicked ifup bond0 br0 br1 br2
```

Or also all interfaces:

```
wicked ifup all
```

13.6.1.5 Using Tunnels with Wicked

When you need to use tunnels with Wicked, the `TUNNEL_DEVICE` is used for this. It permits to specify an optional device name to bind the tunnel to the device. The tunneled packets will only be routed via this device.

For more information, refer to **`man 5 ifcfg-tunnel`**.

13.6.1.6 Handling Incremental Changes

With **`wicked`**, there is no need to actually take down an interface to reconfigure it (unless it is required by the kernel). For example, to add another IP address or route to a statically configured network interface, add the IP address to the interface definition, and do another "ifup" operation. The server will try hard to update only those settings that have changed. This applies to link-level options such as the device MTU or the MAC address, and network-level settings, such as addresses, routes, or even the address configuration mode (for example, when moving from a static configuration to DHCP).

Things get tricky of course with virtual interfaces combining several real devices such as bridges or bonds. For bonded devices, it is not possible to change certain parameters while the device is up. Doing that will result in an error.

However, what should still work, is the act of adding or removing the child devices of a bond or bridge, or choosing a bond's primary interface.

13.6.1.7 Wicked Extensions: Address Configuration

`wicked` is designed to be extensible with shell scripts. These extensions can be defined in the `config.xml` file.

Currently, several classes of extensions are supported:

- link configuration: these are scripts responsible for setting up a device's link layer according to the configuration provided by the client, and for tearing it down again.
- address configuration: these are scripts responsible for managing a device's address configuration. Usually address configuration and DHCP are managed by **`wicked`** itself, but can be implemented by means of extensions.
- firewall extension: these scripts can apply firewall rules.

Typically, extensions have a start and a stop command, an optional “pid file”, and a set of environment variables that get passed to the script.

To illustrate how this is supposed to work, look at a firewall extension defined in `etc/server.xml`:

```
<dbus-service interface="org.opensuse.Network.Firewall">
 <action name="firewallUp"   command="/etc/wicked/extensions/firewall up"/>
 <action name="firewallDown" command="/etc/wicked/extensions/firewall down"/>

 <!-- default environment for all calls to this extension script -->
 <putenv name="WICKED_OBJECT_PATH" value="$object-path"/>
 <putenv name="WICKED_INTERFACE_NAME" value="$property:name"/>
 <putenv name="WICKED_INTERFACE_INDEX" value="$property:index"/>
</dbus-service>
```

The extension is attached to the `<dbus-service>` tag and defines commands to execute for the actions of this interface. Further, the declaration can define and initialize environment variables passed to the actions.

13.6.1.8 Wicked Extensions: Configuration Files

You can extend the handling of configuration files with scripts as well. For example, DNS updates from leases are ultimately handled by the `extensions/resolver` script, with behavior configured in `server.xml`:

```
<system-updater name="resolver">
 <action name="backup" command="/etc/wicked/extensions/resolver backup"/>
 <action name="restore" command="/etc/wicked/extensions/resolver restore"/>
 <action name="install" command="/etc/wicked/extensions/resolver install"/>
 <action name="remove" command="/etc/wicked/extensions/resolver remove"/>
</system-updater>
```

When an update arrives in `wickedd`, the system updater routines parse the lease and call the appropriate commands (`backup`, `install`, etc.) in the resolver script. This in turn configures the DNS settings using **`/sbin/netconfig`**, or by manually writing `/run/netconfig/resolv.conf` as a fallback.

13.6.2 Configuration Files

This section provides an overview of the network configuration files and explains their purpose and the format used.

13.6.2.1 /etc/wicked/common.xml

The /etc/wicked/common.xml file contains common definitions that should be used by all applications. It is sourced/included by the other configuration files in this directory. Although you can use this file to enable debugging across all wicked components, we recommend to use the file /etc/wicked/local.xml for this purpose. After applying maintenance updates you might lose your changes as the /etc/wicked/common.xml might be overwritten. The /etc/wicked/common.xml file includes the /etc/wicked/local.xml in the default installation, thus you typically do not need to modify the /etc/wicked/common.xml.

In case you want to disable nanny by setting the <use-nanny> to false, restart the wickedd.service and then run the following command to apply all configurations and policies:

```
tux > sudo wicked ifup all
```

Note: Configuration Files

The wickedd, wicked, or nanny programs try to read /etc/wicked/common.xml if their own configuration files do not exist.

13.6.2.2 /etc/wicked/server.xml

The file /etc/wicked/server.xml is read by the wickedd server process at start-up. The file stores extensions to the /etc/wicked/common.xml. On top of that this file configures handling of a resolver and receiving information from addrconf supplicants, for example DHCP.

We recommend to add changes required to this file into a separate file /etc/wicked/server-local.xml, that gets included by /etc/wicked/server.xml. By using a separate file you avoid overwriting of your changes during maintenance updates.

13.6.2.3 /etc/wicked/client.xml

The /etc/wicked/client.xml is used by the **wicked** command. The file specifies the location of a script used when discovering devices managed by ibft and configures locations of network interface configurations.

We recommend to add changes required to this file into a separate file /etc/wicked/client-local.xml, that gets included by /etc/wicked/server.xml. By using a separate file you avoid overwriting of your changes during maintenance updates.

13.6.2.4 /etc/wicked/nanny.xml

The /etc/wicked/nanny.xml configures types of link layers. We recommend to add specific configuration into a separate file: /etc/wicked/nanny-local.xml to avoid losing the changes during maintenance updates.

13.6.2.5 /etc/sysconfig/network/ifcfg-*

These files contain the traditional configurations for network interfaces. In openSUSE prior to Leap, this was the only supported format besides iBFT firmware.

Note: **wicked** and the ifcfg-* Files

wicked reads these files if you specify the compat: prefix. According to the openSUSE Leap default configuration in /etc/wicked/client.xml, **wicked** tries these files before the XML configuration files in /etc/wicked/ifconfig.

The --ifconfig switch is provided mostly for testing only. If specified, default configuration sources defined in /etc/wicked/ifconfig are not applied.

The ifcfg-* files include information such as the start mode and the IP address. Possible parameters are described in the manual page of ifup. Additionally, most variables from the dhcp and wireless files can be used in the ifcfg-* files if a general setting should be used for only one interface. However, most of the /etc/sysconfig/network/config variables are global and cannot be overridden in ifcfg-files. For example, NETCONFIG_* variables are global.

For configuring macvlan and macvtab interfaces, see the ifcfg-macvlan and ifcfg-macvtap man pages. For example, for a macvlan interface provide a ifcfg-macvlan0 with settings as follows:

```
STARTMODE='auto'
MACVLAN_DEVICE='eth0'
#MACVLAN_MODE='vepa'
#LLADDR=02:03:04:05:06:aa
```

For ifcfg.template, see *Section 13.6.2.6, "/etc/sysconfig/network/config, /etc/sysconfig/network/dhcp, and /etc/sysconfig/network/wireless"*.

13.6.2.6 `/etc/sysconfig/network/config`, `/etc/sysconfig/network/dhcp`, and `/etc/sysconfig/network/wireless`

The file `config` contains general settings for the behavior of **`ifup`**, **`ifdown`** and **`ifstatus`**. `dhcp` contains settings for DHCP and `wireless` for wireless LAN cards. The variables in all three configuration files are commented. Some variables from `/etc/sysconfig/network/config` can also be used in `ifcfg-*` files, where they are given a higher priority. The `/etc/sysconfig/network/ifcfg.template` file lists variables that can be specified in a per interface scope. However, most of the `/etc/sysconfig/network/config` variables are global and cannot be overridden in ifcfg-files. For example, `NETWORKMANAGER` or `NETCONFIG_*` variables are global.

Note: Using DHCPv6

In openSUSE prior to Leap, DHCPv6 used to work even on networks where IPv6 Router Advertisements (RAs) were not configured properly. Starting with openSUSE Leap, DHCPv6 will correctly require that at least one of the routers on the network sends out RAs that indicate that this network is managed by DHCPv6.

For networks where the router cannot be configured correctly, the `ifcfg` option allows the user to override this behavior by specifying `DHCLIENT6_MODE='managed'` in the `ifcfg` file. You can also activate this workaround with a boot parameter in the installation system:

```
ifcfg=eth0=dhcp6,DHCLIENT6_MODE=managed
```

13.6.2.7 `/etc/sysconfig/network/routes` and `/etc/sysconfig/network/ifroute-*`

The static routing of TCP/IP packets is determined by the `/etc/sysconfig/network/routes` and `/etc/sysconfig/network/ifroute-*` files. All the static routes required by the various system tasks can be specified in `/etc/sysconfig/network/routes`: routes to a host, routes to a host via a gateway and routes to a network. For each interface that needs individual routing, define an additional configuration file: `/etc/sysconfig/network/ifroute-*`. Replace the wild card (`*`) with the name of the interface. The entries in the routing configuration files look like this:

```
# Destination     Gateway           Netmask           Interface  Options
```

The route's destination is in the first column. This column may contain the IP address of a network or host or, in the case of *reachable* name servers, the fully qualified network or host name. The network should be written in CIDR notation (address with the associated routing prefix-length) such as 10.10.0.0/16 for IPv4 or fc00::/7 for IPv6 routes. The keyword `default` indicates that the route is the default gateway in the same address family as the gateway. For devices without a gateway use explicit 0.0.0.0/0 or ::/0 destinations.

The second column contains the default gateway or a gateway through which a host or network can be accessed.

The third column is deprecated; it used to contain the IPv4 netmask of the destination. For IPv6 routes, the default route, or when using a prefix-length (CIDR notation) in the first column, enter a dash (`-`) here.

The fourth column contains the name of the interface. If you leave it empty using a dash (`-`), it can cause unintended behavior in `/etc/sysconfig/network/routes`. For more information, see the `routes` man page.

An (optional) fifth column can be used to specify special options. For details, see the `routes` man page.

EXAMPLE 13.5: COMMON NETWORK INTERFACES AND SOME STATIC ROUTES

```
# --- IPv4 routes in CIDR prefix notation:
# Destination     [Gateway]         -                 Interface
127.0.0.0/8       -                 -                 lo
204.127.235.0/24  -                 -                 eth0
default           204.127.235.41    -                 eth0
207.68.156.51/32  207.68.145.45     -                 eth1
192.168.0.0/16    207.68.156.51     -                 eth1

# --- IPv4 routes in deprecated netmask notation"
# Destination     [Dummy/Gateway]   Netmask           Interface
#
127.0.0.0         0.0.0.0           255.255.255.0     lo
204.127.235.0     0.0.0.0           255.255.255.0     eth0
default           204.127.235.41    0.0.0.0           eth0
207.68.156.51     207.68.145.45     255.255.255.255   eth1
192.168.0.0       207.68.156.51     255.255.0.0       eth1

# --- IPv6 routes are always using CIDR notation:
# Destination     [Gateway]               -           Interface
2001:DB8:100::/64 -                       -           eth0
2001:DB8:100::/32 fe80::216:3eff:fe6d:c042 -          eth0
```

13.6.2.8 /var/run/netconfig/resolv.conf

The domain to which the host belongs is specified in `/var/run/netconfig/resolv.conf` (keyword `search`). Up to six domains with a total of 256 characters can be specified with the `search` option. When resolving a name that is not fully qualified, an attempt is made to generate one by attaching the individual `search` entries. Up to three name servers can be specified with the `nameserver` option, each on a line of its own. Comments are preceded by hash mark or semicolon signs (`#` or `;`). As an example, see *Example 13.6, "/var/run/netconfig/resolv.conf"*.

However, `/etc/resolv.conf` should not be edited by hand. It is generated by the **`netconfig`** script and is a symbolic link to `/run/netconfig/resolv.conf`. To define static DNS configuration without using YaST, edit the appropriate variables manually in the `/etc/sysconfig/network/config` file:

`NETCONFIG_DNS_STATIC_SEARCHLIST`
: list of DNS domain names used for host name lookup

`NETCONFIG_DNS_STATIC_SERVERS`
: list of name server IP addresses to use for host name lookup

`NETCONFIG_DNS_FORWARDER`
: the name of the DNS forwarder that needs to be configured, for example `bind` or `resolver`

`NETCONFIG_DNS_RESOLVER_OPTIONS`
: arbitrary options that will be written to `/var/run/netconfig/resolv.conf`, for example:

```
debug attempts:1 timeout:10
```

For more information, see the `resolv.conf` man page.

`NETCONFIG_DNS_RESOLVER_SORTLIST`
: list of up to 10 items, for example:

```
130.155.160.0/255.255.240.0 130.155.0.0
```

For more information, see the `resolv.conf` man page.

To disable DNS configuration using netconfig, set `NETCONFIG_DNS_POLICY=''`. For more information about **`netconfig`**, see the `netconfig(8)` man page (**`man 8 netconfig`**).

EXAMPLE 13.6: `/var/run/netconfig/resolv.conf`

```
# Our domain
search example.com
#
# We use dns.example.com (192.168.1.116) as nameserver
nameserver 192.168.1.116
```

13.6.2.9 `/sbin/netconfig`

`netconfig` is a modular tool to manage additional network configuration settings. It merges statically defined settings with settings provided by autoconfiguration mechanisms as DHCP or PPP according to a predefined policy. The required changes are applied to the system by calling the netconfig modules that are responsible for modifying a configuration file and restarting a service or a similar action.

`netconfig` recognizes three main actions. The **`netconfig modify`** and **`netconfig remove`** commands are used by daemons such as DHCP or PPP to provide or remove settings to netconfig. Only the **`netconfig update`** command is available for the user:

`modify`

The **`netconfig modify`** command modifies the current interface and service specific dynamic settings and updates the network configuration. Netconfig reads settings from standard input or from a file specified with the `--lease-file` *`FILENAME`* option and internally stores them until a system reboot (or the next modify or remove action). Already existing settings for the same interface and service combination are overwritten. The interface is specified by the `-i` *`INTERFACE_NAME`* parameter. The service is specified by the `-s` *`SERVICE_NAME`* parameter.

`remove`

The **`netconfig remove`** command removes the dynamic settings provided by 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 to update a specified service only (`dns`, `nis`, or `ntp`).

The netconfig policy and the static configuration settings are defined either manually or using YaST in the `/etc/sysconfig/network/config` file. The dynamic configuration settings provided by autoconfiguration tools such as DHCP or PPP are delivered directly by these tools with the **`netconfig modify`** and **`netconfig remove`** actions. When NetworkManager is enabled, netconfig (in policy mode `auto`) uses only NetworkManager settings, ignoring settings from any other interfaces configured using the traditional ifup method. If NetworkManager does not provide any setting, static settings are used as a fallback. A mixed usage of NetworkManager and the **`wicked`** method is not supported.

For more information about **`netconfig`**, see **`man 8 netconfig`**.

13.6.2.10 `/etc/hosts`

In this file, shown in *Example 13.7, "`/etc/hosts`"*, IP addresses are assigned to host names. If no name server is implemented, all hosts to which an IP connection will be set up must be listed here. For each host, enter a line consisting of the IP address, the fully qualified host name, and the host name into the file. The IP address must be at the beginning of the line and the entries separated by blanks and tabs. Comments are always preceded by the `#` sign.

EXAMPLE 13.7: `/etc/hosts`

```
127.0.0.1 localhost
192.168.2.100 jupiter.example.com jupiter
192.168.2.101 venus.example.com venus
```

13.6.2.11 `/etc/networks`

Here, network names are converted to network addresses. The format is similar to that of the `hosts` file, except the network names precede the addresses. See *Example 13.8, "`/etc/networks`"*.

EXAMPLE 13.8: `/etc/networks`

```
loopback    127.0.0.0
localnet    192.168.0.0
```

13.6.2.12 `/etc/host.conf`

Name resolution—the translation of host and network names via the *resolver* library—is controlled by this file. This file is only used for programs linked to libc4 or libc5. For current glibc programs, refer to the settings in `/etc/nsswitch.conf`. Each parameter must always be

entered on a separate line. Comments are preceded by a `#` sign. *Table 13.2, "Parameters for /etc/host.conf"* shows the parameters available. A sample `/etc/host.conf` is shown in *Example 13.9, "/etc/host.conf"*.

TABLE 13.2: PARAMETERS FOR /ETC/HOST.CONF

order *hosts, bind*	Specifies in which order the services are accessed for the name resolution. Available arguments are (separated by blank spaces or commas):
	hosts: searches the `/etc/hosts` file
	bind: accesses a name server
	nis: uses NIS
multi *on/off*	Defines if a host entered in `/etc/hosts` can have multiple IP addresses.
nospoof *on* spoofalert *on/off*	These parameters influence the name server *spoofing* but do not exert any influence on the network configuration.
trim *domainname*	The specified domain name is separated from the host name after host name resolution (as long as the host name includes the domain name). This option is useful only if names from the local domain are in the `/etc/hosts` file, but should still be recognized with the attached domain names.

EXAMPLE 13.9: `/etc/host.conf`

```
# We have named running
order hosts bind
# Allow multiple address
multi on
```

13.6.2.13 /etc/nsswitch.conf

The introduction of the GNU C Library 2.0 was accompanied by the introduction of the *Name Service Switch* (NSS). Refer to the `nsswitch.conf(5)` man page and *The GNU C Library Reference Manual* for details.

The order for queries is defined in the file `/etc/nsswitch.conf`. A sample `nsswitch.conf` is shown in *Example 13.10, "/etc/nsswitch.conf"*. Comments are preceded by `#` signs. In this example, the entry under the `hosts` database means that a request is sent to `/etc/hosts` (`files`) via DNS (see *Chapter 19, The Domain Name System*).

EXAMPLE 13.10: `/etc/nsswitch.conf`

```
passwd:     compat
group:      compat

hosts:      files dns
networks:   files dns

services:   db files
protocols:  db files
rpc:        files
ethers:     files
netmasks:   files
netgroup:   files nis
publickey:  files

bootparams: files
automount:  files nis
aliases:    files nis
shadow:     compat
```

The "databases" available over NSS are listed in *Table 13.3, "Databases Available via /etc/nsswitch.conf"*. The configuration options for NSS databases are listed in *Table 13.4, "Configuration Options for NSS "Databases""*.

TABLE 13.3: DATABASES AVAILABLE VIA /ETC/NSSWITCH.CONF

`aliases`	Mail aliases implemented by `sendmail`; see `man 5 aliases`.
`ethers`	Ethernet addresses.
`netmasks`	List of networks and their subnet masks. Only needed, if you use subnetting.

`group`	User groups used by `getgrent`. See also the man page for **`group`**.
`hosts`	Host names and IP addresses, used by `gethostbyname` and similar functions.
`netgroup`	Valid host and user lists in the network for controlling access permissions; see the `netgroup(5)` man page.
`networks`	Network names and addresses, used by `getnetent`.
`publickey`	Public and secret keys for Secure_RPC used by NFS and NIS+.
`passwd`	User passwords, used by `getpwent`; see the `passwd(5)` man page.
`protocols`	Network protocols, used by `getprotoent`; see the `protocols(5)` man page.
`rpc`	Remote procedure call names and addresses, used by `getrpcbyname` and similar functions.
`services`	Network services, used by `getservent`.
`shadow`	Shadow passwords of users, used by `getspnam`; see the `shadow(5)` man page.

TABLE 13.4: CONFIGURATION OPTIONS FOR NSS "DATABASES"

`files`	directly access files, for example, `/etc/aliases`
`db`	access via a database
`nis`, `nisplus`	NIS, see also *Book "*Security Guide*", Chapter 3 "Using NIS"*

`dns`	can only be used as an extension for `hosts` and `networks`
`compat`	can only be used as an extension for `passwd`, `shadow` and `group`

13.6.2.14 `/etc/nscd.conf`

This file is used to configure nscd (name service cache daemon). See the `nscd(8)` and `nscd.conf(5)` man pages. By default, the system entries of `passwd`, `groups` and `hosts` are cached by nscd. This is important for the performance of directory services, like NIS and LDAP, because otherwise the network connection needs to be used for every access to names, groups or hosts.

If the caching for `passwd` is activated, it usually takes about fifteen seconds until a newly added local user is recognized. Reduce this waiting time by restarting nscd with:

```
tux > sudo systemctl restart nscd
```

13.6.2.15 `/etc/HOSTNAME`

`/etc/HOSTNAME` contains the fully qualified host name (FQHN). The fully qualified host name is the host name with the domain name attached. This file must contain only one line (in which the host name is set). It is read while the machine is booting.

13.6.3 Testing the Configuration

Before you write your configuration to the configuration files, you can test it. To set up a test configuration, use the **`ip`** command. To test the connection, use the **`ping`** command.

The command **`ip`** changes the network configuration directly without saving it in the configuration file. Unless you enter your configuration in the correct configuration files, the changed network configuration is lost on reboot.

Note: **ifconfig** and **route** Are Obsolete

The **ifconfig** and **route** tools are obsolete. Use **ip** instead. **ifconfig**, for example, limits interface names to 9 characters.

13.6.3.1 Configuring a Network Interface with **ip**

ip is a tool to show and configure network devices, routing, policy routing, and tunnels.

ip is a very complex tool. Its common syntax is **ip** *OPTIONS OBJECT COMMAND*. You can work with the following objects:

link
: This object represents a network device.

address
: This object represents the IP address of device.

neighbor
: This object represents an ARP or NDISC cache entry.

route
: This object represents the routing table entry.

rule
: This object represents a rule in the routing policy database.

maddress
: This object represents a multicast address.

mroute
: This object represents a multicast routing cache entry.

tunnel
: This object represents a tunnel over IP.

If no command is given, the default command is used (usually **list**).

Change the state of a device with the command **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-cref.pdf.

13.6.3.2 Testing a Connection with ping

The **ping** command is the standard tool for testing whether a TCP/IP connection works. It uses the ICMP protocol to send a small data packet, ECHO_REQUEST datagram, to the destination host, requesting an immediate reply. If this works, **ping** displays a message to that effect. This indicates that the network link is functioning.

ping does more than only test the function of the connection between two computers: it also provides some basic information about the quality of the connection. In *Example 13.11, "Output of the Command ping"*, you can see an example of the **ping** output. The second-to-last line contains information about the number of transmitted packets, packet loss, and total time of **ping** running.

As the destination, you can use a host name or IP address, for example, **ping** example.com or **ping** 192.168.3.100. The program sends packets until you press Ctrl–C.

If you only need to check the functionality of the connection, you can limit the number of the packets with the -c option. For example to limit ping to three packets, enter **ping** -c 3 example.com.

EXAMPLE 13.11: OUTPUT OF THE COMMAND PING

```
ping -c 3 example.com
```

```
PING example.com (192.168.3.100) 56(84) bytes of data.
64 bytes from example.com (192.168.3.100): icmp_seq=1 ttl=49 time=188 ms
64 bytes from example.com (192.168.3.100): icmp_seq=2 ttl=49 time=184 ms
64 bytes from example.com (192.168.3.100): icmp_seq=3 ttl=49 time=183 ms
--- example.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 183.417/185.447/188.259/2.052 ms
```

The default interval between two packets is one second. To change the interval, ping provides the option `-i`. For example, to increase the ping interval to ten seconds, enter **`ping`** `-i 10 example.com`.

In a system with multiple network devices, it is sometimes useful to send the ping through a specific interface address. To do so, use the `-I` option with the name of the selected device, for example, **`ping`** `-I wlan1 example.com`.

For more options and information about using ping, enter **`ping`** `-h` or see the `ping (8)` man page.

Tip: Pinging IPv6 Addresses

For IPv6 addresses use the **`ping6`** command. Note, to ping link-local addresses, you must specify the interface with `-I`. The following command works, if the address is reachable via `eth1`:

```
ping6 -I eth1 fe80::117:21ff:feda:a425
```

13.6.4 Unit Files and Start-Up Scripts

Apart from the configuration files described above, there are also systemd unit files and various scripts that load the network services while the machine is booting. These are started when the system is switched to the `multi-user.target` target. Some of these unit files and scripts are described in *Some Unit Files and Start-Up Scripts for Network Programs*. For more information about `systemd`, see *Chapter 10, The `systemd` Daemon* and for more information about the `systemd` targets, see the man page of `systemd.special` (**`man systemd.special`**).

SOME UNIT FILES AND START-UP SCRIPTS FOR NETWORK PROGRAMS

`network.target`

`network.target` is the systemd target for networking, but its mean depends on the settings provided by the system administrator.

For more information, see http://www.freedesktop.org/wiki/Software/systemd/NetworkTarget/.

`multi-user.target`

`multi-user.target` is the systemd target for a multiuser system with all required network services.

`rpcbind`

Starts the rpcbind utility that converts RPC program numbers to universal addresses. It is needed for RPC services, such as an NFS server.

`ypserv`

Starts the NIS server.

`ypbind`

Starts the NIS client.

`/etc/init.d/nfsserver`

Starts the NFS server.

`/etc/init.d/postfix`

Controls the postfix process.

13.7 Basic Router Setup

A router is a networking device that delivers and receives data (network packets) to or from more than one network back and forth. You often use a router to connect your local network to the remote network (Internet) or to connect local network segments. With openSUSE Leap you can build a router with features such as NAT (Network Address Translation) or advanced firewalling.

The following are basic steps to turn openSUSE Leap into a router.

1. Enable forwarding, for example in `/etc/sysctl.d/50-router.conf`

```
net.ipv4.conf.all.forwarding = 1
net.ipv6.conf.all.forwarding = 1
```

Then provide a static IPv4 and IPv6 IP setup for the interfaces. Enabling forwarding disables several mechanisms, for example IPv6 does not accept an IPv6 RA (router advertisement) anymore, which also prevents the creation of a default route.

2. In many situations (for example, when you can reach the same network via more than one interface, or when VPN usually is used and already on "normal multi-home hosts"), you must disable the IPv4 reverse path filter (this feature does not currently exist for IPv6):

   ```
   net.ipv4.conf.all.rp_filter = 0
   ```

   You can also filter with firewall settings instead.

3. To accept an IPv6 RA (from the router on an external, uplink, or ISP interface) and create a default (or also a more specific) IPv6 route again, set:

   ```
   net.ipv6.conf.${ifname}.accept_ra = 2
   net.ipv6.conf.${ifname}.autoconf = 0
   ```

   (Note: "`eth0.42`" needs to be written as `eth0/42` in a dot-separated sysfs path.)

More router behavior and forwarding dependencies are described in https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt.

To provide IPv6 on your internal (DMZ) interfaces, and announce yourself as an IPv6 router and "autoconf networks" to the clients, install and configure `radvd` in `/etc/radvd.conf`, for example:

```
interface eth0
{
    IgnoreIfMissing on;         # do not fail if interface missed

    AdvSendAdvert on;           # enable sending RAs
    AdvManagedFlag on;          # IPv6 addresses managed via DHCPv6
    AdvOtherConfigFlag on;      # DNS, NTP... only via DHCPv6

    AdvDefaultLifetime 3600;    # client default route lifetime of 1 hour

    prefix 2001:db8:0:1::/64    # (/64 is default and required for autoconf)
    {
        AdvAutonomous off;         # Disable address autoconf (DHCPv6 only)

        AdvValidLifetime 3600;     # prefix (autoconf addr) is valid 1 h
        AdvPreferredLifetime 1800; # prefix (autoconf addr) is prefered 1/2 h
    }
}
```

Configure the firewall to masquerade traffic with NAT from the LAN into the WAN and to block inbound traffic on the WAN interface:

```
tux > sudo firewall-cmd --permanent --zone=external --change-interface=WAN_INTERFACE
tux > sudo firewall-cmd --permanent --zone=external --add-masquerade
tux > sudo firewall-cmd --permanent --zone=internal --change-interface=LAN_INTERFACE
tux > sudo firewall-cmd --reload
```

13.8 Setting Up Bonding Devices

For some systems, there is a desire to implement network connections that comply to more than the standard data security or availability requirements of a typical Ethernet device. In these cases, several Ethernet devices can be aggregated to a single bonding device.

The configuration of the bonding device is done by means of bonding module options. The behavior is mainly affected by the mode of the bonding device. By default, this is `active-backup` which means that a different slave device will become active if the active slave fails. The following bonding modes are available:

***0* (balance-rr)**

Packets are transmitted in round-robin fashion from the first to the last available interface. Provides fault tolerance and load balancing.

***1* (active-backup)**

Only one network interface is active. If it fails, a different interface becomes active. This setting is the default for openSUSE Leap. Provides fault tolerance.

***2* (balance-xor)**

Traffic is split between all available interfaces based on the following policy: `[(source MAC address XOR'd with destination MAC address XOR packet type ID) modulo slave count]` Requires support from the switch. Provides fault tolerance and load balancing.

***3* (broadcast)**

All traffic is broadcast on all interfaces. Requires support from the switch. Provides fault tolerance.

***4* (802.3ad)**

Aggregates interfaces into groups that share the same speed and duplex settings. Requires **`ethtool`** support in the interface drivers, and a switch that supports and is configured for IEEE 802.3ad Dynamic link aggregation. Provides fault tolerance and load balancing.

5 (balance-tlb)

Adaptive transmit load balancing. Requires **`ethtool`** support in the interface drivers but not switch support. Provides fault tolerance and load balancing.

6 (balance-alb)

Adaptive load balancing. Requires **`ethtool`** support in the interface drivers but not switch support. Provides fault tolerance and load balancing.

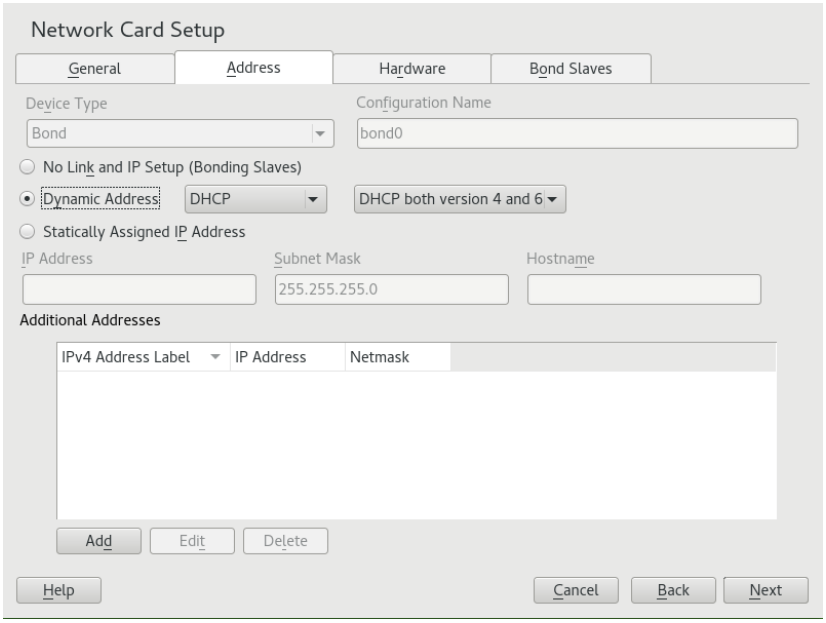
For a more detailed description of the modes, see https://www.kernel.org/doc/Documentation/networking/bonding.txt.

Tip: Bonding and Xen

Using bonding devices is only of interest for machines where you have multiple real network cards available. In most configurations, this means that you should use the bonding configuration only in Dom0. Only if you have multiple network cards assigned to a VM Guest system it may also be useful to set up the bond in a VM Guest.

To configure a bonding device, use the following procedure:

1. Run *YaST* › *System* › *Network Settings*.
2. Use *Add* and change the *Device Type* to *Bond*. Proceed with *Next*.



3. Select how to assign the IP address to the bonding device. Three methods are at your disposal:

   - No IP Address
   - Dynamic Address (with DHCP or Zeroconf)
   - Statically assigned IP Address

   Use the method that is appropriate for your environment.

4. In the *Bond Slaves* tab, select the Ethernet devices that should be included into the bond by activating the related check box.

5. Edit the *Bond Driver Options* and choose a bonding mode.

6. Make sure that the parameter `miimon=100` is added to the *Bond Driver Options*. Without this parameter, the data integrity is not checked regularly.

7. Click *Next* and leave YaST with *OK* to create the device.

13.8.1 Hotplugging of Bonding Slaves

In specific network environments (such as High Availability), there are cases when you need to replace a bonding slave interface with another one. The reason may be a constantly failing network device. The solution is to set up hotplugging of bonding slaves.

The bond is configured as usual (according to **man 5 ifcfg-bonding**), for example:

```
ifcfg-bond0
          STARTMODE='auto' # or 'onboot'
          BOOTPROTO='static'
          IPADDR='192.168.0.1/24'
          BONDING_MASTER='yes'
          BONDING_SLAVE_0='eth0'
          BONDING_SLAVE_1='eth1'
          BONDING_MODULE_OPTS='mode=active-backup miimon=100'
```

The slaves are specified with `STARTMODE=hotplug` and `BOOTPROTO=none`:

```
ifcfg-eth0
          STARTMODE='hotplug'
          BOOTPROTO='none'

ifcfg-eth1
          STARTMODE='hotplug'
          BOOTPROTO='none'
```

`BOOTPROTO=none` uses the **ethtool** options (when provided), but does not set the link up on **ifup eth0**. The reason is that the slave interface is controlled by the bond master.

`STARTMODE=hotplug` causes the slave interface to join the bond automatically when it is available.

The `udev` rules in `/etc/udev/rules.d/70-persistent-net.rules` need to be changed to match the device by bus ID (udev `KERNELS` keyword equal to "SysFS BusID" as visible in **hwinfo --netcard**) instead of by MAC address. This allows replacement of defective hardware (a network card in the same slot but with a different MAC) and prevents confusion when the bond changes the MAC address of all its slaves.

For example:

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
KERNELS=="0000:00:19.0", ATTR{dev_id}=="0x0", ATTR{type}=="1",
KERNEL=="eth*", NAME="eth0"
```

At boot time, the systemd `network.service` does not wait for the hotplug slaves, but for the bond to become ready, which requires at least one available slave. When one of the slave interfaces gets removed (unbind from NIC driver, **`rmmod`** of the NIC driver or true PCI hotplug remove) from the system, the kernel removes it from the bond automatically. When a new card is added to the system (replacement of the hardware in the slot), `udev` renames it using the bus-based persistent name rule to the name of the slave, and calls **`ifup`** for it. The **`ifup`** call automatically joins it into the bond.

13.9 Setting Up Team Devices for Network Teaming

The term "link aggregation" is the general term which describes combining (or aggregating) a network connection to provide a logical layer. Sometimes you find the terms "channel teaming", "Ethernet bonding", "port truncating", etc. which are synonyms and refer to the same concept.

This concept is widely known as "bonding" and was originally integrated into the Linux kernel (see *Section 13.8, "Setting Up Bonding Devices"* for the original implementation). The term *Network Teaming* is used to refer to the new implementation of this concept.

The main difference between bonding and Network Teaming is that teaming supplies a set of small kernel modules responsible for providing an interface for teamd instances. Everything else is handled in user space. This is different from the original bonding implementation which contains all of its functionality exclusively in the kernel. For a comparison refer to *Table 13.5, "Feature Comparison between Bonding and Team"*.

TABLE 13.5: FEATURE COMPARISON BETWEEN BONDING AND TEAM

Feature	Bonding	Team
broadcast, round-robin TX policy	yes	yes
active-backup TX policy	yes	yes
LACP (802.3ad) support	yes	yes
hash-based TX policy	yes	yes
user can set hash function	no	yes
TX load-balancing support (TLB)	yes	yes

Feature	Bonding	Team
TX load-balancing support for LACP	no	yes
Ethtool link monitoring	yes	yes
ARP link monitoring	yes	yes
NS/NA (IPV6) link monitoring	no	yes
RCU locking on TX/RX paths	no	yes
port prio and stickiness	no	yes
separate per-port link monitoring setup	no	yes
multiple link monitoring setup	limited	yes
VLAN support	yes	yes
multiple device stacking	yes	yes
Source: http://libteam.org/files/teamdev.pp.pdf		

Both implementations, bonding and Network Teaming, can be used in parallel. Network Teaming is an alternative to the existing bonding implementation. It does not replace bonding.

Network Teaming can be used for different use cases. The two most important use cases are explained later and involve:

- Load balancing between different network devices.
- Failover from one network device to another in case one of the devices should fail.

Currently, there is no YaST module to support creating a teaming device. You need to configure Network Teaming manually. The general procedure is shown below which can be applied for all your Network Teaming configurations:

PROCEDURE 13.1: GENERAL PROCEDURE

1. Make sure you have all the necessary packages installed. Install the packages `libteam-tools`, `libteamdctl0`, and `python-libteam`.

2. Create a configuration file under `/etc/sysconfig/network/`. Usually it will be `ifcfg-team0`. If you need more than one Network Teaming device, give them ascending numbers. This configuration file contains several variables which are explained in the man pages (see **`man ifcfg`** and **`man ifcfg-team`**). An example configuration can be found in your system in the file `/etc/sysconfig/network/ifcfg.template`.

3. Remove the configuration files of the interfaces which will be used for the teaming device (usually `ifcfg-eth0` and `ifcfg-eth1`).
   It is recommended to make a backup and remove both files. Wicked will re-create the configuration files with the necessary parameters for teaming.

4. Optionally, check if everything is included in Wicked's configuration file:

   ```
   tux > sudo wicked show-config
   ```

5. Start the Network Teaming device `team0`:

   ```
   tux > sudo wicked ifup all team0
   ```

   In case you need additional debug information, use the option `--debug all` after the **`all`** subcommand.

6. Check the status of the Network Teaming device. This can be done by the following commands:

   - Get the state of the teamd instance from Wicked:

     ```
     tux > sudo wicked ifstatus --verbose team0
     ```

   - Get the state of the entire instance:

     ```
     tux > sudo teamdctl team0 state
     ```

   - Get the systemd state of the teamd instance:

     ```
     tux > sudo systemctl status teamd@team0
     ```

   Each of them shows a slightly different view depending on your needs.

7. In case you need to change something in the `ifcfg-team0` file afterward, reload its configuration with:

```
tux > sudo wicked ifreload team0
```

Do *not* use **`systemctl`** for starting or stopping the teaming device! Instead, use the **`wicked`** command as shown above.

To completely remove the team device, use this procedure:

PROCEDURE 13.2: REMOVING A TEAM DEVICE

1. Stop the Network Teaming device `team0`:

```
tux > sudo wicked ifdown team0
```

2. Rename the file `/etc/sysconfig/network/ifcfg-team0` to `/etc/sysconfig/network/.ifcfg-team0`. Inserting a dot in front of the file name makes it "invisible" for wicked. If you really do not need the configuration anymore, you can also remove the file.

3. Reload the configuration:

```
tux > sudo wicked ifreload all
```

13.9.1 Use Case: Load Balancing with Network Teaming

Load balancing is used to improve bandwidth. Use the following configuration file to create a Network Teaming device with load balancing capabilities. Proceed with *Procedure 13.1, "General Procedure"* to set up the device. Check the output with **`teamdctl`**.

EXAMPLE 13.12: CONFIGURATION FOR LOAD BALANCING WITH NETWORK TEAMING

```
STARTMODE=auto ❶
BOOTPROTO=static ❷
IPADDRESS="192.168.1.1/24" ❷
IPADDR6="fd00:deca:fbad:50::1/64" ❷

TEAM_RUNNER="loadbalance" ❸
TEAM_LB_TX_HASH="ipv4,ipv6,eth,vlan"
TEAM_LB_TX_BALANCER_NAME="basic"
TEAM_LB_TX_BALANCER_INTERVAL="100"

TEAM_PORT_DEVICE_0="eth0" ❹
```

```
TEAM_PORT_DEVICE_1="eth1" ❹

TEAM_LW_NAME="ethtool" ❺
TEAM_LW_ETHTOOL_DELAY_UP="10" ❻
TEAM_LW_ETHTOOL_DELAY_DOWN="10" ❻
```

❶ Controls the start of the teaming device. The value of `auto` means, the interface will be set up when the network service is available and will be started automatically on every reboot. In case you need to control the device yourself (and prevent it from starting automatically), set `STARTMODE` to `manual`.

❷ Sets a static IP address (here `192.168.1.1` for IPv4 and `fd00:deca:fbad:50::1` for IPv6). If the Network Teaming device should use a dynamic IP address, set `BOOTPROTO="dhcp"` and remove (or comment) the line with `IPADDRESS` and `IPADDR6`.

❸ Sets `TEAM_RUNNER` to `loadbalance` to activate the load balancing mode.

❹ Specifies one or more devices which should be aggregated to create the Network Teaming device.

❺ Defines a link watcher to monitor the state of subordinate devices. The default value `ethtool` checks only if the device is up and accessible. This makes this check fast enough. However, it does not check if the device can really send or receive packets.
If you need a higher confidence in the connection, use the `arp_ping` option. This sends pings to an arbitrary host (configured in the `TEAM_LW_ARP_PING_TARGET_HOST` variable). The Network Teaming device is considered to be up only if the replies are received.

❻ Defines the delay in milliseconds between the link coming up (or down) and the runner being notified.

13.9.2 Use Case: Failover with Network Teaming

Failover is used to ensure high availability of a critical Network Teaming device by involving a parallel backup network device. The backup network device is running all the time and takes over if and when the main device fails.

Use the following configuration file to create a Network Teaming device with failover capabilities. Proceed with *Procedure 13.1, "General Procedure"* to set up the device. Check the output with **`teamdctl`**.

EXAMPLE 13.13: CONFIGURATION FOR DHCP NETWORK TEAMING DEVICE

```
STARTMODE=auto ❶
```

```
BOOTPROTO=static ❷
IPADDR="192.168.1.2/24" ❷
IPADDR6="fd00:deca:fbad:50::2/64" ❷

TEAM_RUNNER=activebackup ❸
TEAM_PORT_DEVICE_0="eth0" ❹
TEAM_PORT_DEVICE_1="eth1" ❹

TEAM_LW_NAME=ethtool ❺
TEAM_LW_ETHTOOL_DELAY_UP="10" ❻
TEAM_LW_ETHTOOL_DELAY_DOWN="10" ❻
```

❶ Controls the start of the teaming device. The value of `auto` means the interface will be set up when the network service is available and will be started automatically on every reboot. In case you need to control the device yourself (and prevent it from starting automatically), set `STARTMODE` to `manual`.

❷ Sets a static IP address (here `192.168.1.2` for IPv4 and `fd00:deca:fbad:50::2` for IPv6). If the Network Teaming device should use a dynamic IP address, set `BOOTPROTO="dhcp"` and remove (or comment) the line with `IPADDRESS` and `IPADDR6`.

❸ Sets `TEAM_RUNNER` to `activebackup` to activate the failover mode.

❹ Specifies one or more devices which should be aggregated to create the Network Teaming device.

❺ Defines a link watcher to monitor the state of subordinate devices. The default value `ethtool` checks only if the device is up and accessible. This makes this check fast enough. However, it does not check if the device can really send or receive packets.
If you need a higher confidence in the connection, use the `arp_ping` option. This sends pings to an arbitrary host (configured in the `TEAM_LW_ARP_PING_TARGET_HOST` variable). Only if the replies are received, the Network Teaming device is considered to be up.

❻ Defines the delay in milliseconds between the link coming up (or down) and the runner being notified.

13.9.3 Use Case: VLAN over Team Device

VLAN is an abbreviation of *Virtual Local Area Network*. It allows the running of multiple *logical* (virtual) Ethernets over one single physical Ethernet. It logically splits the network into different broadcast domains so that packets are only switched between ports that are designated for the same VLAN.

The following use case creates two static VLANs on top of a team device:

- `vlan0`, bound to the IP address `192.168.10.1`
- `vlan1`, bound to the IP address `192.168.20.1`

Proceed as follows:

1. Enable the VLAN tags on your switch. To use load balancing for your team device, your switch needs to be capable of *Link Aggregation Control Protocol* (LACP) (802.3ad). Consult your hardware manual about the details.
2. Decide if you want to use load balancing or failover for your team device. Set up your team device as described in *Section 13.9.1, "Use Case: Load Balancing with Network Teaming"* or *Section 13.9.2, "Use Case: Failover with Network Teaming"*.
3. In `/etc/sysconfig/network` create a file `ifcfg-vlan0` with the following content:

```
STARTMODE="auto"
BOOTPROTO="static" ❶
IPADDR='192.168.10.1/24' ❷
ETHERDEVICE="team0" ❸
VLAN_ID="0" ❹
VLAN='yes'
```

   ❶ Defines a fixed IP address, specified in `IPADDR`.

   ❷ Defines the IP address, here with its netmask.

   ❸ Contains the real interface to use for the VLAN interface, here our team device (`team0`).

   ❹ Specifies a unique ID for the VLAN. Preferably, the file name and the `VLAN_ID` corresponds to the name `ifcfg-vlan`*`VLAN_ID`*. In our case `VLAN_ID` is `0` which leads to the file name `ifcfg-vlan0`.

4. Copy the file `/etc/sysconfig/network/ifcfg-vlan0` to `/etc/sysconfig/network/ifcfg-vlan1` and change the following values:
   - `IPADDR` from `192.168.10.1/24` to `192.168.20.1/24`.
   - `VLAN_ID` from `0` to `1`.
5. Start the two VLANs:

```
root # wicked ifup vlan0 vlan1
```

6. Check the output of **ifconfig**:

```
root # ifconfig -a
[...]
vlan0     Link encap:Ethernet  HWaddr 08:00:27:DC:43:98
          inet addr:192.168.10.1 Bcast:192.168.10.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fedc:4398/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:816 (816.0 b)

vlan1     Link encap:Ethernet  HWaddr 08:00:27:DC:43:98
          inet addr:192.168.20.1 Bcast:192.168.20.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fedc:4398/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:816 (816.0 b)
```

13.10 Software-Defined Networking with Open vSwitch

Software-defined networking (SDN) means separating the system that controls where traffic is sent (the *control plane*) from the underlying system that forwards traffic to the selected destination (the *data plane*, also called the *forwarding plane*). This means that the functions previously fulfilled by a single, usually inflexible switch can now be separated between a switch (data plane) and its controller (control plane). In this model, the controller is programmable and can be very flexible and adapt quickly to changing network conditions.

Open vSwitch is software that implements a distributed virtual multilayer switch that is compatible with the OpenFlow protocol. OpenFlow allows a controller application to modify the configuration of a switch. OpenFlow is layered onto the TCP protocol and is implemented in a range of hardware and software. A single controller can thus drive multiple, very different switches.

13.10.1 Advantages of Open vSwitch

Software-defined networking with Open vSwitch brings several advantages with it, especially when you used together with virtual machines:

- Networking states can be identified easily.
- Networks and their live state can be moved from one host to another.
- Network dynamics are traceable and external software can be enabled to respond to them.
- You can apply and manipulate tags in network packets to identify which machine they are coming from or going to and maintain other networking context. Tagging rules can be configured and migrated.
- Open vSwitch implements the GRE protocol (*Generic Routing Encapsulation*). This allows you, for example, to connect private VM networks to each other.
- Open vSwitch can be used on its own, but is designed to integrate with networking hardware and can control hardware switches.

13.10.2 Installing Open vSwitch

1. Install Open vSwitch and supplementary packages:

   ```
   root # zypper install openvswitch openvswitch-switch
   ```

   If you plan to use Open vSwitch together with the KVM hypervisor, additionally install `tunctl` . If you plan to use Open vSwitch together with the Xen hypervisor, additionally install `openvswitch-kmp-xen` .

2. Enable the Open vSwitch service:

   ```
   root # systemctl enable openvswitch
   ```

3. Either restart the computer or use **`systemctl`** to start the Open vSwitch service immediately:

   ```
   root # systemctl start openvswitch
   ```

4. To check whether Open vSwitch was activated correctly, use:

   ```
   root # systemctl status openvswitch
   ```

13.10.3 Overview of Open vSwitch Daemons and Utilities

Open vSwitch consists of several components. Among them are a kernel module and various user space components. The kernel module is used for accelerating the data path, but is not necessary for a minimal Open vSwitch installation.

13.10.3.1 Daemons

The central executables of Open vSwitch are its two daemons. When you start the `openvswitch` service, you are indirectly starting them.

The main Open vSwitch daemon (**`ovs-vswitchd`**) provides the implementation of a switch. The Open vSwitch database daemon (**`ovsdb-server`**) serves the database that stores the configuration and state of Open vSwitch.

13.10.3.2 Utilities

Open vSwitch also comes with several utilities that help you work with it. The following list is not exhaustive, but instead describes important commands only.

`ovsdb-tool`
: Create, upgrade, compact, and query Open vSwitch databases. Do transactions on Open vSwitch databases.

`ovs-appctl`
: Configure a running **`ovs-vswitchd`** or **`ovsdb-server`** daemon.

`ovs-dpctl`, **`ovs-dpctl-top`**
: Create, modify, visualize, and delete data paths. Using this tool can interfere with **`ovs-vswitchd`** also performing data path management. Therefore, it is often used for diagnostics only.

  `ovs-dpctl-top` creates a **`top`**-like visualization for data paths.

`ovs-ofctl`
: Manage any switches adhering to the OpenFlow protocol. **`ovs-ofctl`** is not limited to interacting with Open vSwitch.

`ovs-vsctl`

Provides a high-level interface to the configuration database. It can be used to query and modify the database. In effect, it shows the status of **ovs-vswitchd** and can be used to configure it.

13.10.4 Creating a Bridge with Open vSwitch

The following example configuration uses the Wicked network service that is used by default on openSUSE Leap. To learn more about Wicked, see *Section 13.6, "Configuring a Network Connection Manually"*.

When you have installed and started Open vSwitch, proceed as follows:

1. To configure a bridge for use by your virtual machine, create a file with content like this:

```
STARTMODE='auto' ❶
BOOTPROTO='dhcp' ❷
OVS_BRIDGE='yes' ❸
OVS_BRIDGE_PORT_DEVICE_1='eth0' ❹
```

   ❶ Set up the bridge automatically when the network service is started.

   ❷ The protocol to use for configuring the IP address.

   ❸ Mark the configuration as an Open vSwitch bridge.

   ❹ Choose which device/devices should be added to the bridge. To add more devices, append additional lines for each of them to the file:

```
OVS_BRIDGE_PORT_DEVICE_SUFFIX='DEVICE'
```

   The *SUFFIX* can be any alphanumeric string. However, to avoid overwriting a previous definition, make sure the *SUFFIX* of each device is unique.

   Save the file in the directory `/etc/sysconfig/network` under the name `ifcfg-br0`. Instead of *br0*, you can use any name you want. However, the file name needs to begin with `ifcfg-`.

   To learn about further options, refer to the man pages of `ifcfg` (**man 5 ifcfg**) and `ifcfg-ovs-bridge` (**man 5 ifcfg-ovs-bridge**).

2. Now start the bridge:

```
root # wicked ifup br0
```

When Wicked is done, it should output the name of the bridge and next to it the state `up`.

13.10.5 Using Open vSwitch Directly with KVM

After having created the bridge as described in *Section 13.10.4, "Creating a Bridge with Open vSwitch"*, you can use Open vSwitch to manage the network access of virtual machines created with KVM/QEMU.

1. To be able to best use the capabilities of Wicked, make some further changes to the bridge configured before. Open the previously created `/etc/sysconfig/network/ifcfg-br0` and append a line for another port device:

   ```
   OVS_BRIDGE_PORT_DEVICE_2='tap0'
   ```

   Additionally, set `BOOTPROTO` to `none`. The file should now look like this:

   ```
   STARTMODE='auto'
   BOOTPROTO='none'
   OVS_BRIDGE='yes'
   OVS_BRIDGE_PORT_DEVICE_1='eth0'
   OVS_BRIDGE_PORT_DEVICE_2='tap0'
   ```

   The new port device *`tap0`* will be configured in the next step.

2. Now add a configuration file for the *`tap0`* device:

   ```
   STARTMODE='auto'
   BOOTPROTO='none'
   TUNNEL='tap'
   ```

   Save the file in the directory `/etc/sysconfig/network` under the name `ifcfg-tap0`.

Tip: Allowing Other Users to Access the Tap Device

To be able to use this tap device from a virtual machine started as a user who is not `root`, append:

```
TUNNEL_SET_OWNER=USER_NAME
```

To allow access for an entire group, append:

```
TUNNEL_SET_GROUP=GROUP_NAME
```

3. Finally, open the configuration for the device defined as the first `OVS_BRIDGE_PORT_DEVICE`. If you did not change the name, that should be `eth0`. Therefore, open `/etc/sysconfig/network/ifcfg-eth0` and make sure that the following options are set:

   ```
   STARTMODE='auto'
   BOOTPROTO='none'
   ```

   If the file does not exist yet, create it.

4. Restart the bridge interface using Wicked:

   ```
   root # wicked ifreload br0
   ```

   This will also trigger a reload of the newly defined bridge port devices.

5. To start a virtual machine, use, for example:

   ```
   root # qemu-kvm \
   -drive file=/PATH/TO/DISK-IMAGE ❶ \
   -m 512 -net nic,vlan=0,macaddr=00:11:22:EE:EE:EE \
   -net tap,ifname=tap0,script=no,downscript=no ❷
   ```

   ❶ The path to the QEMU disk image you want to start.

   ❷ Use the tap device (`tap0`) created before.

   For further information on the usage of KVM/QEMU, see *Book "*Virtualization Guide".

13.10.6 Using Open vSwitch with `libvirt`

After having created the bridge as described before in *Section 13.10.4, "Creating a Bridge with Open vSwitch"*, you can add the bridge to an existing virtual machine managed with `libvirt`. Since `libvirt` has some support for Open vSwitch bridges already, you can use the bridge created in *Section 13.10.4, "Creating a Bridge with Open vSwitch"* without further changes to the networking configuration.

1. Open the domain XML file for the intended virtual machine:

```
root # virsh edit VM_NAME
```

   Replace `VM_NAME` with the name of the desired virtual machine. This will open your default text editor.

2. Find the networking section of the document by looking for a section starting with `<interface type="...">` and ending in `</interface>`.
   Replace the existing section with a networking section that looks somewhat like this:

```
<interface type='bridge'>
  <source bridge='br0'/>
  <virtualport type='openvswitch'/>
</interface>
```

   Important: Compatibility of `virsh iface-*` and Virtual Machine Manager with Open vSwitch

   At the moment, the Open vSwitch compatibility of `libvirt` is not exposed through the `virsh iface-*` tools and Virtual Machine Manager. If you use any of these tools, your configuration can break.

3. You can now start or restart the virtual machine as usual.

For further information on the usage of `libvirt`, see *Book* "Virtualization Guide".

13.10.7 For More Information

http://openvswitch.org/support/

The documentation section of the Open vSwitch project Web site

https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

Whitepaper by the Open Networking Foundation about software-defined networking and the OpenFlow protocol

14 UEFI (Unified Extensible Firmware Interface)

UEFI (Unified Extensible Firmware Interface) is the interface between the firmware that comes with the system hardware, all the hardware components of the system, and the operating system.

UEFI is becoming more and more available on PC systems and thus is replacing the traditional PC-BIOS. UEFI, for example, properly supports 64-bit systems and offers secure booting ("Secure Boot", firmware version 2.3.1c or better required), which is one of its most important features. Lastly, with UEFI a standard firmware will become available on all x86 platforms.

UEFI additionally offers the following advantages:

- Booting from large disks (over 2 TiB) with a GUID Partition Table (GPT).
- CPU-independent architecture and drivers.
- Flexible pre-OS environment with network capabilities.
- CSM (Compatibility Support Module) to support booting legacy operating systems via a PC-BIOS-like emulation.

For more information, see http://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface. The following sections are not meant as a general UEFI overview; these are only hints about how some features are implemented in openSUSE Leap.

14.1 Secure Boot

In the world of UEFI, securing the bootstrapping process means establishing a chain of trust. The "platform" is the root of this chain of trust; in the context of openSUSE Leap, the mainboard and the on-board firmware could be considered the "platform". In other words, it is the hardware vendor, and the chain of trust flows from that hardware vendor to the component manufacturers, the OS vendors, etc.

The trust is expressed via public key cryptography. The hardware vendor puts a so-called Platform Key (PK) into the firmware, representing the root of trust. The trust relationship with operating system vendors and others is documented by signing their keys with the Platform Key.

Finally, security is established by requiring that no code will be executed by the firmware unless it has been signed by one of these "trusted" keys—be it an OS boot loader, some driver located in the flash memory of some PCI Express card or on disk, or be it an update of the firmware itself.

To use Secure Boot, you need to have your OS loader signed with a key trusted by the firmware, and you need the OS loader to verify that the kernel it loads can be trusted.

Key Exchange Keys (KEK) can be added to the UEFI key database. This way, you can use other certificates, as long as they are signed with the private part of the PK.

14.1.1 Implementation on openSUSE Leap

Microsoft's Key Exchange Key (KEK) is installed by default.

Note: GUID Partitioning Table (GPT) Required

The Secure Boot feature is enabled by default on UEFI/x86_64 installations. You can find the *Enable Secure Boot Support* option in the *Boot Code Options* tab of the *Boot Loader Settings* dialog. It supports booting when the secure boot is activated in the firmware, while making it possible to boot when it is deactivated.

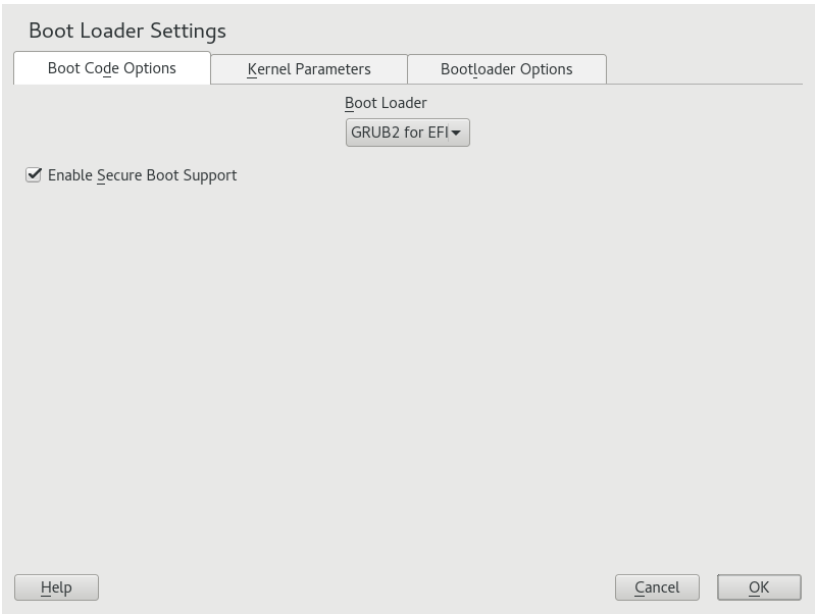


FIGURE 14.1: SECURE BOOT SUPPORT

The Secure Boot feature requires that a GUID Partitioning Table (GPT) replaces the old partitioning with a Master Boot Record (MBR). If YaST detects EFI mode during the installation, it will try to create a GPT partition. UEFI expects to find the EFI programs on a FAT-formatted EFI System Partition (ESP).

Supporting UEFI Secure Boot requires having a boot loader with a digital signature that the firmware recognizes as a trusted key. That key is trusted by the firmware a priori, without requiring any manual intervention.

There are two ways of getting there. One is to work with hardware vendors to have them endorse a SUSE key, which SUSE then signs the boot loader with. The other way is to go through Microsoft's Windows Logo Certification program to have the boot loader certified and have Microsoft recognize the SUSE signing key (that is, have it signed with their KEK). By now, SUSE got the loader signed by UEFI Signing Service (that is Microsoft in this case).

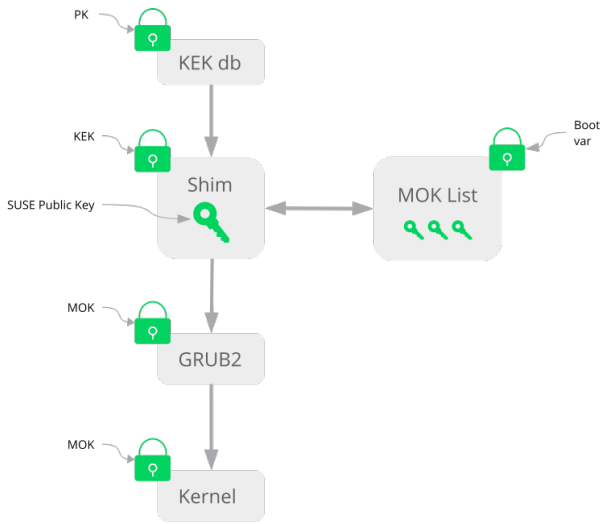


FIGURE 14.2: UEFI: SECURE BOOT PROCESS

At the implementation layer, SUSE uses the `shim` loader which is installed by default. It is a smart solution that avoids legal issues, and simplifies the certification and signing step considerably. The `shim` loader's job is to load a boot loader such as GRUB 2 and verify it; this boot loader in turn will load kernels signed by a SUSE key only.

There are two types of trusted users:

- First, those who hold the keys. The Platform Key (PK) allows almost everything. The Key Exchange Key (KEK) allows all a PK can except changing the PK.
- Second, anyone with physical access to the machine. A user with physical access can reboot the machine, and configure UEFI.

UEFI offers two types of variables to fulfill the needs of those users:

- The first is the so-called "Authenticated Variables", which can be updated from both within the boot process (the so-called Boot Services Environment) and the running OS. This can be done only when the new value of the variable is signed with the same key that the old value of the variable was signed with. And they can only be appended to or changed to a value with a higher serial number.
- The second is the so-called "Boot Services Only Variables". These variables are accessible to any code that runs during the boot process. After the boot process ends and before the OS starts, the boot loader must call the `ExitBootServices` call. After that, these variables are no longer accessible, and the OS cannot touch them.

The various UEFI key lists are of the first type, as this allows online updating, adding, and blacklisting of keys, drivers, and firmware fingerprints. It is the second type of variable, the "Boot Services Only Variable", that helps to implement Secure Boot in a secure and open source-friendly manner, and thus compatible with GPLv3.

SUSE starts with `shim`—a small and simple EFI boot loader signed by SUSE and Microsoft.

This allows `shim` to load and execute.

`shim` then goes on to verify that the boot loader it wants to load is trusted. In a default situation `shim` will use an independent SUSE certificate embedded in its body. In addition, `shim` will allow to "enroll" additional keys, overriding the default SUSE key. In the following, we call them "Machine Owner Keys" or MOKs for short.

Next the boot loader will verify and then boot the kernel, and the kernel will do the same on the modules.

14.1.2 MOK (Machine Owner Key)

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 rather than a single MOK, you can make `shim` trust keys from several 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:

   ```
   tux > openssl pkcs12 -export -inkey key.asc -in cert.pem \
     -name kernel_cert -out cert.p12
   ```

3. Generate an NSS database for use with **`pesign`**:

   ```
   tux > certutil -d . -N
   ```

4. Import the key and the certificate contained in PKCS#12 into the NSS database:

```
tux > pk12util -d . -i cert.p12
```

5. "Bless" the kernel with the new signature using **`pesign`**:

```
tux > pesign -n . -c kernel_cert -i arch/x86/boot/bzImage \
  -o vmlinuz.signed -s
```

6. List the signatures on the kernel image:

```
tux > pesign -n . -S -i vmlinuz.signed
```

   At that point, you can install the kernel in `/boot` as usual. Because the kernel now has a custom signature the certificate used for signing needs to be imported into the UEFI firmware or MOK.

7. Convert the certificate to the DER format for import into the firmware or MOK:

```
tux > openssl x509 -in cert.pem -outform der -out cert.der
```

8. Copy the certificate to the ESP for easier access:

```
tux > sudo cp cert.der /boot/efi/
```

9. Use **`mokutil`** to launch the MOK list automatically.

   - a. Import the certificate to MOK:

     ```
     tux > mokutil --root-pw --import cert.der
     ```

     The `--root-pw` option enables usage of the `root` user directly.

   - b. Check the list of certificates that are prepared to be enrolled:

     ```
     tux > mokutil --list-new
     ```

   - c. Reboot the system; `shim` should launch MokManager. You need to enter the `root` password to confirm the import of the certificate to the MOK list.

   - d. Check if the newly imported key was enrolled:

     ```
     tux > mokutil --list-enrolled
     ```

- a. Alternatively, this is the procedure if you want to launch MOK manually:
  Reboot
- b. In the GRUB 2 menu press the 'c' key.
- c. Type:

```
chainloader $efibootdir/MokManager.efi
boot
```

- d. Select *Enroll key from disk*.
- e. Navigate to the `cert.der` file and press Enter.
- f. Follow the instructions to enroll the key. Normally this should be pressing '0' and then 'y' to confirm.
  Alternatively, the firmware menu may provide ways to add a new key to the Signature Database.

14.1.4 Using Non-Inbox Drivers

There is no support for adding non-inbox drivers (that is, drivers that do not come with openSUSE Leap) during installation with Secure Boot enabled. The signing key used for SolidDriver/PLDP is not trusted by default.

It is possible to install third party drivers during installation with Secure Boot enabled in two different ways. In both cases:

- Add the needed keys to the firmware database via firmware/system management tools before the installation. This option depends on the specific hardware you are using. Consult your hardware vendor for more information.
- Use a bootable driver ISO from https://drivers.suse.com/ or your hardware vendor to enroll the needed keys in the MOK list at first boot.

To use the bootable driver ISO to enroll the driver keys to the MOK list, follow these steps:

1. Burn the ISO image above to an empty CD/DVD medium.
2. Start the installation using the new CD/DVD medium, having the standard installation media at hand or a URL to a network installation server.

If doing a network installation, enter the URL of the network installation source on the boot command line using the `install=` option.
If doing installation from optical media, the installer will first boot from the driver kit and then ask to insert the first installation disk of the product.

3. An initrd containing updated drivers will be used for installation.

For more information, see https://drivers.suse.com/doc/Usage/Secure_Boot_Certificate.html.

14.1.5 Features and Limitations

When booting in Secure Boot mode, the following features apply:

- Installation to UEFI default boot loader location, a mechanism to keep or restore the EFI boot entry.
- Reboot via UEFI.
- Xen hypervisor will boot with UEFI when there is no legacy BIOS to fall back to.
- UEFI IPv6 PXE boot support.
- UEFI videomode support, the kernel can retrieve video mode from UEFI to configure KMS mode with the same parameters.
- UEFI booting from USB devices is supported.

When booting in Secure Boot mode, the following limitations apply:

- To ensure that Secure Boot cannot be easily circumvented, some kernel features are disabled when running under Secure Boot.
- Boot loader, kernel, and kernel modules must be signed.
- Kexec and Kdump are disabled.
- Hibernation (suspend on disk) is disabled.
- Access to `/dev/kmem` and `/dev/mem` is not possible, not even as root user.
- Access to the I/O port is not possible, not even as root user. All X11 graphical drivers must use a kernel driver.
- PCI BAR access through sysfs is not possible.

- `custom_method` in ACPI is not available.
- debugfs for asus-wmi module is not available.
- the `acpi_rsdp` parameter does not have any effect on the kernel.

14.2 For More Information

- 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:

```
tux > mv ~/.bashrc ~/.bashrc.old
```

```
tux > cp /etc/skel/.bashrc ~/.bashrc
tux > mv ~/.profile ~/.profile.old
tux > cp /etc/skel/.profile ~/.profile
```

Then copy personal adjustments back from the `*.old` files.

15.1.2 The cron Package

Use `cron` to automatically run commands in the background at predefined times. `cron` uses specially formatted time tables, and the tool comes with several default ones. Users can also specify custom tables, if needed.

The cron tables are located in `/var/spool/cron/tabs`. `/etc/crontab` serves as a systemwide cron table. Enter the user name to run the command directly after the time table and before the command. In *Example 15.1, "Entry in /etc/crontab"*, `root` is entered. Package-specific tables, located in `/etc/cron.d`, have the same format. See the **`cron`** man page (**`man cron`**).

EXAMPLE 15.1: ENTRY IN /ETC/CRONTAB

```
1-59/5 * * * *   root   test -x /usr/sbin/atrun && /usr/sbin/atrun
```

You cannot edit `/etc/crontab` by calling the command **`crontab -e`**. This file must be loaded directly into an editor, then modified and saved.

Several packages install shell scripts to the directories `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly` and `/etc/cron.monthly`, whose execution is controlled by `/usr/lib/cron/run-crons`. `/usr/lib/cron/run-crons` is run every 15 minutes from the main table (`/etc/crontab`). This guarantees that processes that may have been neglected can be run at the proper time.

To run the `hourly`, `daily` or other periodic maintenance scripts at custom times, remove the time stamp files regularly using `/etc/crontab` entries (see *Example 15.2, "/etc/crontab: Remove Time Stamp Files"*, which removes the `hourly` one before every full hour, the `daily` one once a day at 2:14 a.m., etc.).

EXAMPLE 15.2: /ETC/CRONTAB: REMOVE TIME STAMP FILES

```
59 *  * * *     root  rm -f /var/spool/cron/lastrun/cron.hourly
14 2  * * *     root  rm -f /var/spool/cron/lastrun/cron.daily
29 2  * * 6     root  rm -f /var/spool/cron/lastrun/cron.weekly
```

```
44 2  1 * *     root  rm -f /var/spool/cron/lastrun/cron.monthly
```

Or you can set `DAILY_TIME` in `/etc/sysconfig/cron` to the time at which `cron.daily` should start. The setting of `MAX_NOT_RUN` ensures that the daily tasks get triggered to run, even if the user did not turn on the computer at the specified `DAILY_TIME` for a longer time. The maximum value of `MAX_NOT_RUN` is 14 days.

The daily system maintenance jobs are distributed to various scripts for reasons of clarity. They are contained in the package `aaa_base`. `/etc/cron.daily` contains, for example, the components `suse.de-backup-rpmdb`, `suse.de-clean-tmp` or `suse.de-cron-local`.

15.1.3 Stopping Cron Status Messages

To avoid the mail-flood caused by cron status messages, the default value of `SEND_MAIL_ON_NO_ERROR` in `/etc/sysconfig/cron` is set to "`no`" for new installations. Even with this setting to "`no`", cron data output will still be sent to the `MAILTO` address, as documented in the cron man page.

In the update case it is recommended to set these values according to your needs.

15.1.4 Log Files: Package logrotate

There are several system services (*daemons*) that, along with the kernel itself, regularly record the system status and specific events onto log files. This way, the administrator can regularly check the status of the system at a certain point in time, recognize errors or faulty functions and troubleshoot them with pinpoint precision. These log files are normally stored in `/var/log` as specified by FHS and grow on a daily basis. The `logrotate` package helps control the growth of these files. For more details refer to *Book "*System Analysis and Tuning Guide*", Chapter 3 "Analyzing and Managing System Log Files", Section 3.3 "Managing Log Files with* **logrotate**".

15.1.5 The **locate** Command

`locate`, a command for quickly finding files, is not included in the standard scope of installed software. If desired, install the package `mlocate`, the successor of the package `findutils-locate`. The `updatedb` process is started automatically every night or about 15 minutes after booting the system.

15.1.6 The `ulimit` Command

With the **`ulimit`** (*user limits*) command, it is possible to set limits for the use of system resources and to have these displayed. **`ulimit`** is especially useful for limiting available memory for applications. With this, an application can be prevented from co-opting too much of the system resources and slowing or even hanging up the operating system.

`ulimit` can be used with various options. To limit memory usage, use the options listed in *Table 15.1, "`ulimit`: Setting Resources for the User"*.

TABLE 15.1: **`ulimit`**: SETTING RESOURCES FOR THE USER

`-m`	The maximum resident set size
`-v`	The maximum amount of virtual memory available to the shell
`-s`	The maximum size of the stack
`-c`	The maximum size of core files created
`-a`	All current limits are reported

Systemwide default entries are set in `/etc/profile`. Editing this file directly is not recommended, because changes will be overwritten during system upgrades. To customize systemwide profile settings, use `/etc/profile.local`. Per-user settings should be made in *`~USER/`*`.bashrc`.

EXAMPLE 15.3: **`ulimit`**: SETTINGS IN `~/.bashrc`

```
# Limits maximum resident set size (physical memory):
ulimit -m 98304

# Limits of virtual memory:
ulimit -v 98304
```

Memory allocations must be specified in KB. For more detailed information, see **`man bash`**.

Important: **`ulimit`** Support

Not all shells support **`ulimit`** directives. PAM (for example, `pam_limits`) offers comprehensive adjustment possibilities as an alternative to **`ulimit`**.

15.1.7 The **free** Command

The **free** command displays the total amount of free and used physical memory and swap space in the system and the buffers and cache consumed by the kernel. The concept of *available RAM* dates back to before the days of unified memory management. The slogan *free memory is bad memory* applies well to Linux. As a result, Linux has always made the effort to balance out caches without actually allowing free or unused memory.

The kernel does not have direct knowledge of any applications or user data. Instead, it manages applications and user data in a *page cache*. If memory runs short, parts of it are written to the swap partition or to files, from which they can initially be read using the **mmap** command (see **man mmap**).

The kernel also contains other caches, such as the *slab cache*, where the caches used for network access are stored. This may explain the differences between the counters in `/proc/meminfo`. Most, but not all, of them can be accessed via `/proc/slabinfo`.

However, if your goal is to find out how much RAM is currently being used, find this information in `/proc/meminfo`.

15.1.8 Man Pages and Info Pages

For some GNU applications (such as tar), the man pages are no longer maintained. For these commands, use the `--help` option to get a quick overview of the info pages, which provide more in-depth instructions. Info is GNU's hypertext system. Read an introduction to this system by entering **info** `info`. Info pages can be viewed with Emacs by entering **emacs** `-f info` or directly in a console with **info**. You can also use tkinfo, xinfo or the help system to view info pages.

15.1.9 Selecting Man Pages Using the **man** Command

To read a man page enter **man** *MAN_PAGE*. If a man page with the same name exists in different sections, they will all be listed with the corresponding section numbers. Select the one to display. If you do not enter a section number within a few seconds, the first man page will be displayed.

To change this to the default system behavior, set `MAN_POSIXLY_CORRECT=1` in a shell initialization file such as `~/.bashrc`.

15.1.10 Settings for GNU Emacs

GNU Emacs is a complex work environment. The following sections cover the configuration files processed when GNU Emacs is started. More information is available at http://www.gnu.org/software/emacs/.

On start-up, Emacs reads several files containing the settings of the user, system administrator and distributor for customization or preconfiguration. The initialization file `~/.emacs` is installed to the home directories of the individual users from `/etc/skel`. `.emacs`, in turn, reads the file `/etc/skel/.gnu-emacs`. To customize the program, copy `.gnu-emacs` to the home directory (with **`cp /etc/skel/.gnu-emacs ~/.gnu-emacs`**) and make the desired settings there.

`.gnu-emacs` defines the file `~/.gnu-emacs-custom` as `custom-file`. If users make settings with the `customize` options in Emacs, the settings are saved to `~/.gnu-emacs-custom`.

With openSUSE Leap, the `emacs` package installs the file `site-start.el` in the directory `/usr/share/emacs/site-lisp`. The file `site-start.el` is loaded before the initialization file `~/.emacs`. Among other things, `site-start.el` ensures that special configuration files distributed with Emacs add-on packages, such as `psgml`, are loaded automatically. Configuration files of this type are located in `/usr/share/emacs/site-lisp`, too, and always begin with `suse-start-`. The local system administrator can specify systemwide settings in `default.el`.

More information about these files is available in the Emacs info file under *Init File*: `info:/emacs/InitFile`. Information about how to disable the loading of these files (if necessary) is also provided at this location.

The components of Emacs are divided into several packages:

- The base package `emacs`.
- `emacs-x11` (usually installed): the program *with* X11 support.
- `emacs-nox`: the program *without* X11 support.
- `emacs-info`: online documentation in info format.
- `emacs-el`: the uncompiled library files in Emacs Lisp. These are not required at runtime.
- Numerous add-on packages can be installed if needed: `emacs-auctex` (LaTeX), `psgml` (SGML and XML), `gnuserv` (client and server operation) and others.

15.2 Virtual Consoles

Linux is a multiuser and multitasking system. The advantages of these features can be appreciated even on a stand-alone PC system. In text mode, there are six virtual consoles available. Switch between them using Alt–F1 through Alt–F6. The seventh console is reserved for X and the tenth console shows kernel messages.

To switch to a console from X without shutting it down, use Ctrl–Alt–F1 to Ctrl–Alt–F6. To return to X, press Alt–F7.

15.3 Keyboard Mapping

To standardize the keyboard mapping of programs, changes were made to the following files:

```
/etc/inputrc
/etc/X11/Xmodmap
/etc/skel/.emacs
/etc/skel/.gnu-emacs
/etc/skel/.vimrc
/etc/csh.cshrc
/etc/termcap
/usr/share/terminfo/x/xterm
/usr/share/X11/app-defaults/XTerm
/usr/share/emacs/VERSION/site-lisp/term/*.el
```

These changes only affect applications that use **terminfo** entries or whose configuration files are changed directly (**vi**, **emacs**, etc.). Applications not shipped with the system should be adapted to these defaults.

Under X, the compose key (multikey) can be enabled as explained in `/etc/X11/Xmodmap`.

Further settings are possible using the X Keyboard Extension (XKB). 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 localization (*L10N*). The abbreviations I18N and L10N are derived from the first and last letters of the words and, in between, the number of letters omitted.

Settings are made with `LC_` variables defined in the file `/etc/sysconfig/language`. This refers not only to *native language support*, but also to the categories *Messages* (Language), *Character Set*, *Sort Order*, *Time and Date*, *Numbers* and *Money*. Each of these categories can be defined directly with its own variable or indirectly with a master variable in the file `language` (see the **`locale`** man page).

`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 example, if you do not want to use the system-wide `en_US` for program messages, include `LC_MESSAGES=es_ES` so that messages are displayed in Spanish instead.

15.4.2 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_` namespace 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`.
- 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 when they are plugged in and recognized. Users of a certain device need to be informed about any changes in this device's recognized state. `udev` provides the needed infrastructure to dynamically maintain the device node files and symbolic links in the `/dev` directory. `udev` rules provide a way to plug external tools into the kernel device event processing. This allows you to customize `udev` device handling by adding certain scripts to execute as part of kernel device handling, or request and import additional data to evaluate during device handling.

16.1 The /dev Directory

The device nodes in the `/dev` directory provide access to the corresponding kernel devices. With `udev`, the `/dev` directory reflects the current state of the kernel. Every kernel device has one corresponding device file. If a device is disconnected from the system, the device node is removed.

The content of the `/dev` directory is kept on a temporary file system and all files are rendered at every system start-up. Manually created or modified files do not, by design, survive a reboot. Static files and directories that should always be in the `/dev` directory regardless of the state of the corresponding kernel device can be created with systemd-tmpfiles. The configuration files are found in `/usr/lib/tmpfiles.d/` and `/etc/tmpfiles.d/`; for more information, see the `systemd-tmpfiles(8)` man page.

16.2 Kernel uevents and udev

The required device information is exported by the `sysfs` file system. For every device the kernel has detected and initialized, a directory with the device name is created. It contains attribute files with device-specific properties.

Every time a device is added or removed, the kernel sends a uevent to notify `udev` of the change. The `udev` daemon reads and parses all rules from the `/usr/lib/udev/rules.d/*.rules` and `/etc/udev/rules.d/*.rules` files at start-up and keeps them in memory. If rules files are

changed, added or removed, the daemon can reload their in-memory representation with the command **udevadm control --reload**. For more details on udev rules and their syntax, refer to *Section 16.6, "Influencing Kernel Device Event Handling with udev Rules"*.

Every received event is matched against the set of provides rules. The rules can add or change event environment keys, request a specific name for the device node to create, add symbolic links pointing to the node or add programs to run after the device node is created. The driver core uevents are received from a kernel netlink socket.

16.3 Drivers, Kernel Modules and Devices

The kernel bus drivers probe for devices. For every detected device, the kernel creates an internal device structure while the driver core sends a uevent to the udev daemon. Bus devices identify themselves by a specially-formatted ID, which tells what kind of device it is. Usually these IDs consist of vendor and product ID and other subsystem-specific values. Every bus has its own scheme for these IDs, called MODALIAS. The kernel takes the device information, composes a MODALIAS ID string from it and sends that string along with the event. For a USB mouse, it looks like this:

```
MODALIAS=usb:v046DpC03Ed2000dc00dsc00dp00ic03isc01ip02
```

Every device driver carries a list of known aliases for devices it can handle. The list is contained in the kernel module file itself. The program depmod reads the ID lists and creates the file modules.alias in the kernel's /lib/modules directory for all currently available modules. With this infrastructure, module loading is as easy as calling **modprobe** for every event that carries a MODALIAS key. If **modprobe $MODALIAS** is called, it matches the device alias composed for the device with the aliases provided by the modules. If a matching entry is found, that module is loaded. All this is automatically triggered by udev.

16.4 Booting and Initial Device Setup

All device events happening during the boot process before the udev daemon is running are lost, because the infrastructure to handle these events resides on the root file system and is not available at that time. To cover that loss, the kernel provides a uevent file located in the device directory of every device in the sysfs file system. By writing add to that file, the kernel resends the same event as the one lost during boot. A simple loop over all uevent files in /sys triggers all events again to create the device nodes and perform device setup.

As an example, a USB mouse present during boot may not be initialized by the early boot logic, because the driver is not available at that time. The event for the device discovery was lost and failed to find a kernel module for the device. Instead of manually searching for connected devices, `udev` requests all device events from the kernel after the root file system is available, so the event for the USB mouse device runs again. Now it finds the kernel module on the mounted root file system and the USB mouse can be initialized.

From user space, there is no visible difference between a device coldplug sequence and a device discovery during runtime. In both cases, the same rules are used to match and the same configured programs are run.

16.5 Monitoring the Running udev Daemon

The program **`udevadm monitor`** can be used to visualize the driver core events and the timing of the `udev` event processes.

```
UEVENT[1185238505.276660] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1 (usb)
UDEV  [1185238505.279198] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1 (usb)
UEVENT[1185238505.279527] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0 (usb)
UDEV  [1185238505.285573] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0 (usb)
UEVENT[1185238505.298878] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10 (input)
UDEV  [1185238505.305026] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10 (input)
UEVENT[1185238505.305442] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/mouse2 (input)
UEVENT[1185238505.306440] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/event4 (input)
UDEV  [1185238505.325384] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/event4 (input)
UDEV  [1185238505.342257] add   /devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/
input10/mouse2 (input)
```

The `UEVENT` lines show the events the kernel has sent over netlink. The `UDEV` lines show the finished `udev` event handlers. The timing is printed in microseconds. The time between `UEVENT` and `UDEV` is the time `udev` took to process this event or the `udev` daemon has delayed its execution to synchronize this event with related and already running events. For example, events for hard disk partitions always wait for the main disk device event to finish, because the partition events may rely on the data that the main disk event has queried from the hardware.

udevadm monitor --env shows the complete event environment:

```
ACTION=add
DEVPATH=/devices/pci0000:00/0000:00:1d.2/usb3/3-1/3-1:1.0/input/input10
SUBSYSTEM=input
SEQNUM=1181
NAME="Logitech USB-PS/2 Optical Mouse"
PHYS="usb-0000:00:1d.2-1/input0"
UNIQ=""
EV=7
KEY=70000 0 0 0 0
REL=103
MODALIAS=input:b0003v046DpC03Ee0110-e0,1,2,k110,111,112,r0,1,8,amlsfw
```

`udev` also sends messages to syslog. The default syslog priority that controls which messages are sent to syslog is specified in the `udev` configuration file `/etc/udev/udev.conf`. The log priority of the running daemon can be changed with **udevadm control --log_priority=** *LEVEL/NUMBER*.

16.6 Influencing Kernel Device Event Handling with udev Rules

A `udev` rule can match any property the kernel adds to the event itself or any information that the kernel exports to `sysfs`. The rule can also request additional information from external programs. Events are matched against all rules provided in the directories `/usr/lib/udev/rules.d/` (for default rules) and `/etc/udev/rules.d` (system-specific configuration).

Every line in the rules file contains at least one key value pair. There are two kinds of keys, match and assignment keys. If all match keys match their values, the rule is applied and the assignment keys are assigned the specified value. A matching rule may specify the name of the device node, add symbolic links pointing to the node or run a specified program as part of the event handling. If no matching rule is found, the default device node name is used to create the device node. Detailed information about the rule syntax and the provided keys to match or import data are described in the `udev` man page. The following example rules provide a basic introduction to `udev` rule syntax. The example rules are all taken from the `udev` default rule set `/usr/lib/udev/rules.d/50-udev-default.rules`.

EXAMPLE 16.1: EXAMPLE udev RULES

```
# console
```

```
KERNEL=="console", MODE="0600", OPTIONS="last_rule"

# serial devices
KERNEL=="ttyUSB*", ATTRS{product}=="[Pp]alm*Handheld*", SYMLINK+="pilot"

# printer
SUBSYSTEM=="usb", KERNEL=="lp*", NAME="usb/%k", SYMLINK+="usb%k", GROUP="lp"

# kernel firmware loader
SUBSYSTEM=="firmware", ACTION=="add", RUN+="firmware.sh"
```

The `console` rule consists of three keys: one match key (`KERNEL`) and two assign keys (`MODE`, `OPTIONS`). The `KERNEL` match rule searches the device list for any items of the type `console`. Only exact matches are valid and trigger this rule to be executed. The `MODE` key assigns special permissions to the device node, in this case, read and write permissions to the owner of this device only. The `OPTIONS` key makes this rule the last rule to be applied to any device of this type. Any later rule matching this particular device type does not have any effect.

The `serial devices` rule is not available in `50-udev-default.rules` anymore, but it is still worth considering. It consists of two match keys (`KERNEL` and `ATTRS`) and one assign key (`SYMLINK`). The `KERNEL` key searches for all devices of the `ttyUSB` type. Using the `*` wild card, this key matches several of these devices. The second match key, `ATTRS`, checks whether the `product` attribute file in `sysfs` for any `ttyUSB` device contains a certain string. The assign key (`SYMLINK`) triggers the addition of a symbolic link to this device under `/dev/pilot`. The operator used in this key (`+=`) tells `udev` to additionally perform this action, even if previous or later rules add other symbolic links. As this rule contains two match keys, it is only applied if both conditions are met.

The `printer` rule deals with USB printers and contains two match keys which must both apply to get the entire rule applied (`SUBSYSTEM` and `KERNEL`). Three assign keys deal with the naming for this device type (`NAME`), the creation of symbolic device links (`SYMLINK`) and the group membership for this device type (`GROUP`). Using the `*` wild card in the `KERNEL` key makes it match several `lp` printer devices. Substitutions are used in both, the `NAME` and the `SYMLINK` keys to extend these strings by the internal device name. For example, the symbolic link to the first `lp` USB printer would read `/dev/usblp0`.

The `kernel firmware loader` rule makes `udev` load additional firmware by an external helper script during runtime. The `SUBSYSTEM` match key searches for the `firmware` subsystem. The `ACTION` key checks whether any device belonging to the `firmware` subsystem has been added. The `RUN+=` key triggers the execution of the `firmware.sh` script to locate the firmware that is to be loaded.

Some general characteristics are common to all rules:

- Each rule consists of one or more key value pairs separated by a comma.
- A key's operation is determined by the operator. `udev` rules support several operators.
- Each given value must be enclosed by quotation marks.
- Each line of the rules file represents one rule. If a rule is longer than one line, use `\` to join the different lines as you would do in shell syntax.
- `udev` rules support a shell-style pattern that matches the `*`, `?`, and `[]` patterns.
- `udev` rules support substitutions.

16.6.1 Using Operators in udev Rules

Creating keys you can choose from several operators, depending on the type of key you want to create. Match keys will normally be used to find a value that either matches or explicitly mismatches the search value. Match keys contain either of the following operators:

`==`

Compare for equality. If the key contains a search pattern, all results matching this pattern are valid.

`!=`

Compare for non-equality. If the key contains a search pattern, all results matching this pattern are valid.

Any of the following operators can be used with assign keys:

`=`

Assign a value to a key. If the key previously consisted of a list of values, the key resets and only the single value is assigned.

`+=`

Add a value to a key that contains a list of entries.

`:=`

Assign a final value. Disallow any later change by later rules.

16.6.2 Using Substitutions in udev Rules

`udev` rules support the use of placeholders and substitutions. Use them in a similar fashion as you would do in any other scripts. The following substitutions can be used with `udev` rules:

`%r`, `$root`
: The device directory, `/dev` by default.

`%p`, `$devpath`
: The value of `DEVPATH`.

`%k`, `$kernel`
: The value of `KERNEL` or the internal device name.

`%n`, `$number`
: The device number.

`%N`, `$tempnode`
: The temporary name of the device file.

`%M`, `$major`
: The major number of the device.

`%m`, `$minor`
: The minor number of the device.

`%s{`*`ATTRIBUTE`*`}`, `$attr{`*`ATTRIBUTE`*`}`
: The value of a `sysfs` attribute (specified by *`ATTRIBUTE`*).

`%E{`*`VARIABLE`*`}`, `$env{`*`VARIABLE`*`}`
: The value of an environment variable (specified by *`VARIABLE`*).

`%c`, `$result`
: The output of `PROGRAM`.

`%%`
: The `%` character.

`$$`
: The `$` character.

16.6.3 Using udev Match Keys

Match keys describe conditions that must be met before a `udev` rule can be applied. The following match keys are available:

`ACTION`
: The name of the event action, for example, `add` or `remove` when adding or removing a device.

`DEVPATH`
: The device path of the event device, for example, `DEVPATH=/bus/pci/drivers/ipw3945` to search for all events related to the ipw3945 driver.

`KERNEL`
: The internal (kernel) name of the event device.

`SUBSYSTEM`
: The subsystem of the event device, for example, `SUBSYSTEM=usb` for all events related to USB devices.

`ATTR{`*`FILENAME`*`}`
: `sysfs` attributes of the event device. To match a string contained in the `vendor` attribute file name, you could use `ATTR{vendor}=="On[sS]tream"`, for example.

`KERNELS`
: Let `udev` search the device path upward for a matching device name.

`SUBSYSTEMS`
: Let `udev` search the device path upward for a matching device subsystem name.

`DRIVERS`
: Let `udev` search the device path upward for a matching device driver name.

`ATTRS{`*`FILENAME`*`}`
: Let `udev` search the device path upward for a device with matching `sysfs` attribute values.

`ENV{`*`KEY`*`}`
: The value of an environment variable, for example, `ENV{ID_BUS}="ieee1394` to search for all events related to the FireWire bus ID.

`PROGRAM`

Let `udev` execute an external program. To be successful, the program must return with exit code zero. The program's output, printed to STDOUT, is available to the `RESULT` key.

`RESULT`

Match the output string of the last `PROGRAM` call. Either include this key in the same rule as the `PROGRAM` key or in a later one.

16.6.4 Using udev Assign Keys

In contrast to the match keys described above, assign keys do not describe conditions that must be met. They assign values, names and actions to the device nodes maintained by `udev`.

`NAME`

The name of the device node to be created. After a rule has set a node name, all other rules with a `NAME` key for this node are ignored.

`SYMLINK`

The name of a symbolic link related to the node to be created. Multiple matching rules can add symbolic links to be created with the device node. You can also specify multiple symbolic links for one node in one rule using the space character to separate the symbolic link names.

`OWNER, GROUP, MODE`

The permissions for the new device node. Values specified here overwrite anything that has been compiled in.

`ATTR{`*`KEY`*`}`

Specify a value to be written to a `sysfs` attribute of the event device. If the `==` operator is used, this key is also used to match against the value of a `sysfs` attribute.

`ENV{`*`KEY`*`}`

Tell `udev` to export a variable to the environment. If the `==` operator is used, this key is also used to match against an environment variable.

`RUN`

Tell `udev` to add a program to the list of programs to be executed for this device. Keep in mind to restrict this to very short tasks to avoid blocking further events for this device.

`LABEL`

Add a label where a `GOTO` can jump to.

`GOTO`

Tell `udev` to skip several rules and continue with the one that carries the label referenced by the `GOTO` key.

`IMPORT{`*`TYPE`*`}`

Load variables into the event environment such as the output of an external program. `udev` imports variables of several types. If no type is specified, `udev` tries to determine the type itself based on the executable bit of the file permissions.

- `program` tells `udev` to execute an external program and import its output.
- `file` tells `udev` to import a text file.
- `parent` tells `udev` to import the stored keys from the parent device.

`WAIT_FOR_SYSFS`

Tells `udev` to wait for the specified `sysfs` file to be created for a certain device. For example, `WAIT_FOR_SYSFS="ioerr_cnt"` informs `udev` to wait until the `ioerr_cnt` file has been created.

`OPTIONS`

The `OPTION` key may have several values:

- `last_rule` tells `udev` to ignore all later rules.
- `ignore_device` tells `udev` to ignore this event completely.
- `ignore_remove` tells `udev` to ignore all later remove events for the device.
- `all_partitions` tells `udev` to create device nodes for all available partitions on a block device.

16.7 Persistent Device Naming

The dynamic device directory and the `udev` rules infrastructure make it possible to provide stable names for all disk devices—regardless of their order of recognition or the connection used for the device. Every appropriate block device the kernel creates is examined by tools with special knowledge about certain buses, drive types or file systems. Along with the dynamic kernel-provided device node name, `udev` maintains classes of persistent symbolic links pointing to the device:

```
/dev/disk
```

```
|-- by-id
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B -> ../../sda
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part1 -> ../../sda1
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part6 -> ../../sda6
|   |-- scsi-SATA_HTS726060M9AT00_MRH453M4HWHG7B-part7 -> ../../sda7
|   |-- usb-Generic_STORAGE_DEVICE_02773 -> ../../sdd
|   `-- usb-Generic_STORAGE_DEVICE_02773-part1 -> ../../sdd1
|-- by-label
|   |-- Photos -> ../../sdd1
|   |-- SUSE10 -> ../../sda7
|   `-- devel -> ../../sda6
|-- by-path
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0 -> ../../sda
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part1 -> ../../sda1
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part6 -> ../../sda6
|   |-- pci-0000:00:1f.2-scsi-0:0:0:0-part7 -> ../../sda7
|   |-- pci-0000:00:1f.2-scsi-1:0:0:0 -> ../../sr0
|   |-- usb-02773:0:0:2 -> ../../sdd
|   |-- usb-02773:0:0:2-part1 -> ../../sdd1
`-- by-uuid
    |-- 159a47a4-e6e6-40be-a757-a629991479ae -> ../../sda7
    |-- 3e999973-00c9-4917-9442-b7633bd95b9e -> ../../sda6
    `-- 4210-8F8C -> ../../sdd1
```

16.8 Files used by udev

`/sys/*`

Virtual file system provided by the Linux kernel, exporting all currently known devices. This information is used by `udev` to create device nodes in `/dev`

`/dev/*`

Dynamically created device nodes and static content created with systemd-tmpfiles; for more information, see the `systemd-tmpfiles(8)` man page.

The following files and directories contain the crucial elements of the `udev` infrastructure:

`/etc/udev/udev.conf`

Main `udev` configuration file.

`/etc/udev/rules.d/*`

System-specific `udev` event matching rules. You can add custom rules here to modify or override the default rules from `/usr/lib/udev/rules.d/*`.

Files are parsed in alphanumeric order. Rules from files with a higher priority modify or override rules with lower priority. The lower the number, the higher the priority.

`/usr/lib/udev/rules.d/*`

Default `udev` event matching rules. The files in this directory are owned by packages and will be overwritten by updates. Do not add, remove or edit files here, use `/etc/udev/rules.d` instead.

`/usr/lib/udev/*`

Helper programs called from `udev` rules.

`/usr/lib/tmpfiles.d/` and `/etc/tmpfiles.d/`

Responsible for static `/dev` content.

16.9 For More Information

For more information about the `udev` infrastructure, refer to the following man pages:

`udev`

General information about `udev`, keys, rules and other important configuration issues.

`udevadm`

`udevadm` can be used to control the runtime behavior of `udev`, request kernel events, manage the event queue and provide simple debugging mechanisms.

`udevd`

Information about the `udev` event managing daemon.

III Services

17 SLP

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.

18 Time Synchronization with NTP

The NTP (network time protocol) mechanism is a protocol for synchronizing the system time over the network. First, a machine can obtain the time from a server that is a reliable time source. Second, a machine can itself act as a time source for other computers in the network. The goal is twofold—maintaining the absolute time and synchronizing the system time of all machines within a network.

Maintaining an exact system time is important in many situations. The built-in hardware clock does often not meet the requirements of applications such as databases or clusters. Manual correction of the system time would lead to severe problems because, for example, a backward leap can cause malfunction of critical applications. Within a network, it is usually necessary to synchronize the system time of all machines, but manual time adjustment is a bad approach. NTP provides a mechanism to solve these problems. The NTP service continuously adjusts the system time with reliable time servers in the network. It further enables the management of local reference clocks, such as radio-controlled clocks.

Since openSUSE Leap 15, `chrony` is the default implementation of NTP. `chrony` includes two parts; `chronyd` is a daemon that can be started at boot time and **`chronyc`** is a command line interface program to monitor the performance of `chronyd`, and to change various operating parameters at runtime.

Note

To enable time synchronization by means of active directory, follow the instructions found at *Book "*Security Guide*", Chapter 7 "Active Directory Support", Section 7.3.3 "Joining Active Directory Using Windows Domain Membership", Joining an Active Directory Domain Using Windows Domain Membership*.

18.1 Configuring an NTP Client with YaST

The NTP daemon (`chronyd`) coming with the `chrony` package is preset to use the local computer hardware clock as a time reference. The precision of a hardware clock heavily depends on its time source. For example, an atomic clock or GPS receiver is a very precise time source, while a common RTC chip is not a reliable time source. YaST simplifies the configuration of an NTP client.

In the YaST NTP client configuration (*Network Services* › *NTP Configuration*) window, you can specify when to start the NTP daemon, the type of the configuration source, and add custom time servers.

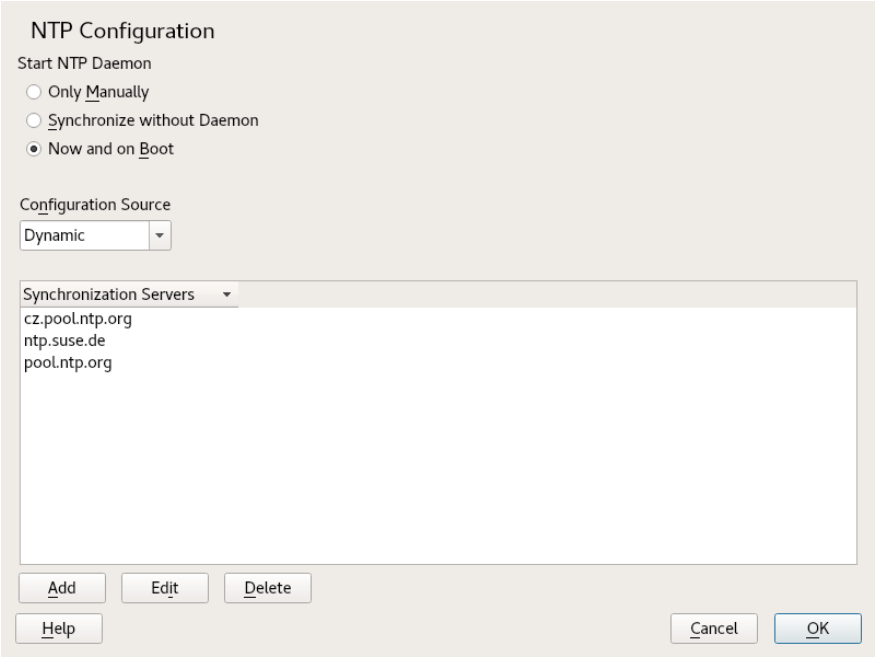


FIGURE 18.1: NTP CONFIGURATION WINDOW

18.1.1 NTP Daemon Start

You can choose from three options for when to start the NTP daemon:

Only Manually

Select *Only Manually*, if you want to manually start the `chrony` daemon.

Synchronize without Daemon

Select *Synchronize without Daemon* to set the system time periodically without a permanently running `chrony`. You can set the *Interval of the Synchronization in Minutes*.

Now and On Boot

Select *Now and On Boot* to start `chronyd` automatically when the system is booted. This setting is recommended.

18.1.2 Type of the Configuration Source

In the *Configuration Source* drop-down box, select either *Dynamic* or *Static*. Set *Static* if your server uses only a fixed set of (public) NTP servers, while *Dynamic* is better if your internal network offers NTP servers via DHCP.

18.1.3 Configure Time Servers

Time servers for the client to query are listed in the lower part of the *NTP Configuration* window. Modify this list as needed with *Add*, *Edit*, and *Delete*.

Click *Add* to add a new time server:

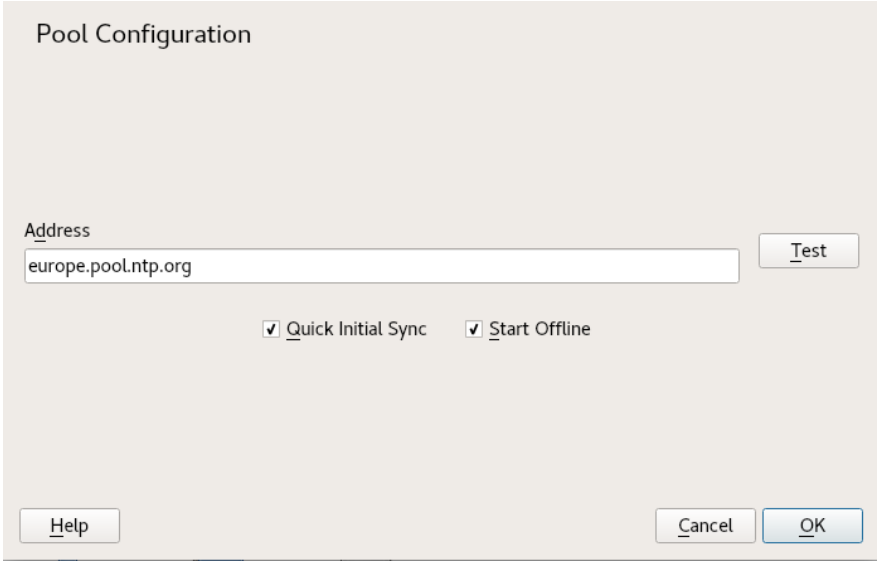


FIGURE 18.2: ADDING A TIME SERVER

1. In the *Address* field, type the URL of the time server or pool of time servers with which you want to synchronize the machine time. After the URL is complete, click *Test* to verify that it points to a valid time source.

2. Activate *Quick Initial Sync* to speed up the time synchronization by sending more requests at the `chronyd` daemon start.

3. Activate *Start Offline* to speed up the boot time on systems that start the `chronyd` daemon automatically and may not have an Internet connection at boot time. This option is useful for example for laptops whose network connection is managed by NetworkManager.

4. Confirm with *OK*.

18.2 Manually Configuring NTP in the Network

`chrony` reads its configuration from the `/etc/chrony.conf` file. To keep the computer clock synchronized, you need to tell `chrony` what time servers to use. You can use specific server names or IP addresses, for example:

```
server 0.europe.pool.ntp.org
server 1.europe.pool.ntp.org
server 2.europe.pool.ntp.org
```

You can also specify a *pool* name. Pool name resolves to several IP addresses:

```
pool pool.ntp.org
```

Tip: Computers on the Same Network

To synchronize time on multiple computers on the same network, we do not recommend to synchronize all of them with an external server. A good practice is to make one computer the time server which is synchronized with an external time server, and the other computers act as its clients. Add a `local` directive to the server's `/etc/chrony.conf` to distinguish it from an authoritative time server:

```
local stratum 10
```

To start `chrony`, run:

```
systemctl start chronyd.service
```

After initializing `chronyd`, it takes some time before the time is stabilized and the drift file for correcting the local computer clock is created. With the drift file, the systematic error of the hardware clock can be computed when the computer is powered on. The correction is used immediately, resulting in a higher stability of the system time.

To enable the service so that `chrony` starts automatically at boot time, run:

```
systemctl enable chronyd.service
```

18.3 Configure chronyd at Runtime Using **chronyc**

You can use **chronyc** to change the behavior of chronyd at runtime. It also generates status reports about the operation of chronyd.

You can run **chronyc** either in interactive or non-interactive mode. To run **chronyc** interactively, enter **chronyc** on the command line. It displays a prompt and waits for your command input. For example, to check how many NTP sources are online or offline, run:

```
root # chronyc
chronyc> activity
200 OK
4 sources online
2 sources offline
1 sources doing burst (return to online)
1 sources doing burst (return to offline)
0 sources with unknown address
```

To exit **chronyc**'s prompt, enter **quit** or **exit**.

If you do not need to use the interactive prompt, enter the command directly:

```
root # chronyc activity
```

Note: Temporary Changes

Changes made using **chronyc** are not permanent. They will be lost after the next chronyd restart. For permanent changes, modify /etc/chrony.conf.

For a complete list of **chronyc** commands, see its manual page (**man 1 chronyc**).

18.4 Dynamic Time Synchronization at Runtime

If the system boots without network connection, chronyd starts up, but it cannot resolve DNS names of the time servers set in the configuration file. This can happen if you use NetworkManager with an encrypted Wi-Fi.

chronyd keeps trying to resolve the time server names specified by the server, pool, and peer directives in an increasing time interval until it succeeds.

If the time server will not be reachable when `chronyd` is started, you can specify the `offline` option:

```
server server_address offline
```

`chronyd` will then not try to poll the server until it is enabled using the following command:

```
root # chronyc online server_address
```

When the `auto_offline` option is set, `chronyd` assumes that the time server has gone offline when two requests have been sent to it without receiving a response. This option avoids the need to run the 'offline' command from **`chronyc`** when disconnecting the network link.

18.5 Setting Up a Local Reference Clock

The software package `chrony` relies on other programs (such as `gpsd`) to access the timing data via the SHM or SOCK driver. Use the `refclock` directive in `/etc/chrony.conf` to specify a hardware reference clock to be used as a time source. It has two mandatory parameters: a driver name and a driver-specific parameter. The two parameters are followed by zero or more `refclock` options. `chronyd` includes the following drivers:

- PPS - driver for the kernel 'pulse per second' API. For example:

  ```
  refclock PPS /dev/pps0 lock NMEA refid GPS
  ```

- SHM - NTP shared memory driver. For example:

  ```
  refclock SHM 0 poll 3 refid GPS1
  refclock SHM 1:perm=0644 refid GPS2
  ```

- SOCK - Unix domain socket driver. For example:

  ```
  refclock SOCK /var/run/chrony.ttyS0.sock
  ```

- PHC - PTP hardware clock driver. For example:

  ```
  refclock PHC /dev/ptp0 poll 0 dpoll -2 offset -37
  refclock PHC /dev/ptp1:nocrossts poll 3 pps
  ```

For more information on individual drivers' options, see **`man 8 chrony.conf`**.

19 The Domain Name System

DNS (domain name system) is needed to resolve the domain names and host names into IP addresses. In this way, the IP address 192.168.2.100 is assigned to the host name `jupiter`, for example. Before setting up your own name server, read the general information about DNS in *Section 13.3, "Name Resolution"*. The following configuration examples refer to BIND, the default DNS server.

19.1 DNS Terminology

Zone

The domain name space is divided into regions called zones. For example, if you have `example.com`, you have the `example` section (or zone) of the `com` domain.

DNS server

The DNS server is a server that maintains the name and IP information for a domain. You can have a primary DNS server for master zone, a secondary server for slave zone, or a slave server without any zones for caching.

Master zone DNS server

The master zone includes all hosts from your network and a DNS server master zone stores up-to-date records for all the hosts in your domain.

Slave zone DNS server

A slave zone is a copy of the master zone. The slave zone DNS server obtains its zone data with zone transfer operations from its master server. The slave zone DNS server responds authoritatively for the zone as long as it has valid (not expired) zone data. If the slave cannot obtain a new copy of the zone data, it stops responding for the zone.

Forwarder

Forwarders are DNS servers to which your DNS server should send queries it cannot answer. To enable different configuration sources in one configuration, `netconfig` is used (see also **`man 8 netconfig`**).

Record

The record is information about name and IP address. Supported records and their syntax are described in BIND documentation. Some special records are:

NS record

An NS record tells name servers which machines are in charge of a given domain zone.

MX record

The MX (mail exchange) records describe the machines to contact for directing mail across the Internet.

SOA record

SOA (Start of Authority) record is the first record in a zone file. The SOA record is used when using DNS to synchronize data between multiple computers.

19.2 Installation

To install a DNS server, start YaST and select *Software › Software Management*. Choose *View › Patterns* and select *DHCP and DNS Server*. Confirm the installation of the dependent packages to finish the installation process.

Alternatively use the following command on the command line:

```
tux > sudo zypper in -t pattern dhcp_dns_server
```

19.3 Configuration with YaST

Use the YaST DNS module to configure a DNS server for the local network. When starting the module for the first time, a wizard starts, prompting you to make a few decisions concerning administration of the server. Completing this initial setup produces a basic server configuration. Use the expert mode to deal with more advanced configuration tasks, such as setting up ACLs, logging, TSIG keys, and other options.

19.3.1 Wizard Configuration

The wizard consists of three steps or dialogs. At the appropriate places in the dialogs, you can enter the expert configuration mode.

1. When starting the module for the first time, the *Forwarder Settings* dialog, shown in *Figure 19.1, "DNS Server Installation: Forwarder Settings"*, opens. The *Local DNS Resolution Policy* allows to set the following options:

   - *Merging forwarders is disabled*
   - *Automatic merging*
   - *Merging forwarders is enabled*
   - *Custom configuration*—If *Custom configuration* is selected, *Custom policy* can be specified; by default (with *Automatic merging* selected), *Custom policy* is set to `auto`, but here you can either set interface names or select from the two special policy names `STATIC` and `STATIC_FALLBACK`.

   In *Local DNS Resolution Forwarder*, specify which service to use: *Using system name servers*, *This name server (bind)*, or *Local dnsmasq server*.
   For more information about all these settings, see **`man 8 netconfig`**.

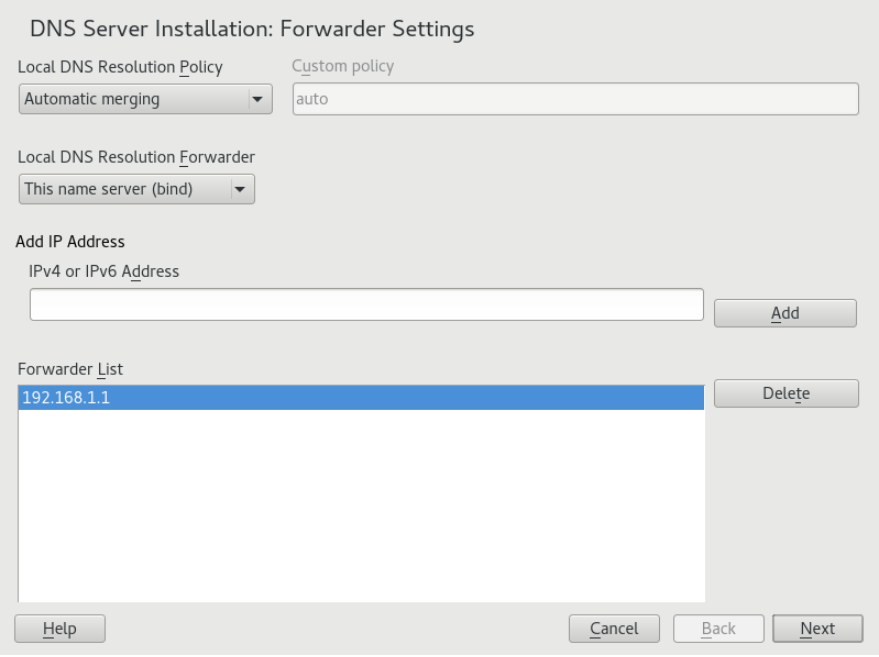


FIGURE 19.1: DNS SERVER INSTALLATION: FORWARDER SETTINGS

   Forwarders are DNS servers to which your DNS server sends queries it cannot answer itself. Enter their IP address and click *Add*.

2. The *DNS Zones* dialog consists of several parts and is responsible for the management of zone files, described in *Section 19.6, "Zone Files"*. For a new zone, provide a name for it in *Name*. To add a reverse zone, the name must end in `.in-addr.arpa`. Finally, select the *Type* (master, slave, or forward). See *Figure 19.2, "DNS Server Installation: DNS Zones"*. Click *Edit* to configure other settings of an existing zone. To remove a zone, click *Delete*.

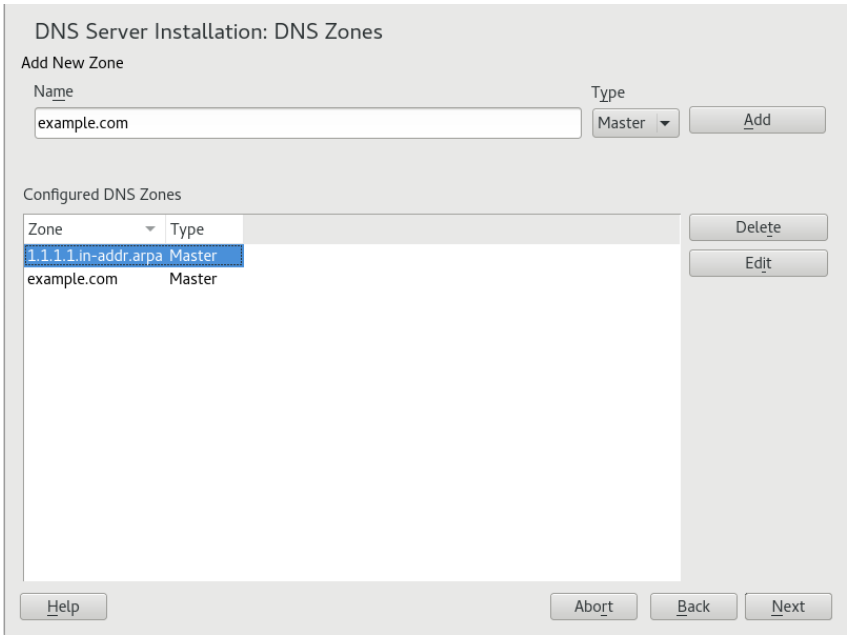


FIGURE 19.2: DNS SERVER INSTALLATION: DNS ZONES

3. In the final dialog, you can open the DNS port in the firewall by clicking *Open Port in Firewall*. Then decide whether to start the DNS server when booting (*On* or *Off*). You can also activate LDAP support. See *Figure 19.3, "DNS Server Installation: Finish Wizard"*.

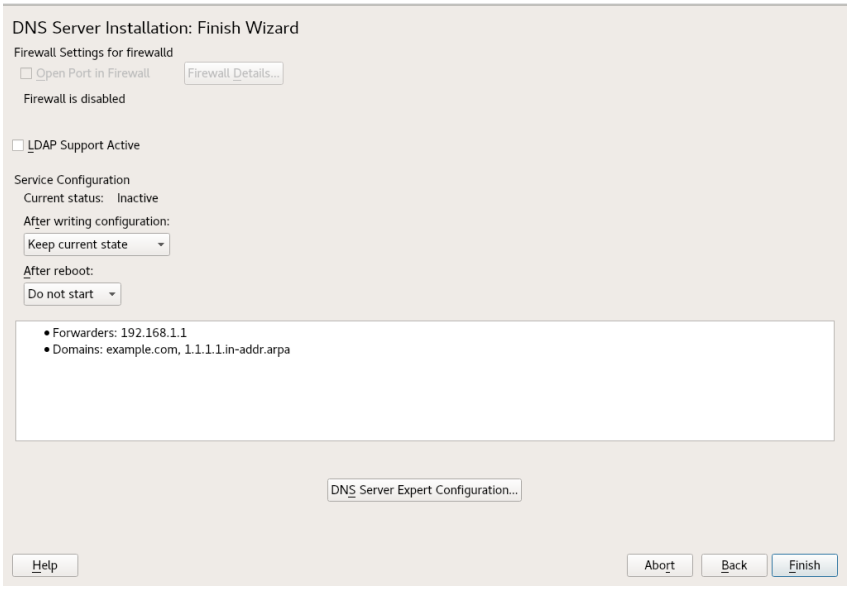


FIGURE 19.3: DNS SERVER INSTALLATION: FINISH WIZARD

19.3.2 Expert Configuration

After starting the module, YaST opens a window displaying several configuration options. Completing it results in a DNS server configuration with the basic functions in place:

19.3.2.1 Start-Up

Under *Start-Up*, define whether the DNS server should be started when the booting the system or manually. To start the DNS server immediately, click *Start DNS Server Now*. To stop the DNS server, click *Stop DNS Server Now*. To save the current settings, select *Save Settings and Reload DNS Server Now*. You can open the DNS port in the firewall with *Open Port in Firewall* and modify the firewall settings with *Firewall Details*.

By selecting *LDAP Support Active*, the zone files are managed by an LDAP database. Any changes to zone data written to the LDAP database are picked up by the DNS server when it is restarted or prompted to reload its configuration.

19.3.2.2 Forwarders

If your local DNS server cannot answer a request, it tries to forward the request to a *Forwarder*, if configured so. This forwarder may be added manually to the *Forwarder List*. If the forwarder is not static like in dial-up connections, *netconfig* handles the configuration. For more information about netconfig, see **man 8 netconfig**.

19.3.2.3 Basic Options

In this section, set basic server options. From the *Option* menu, select the desired item then specify the value in the corresponding text box. Include the new entry by selecting *Add*.

19.3.2.4 Logging

To set what the DNS server should log and how, select *Logging*. Under *Log Type*, specify where the DNS server should write the log data. Use the system-wide log by selecting *System Log* or specify a different file by selecting *File*. In the latter case, additionally specify a name, the maximum file size in megabytes and the number of log file versions to store.

Further options are available under *Additional Logging*. Enabling *Log All DNS Queries* causes *every* query to be logged, in which case the log file could grow extremely large. For this reason, it is not a good idea to enable this option for other than debugging purposes. To log the data traffic during zone updates between DHCP and DNS server, enable *Log Zone Updates*. To log the data traffic during a zone transfer from master to slave, enable *Log Zone Transfer*. See *Figure 19.4, "DNS Server: Logging"*.

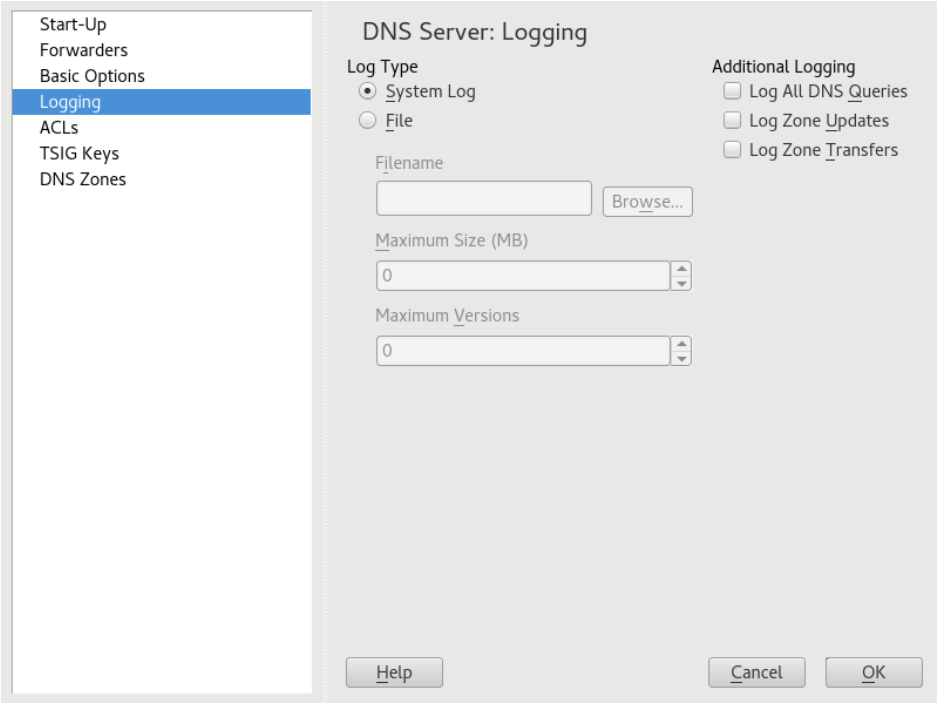


FIGURE 19.4: DNS SERVER: LOGGING

19.3.2.5 ACLs

Use this dialog to define ACLs (access control lists) to enforce access restrictions. After providing a distinct name under *Name*, specify an IP address (with or without netmask) under *Value* in the following fashion:

```
{ 192.168.1/24; }
```

The syntax of the configuration file requires that the address ends with a semicolon and is put into curly braces.

19.3.2.6 TSIG Keys

The main purpose of TSIGs (transaction signatures) is to secure communications between DHCP and DNS servers. They are described in *Section 19.8, "Secure Transactions"*.

To generate a TSIG key, enter a distinctive name in the field labeled *Key ID* and specify the file where the key should be stored (*Filename*). Confirm your choices with *Generate*.

To use a previously created key, leave the *Key ID* field blank and select the file where it is stored under *Filename*. After that, confirm with *Add*.

19.3.2.7 DNS Zones (Adding a Slave Zone)

To add a slave zone, select *DNS Zones*, choose the zone type *Slave*, write the name of the new zone, and click *Add*.

In the *Zone Editor* sub-dialog under *Master DNS Server IP*, specify the master from which the slave should pull its data. To limit access to the server, select one of the ACLs from the list.

19.3.2.8 DNS Zones (Adding a Master Zone)

To add a master zone, select *DNS Zones*, choose the zone type *Master*, write the name of the new zone, and click *Add*. When adding a master zone, a reverse zone is also needed. For example, when adding the zone `example.com` that points to hosts in a subnet `192.168.1.0/24`, you should also add a reverse zone for the IP-address range covered. By definition, this should be named `1.168.192.in-addr.arpa`.

19.3.2.9 DNS Zones (Editing a Master Zone)

To edit a master zone, select *DNS Zones*, select the master zone from the table, and click *Edit*. The dialog consists of several pages: *Basics* (the one opened first), *NS Records*, *MX Records*, *SOA*, and *Records*.

The basic dialog, shown in *Figure 19.5, "DNS Server: Zone Editor (Basics)"*, lets you define settings for dynamic DNS and access options for zone transfers to clients and slave name servers. To permit the dynamic updating of zones, select *Allow Dynamic Updates* and the corresponding TSIG key. The key must have been defined before the update action starts. To enable zone transfers, select the corresponding ACLs. ACLs must have been defined already.

In the *Basics* dialog, select whether to enable zone transfers. Use the listed ACLs to define who can download zones.

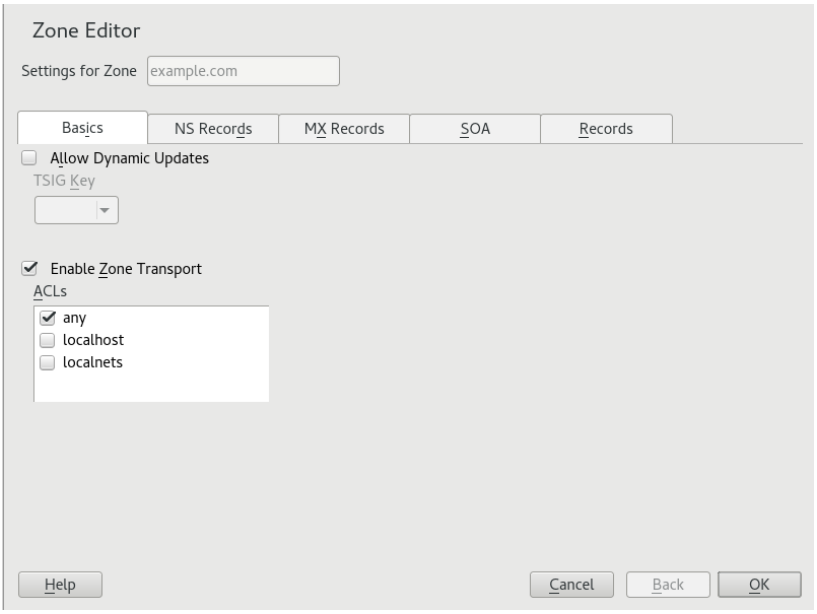


FIGURE 19.5: DNS SERVER: ZONE EDITOR (BASICS)

Zone Editor (NS Records)

The *NS Records* dialog allows you to define alternative name servers for the zones specified. Make sure that your own name server is included in the list. To add a record, enter its name under *Name Server to Add* then confirm with *Add*. See *Figure 19.6, "DNS Server: Zone Editor (NS Records)"*.

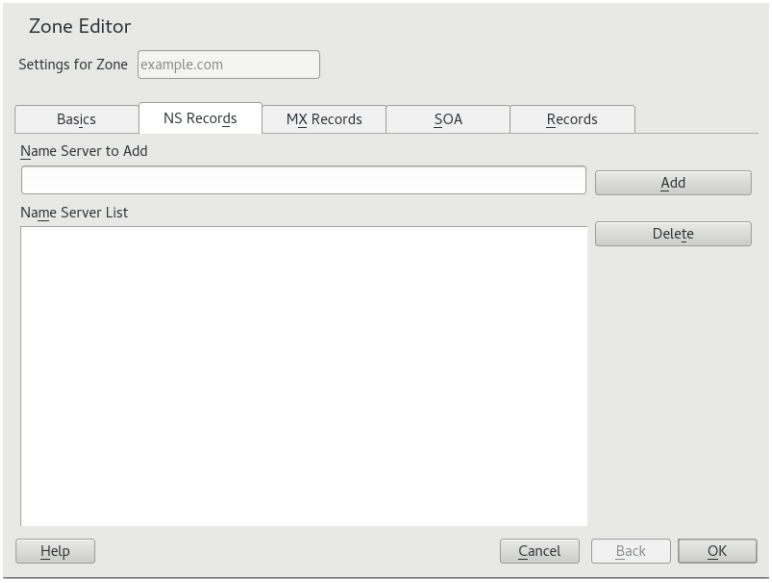


FIGURE 19.6: DNS SERVER: ZONE EDITOR (NS RECORDS)

Zone Editor (MX Records)

To add a mail server for the current zone to the existing list, enter the corresponding address and priority value. After doing so, confirm by selecting *Add*. See *Figure 19.7, "DNS Server: Zone Editor (MX Records)"*.

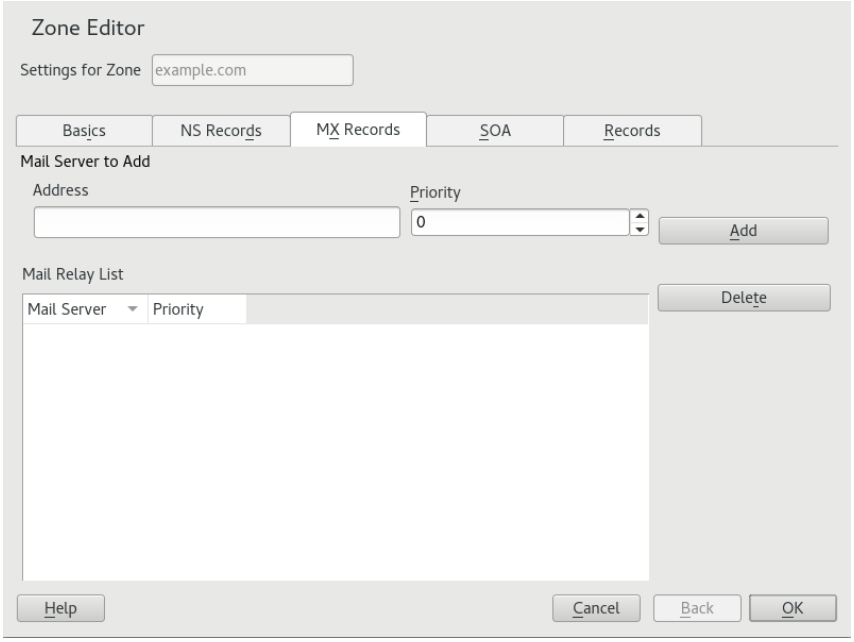


FIGURE 19.7: DNS SERVER: ZONE EDITOR (MX RECORDS)

Zone Editor (SOA)

This page allows you to create SOA (start of authority) records. For an explanation of the individual options, refer to *Example 19.6, "The /var/lib/named/example.com.zone File"*. Changing SOA records is not supported for dynamic zones managed via LDAP.

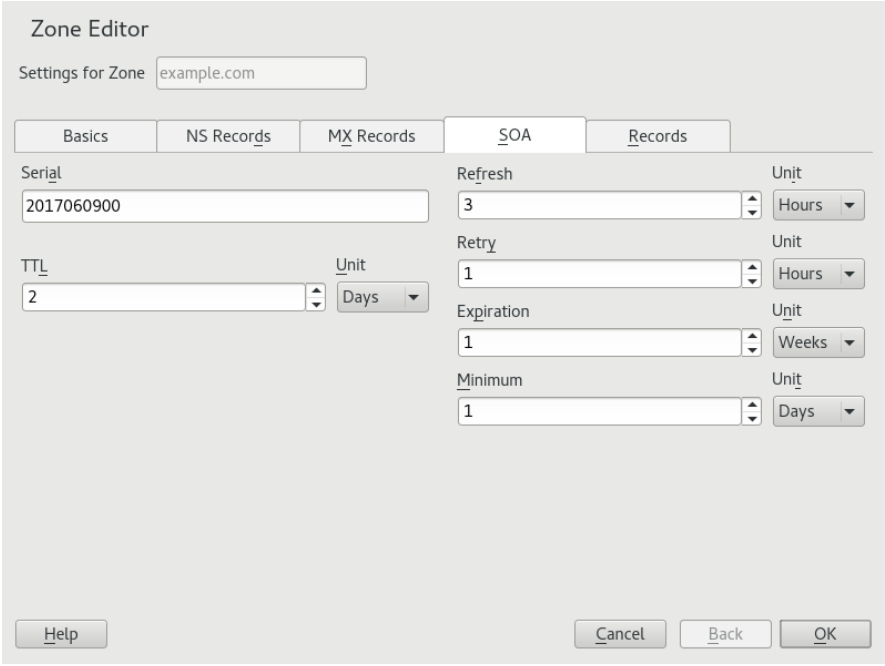


FIGURE 19.8: DNS SERVER: ZONE EDITOR (SOA)

Zone Editor (Records)

This dialog manages name resolution. In *Record Key*, enter the host name then select its type. The *A* type represents the main entry. The value for this should be an IP address (IPv4). Use *AAAA* for IPv6 addresses. *CNAME* is an alias. Use the types *NS* and *MX* for detailed or partial records that expand on the information provided in the *NS Records* and *MX Records* tabs. These three types resolve to an existing `A` record. *PTR* is for reverse zones. It is the opposite of an `A` record, for example:

```
hostname.example.com. IN A 192.168.0.1
1.0.168.192.in-addr.arpa IN PTR hostname.example.com.
```

19.3.2.9.1 Adding Reverse Zones

To add a reverse zone, follow this procedure:

1. Start *YaST* › *DNS Server* › *DNS Zones*.
2. If you have not added a master forward zone, add it and *Edit* it.
3. In the *Records* tab, fill the corresponding *Record Key* and *Value*, then add the record with *Add* and confirm with *OK*. If YaST complains about a non-existing record for a name server, add it in the *NS Records* tab.

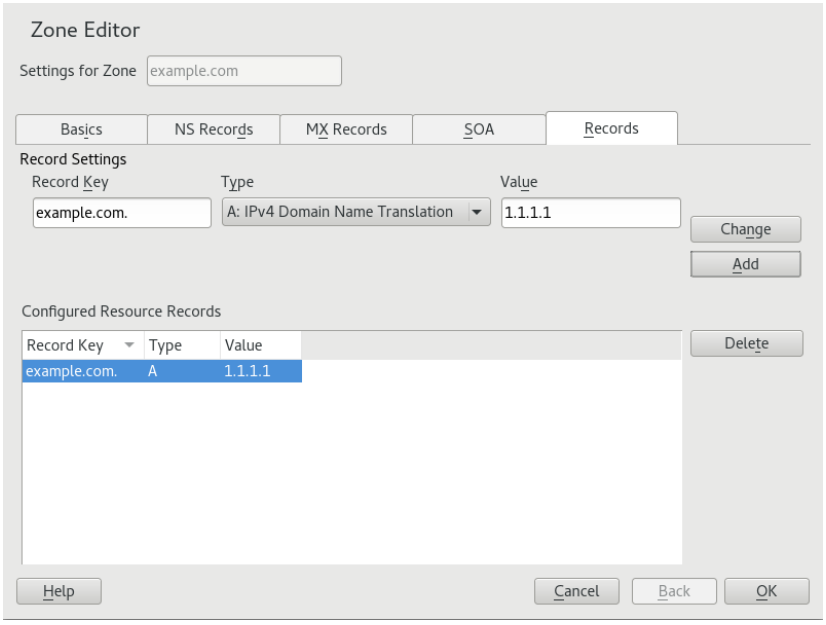


FIGURE 19.9: ADDING A RECORD FOR A MASTER ZONE

4. Back in the *DNS Zones* window, add a reverse master zone.

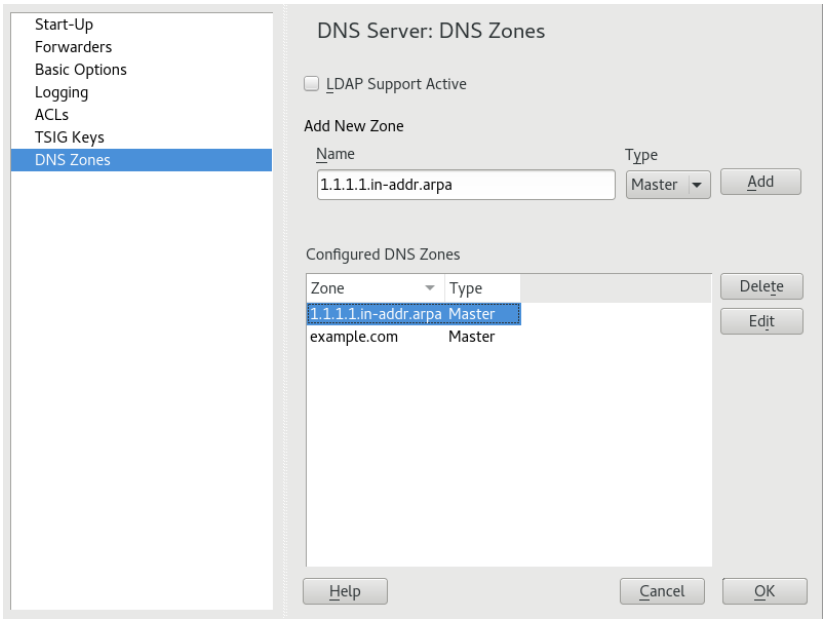


FIGURE 19.10: ADDING A REVERSE ZONE

5. *Edit* the reverse zone, and in the *Records* tab, you can see the *PTR: Reverse translation* record type. Add the corresponding *Record Key* and *Value*, then click *Add* and confirm with *OK*.

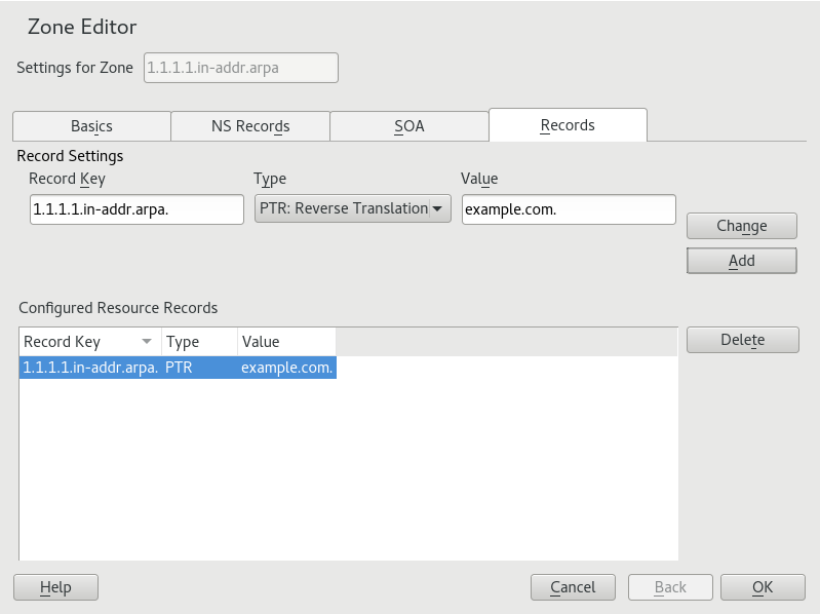


FIGURE 19.11: ADDING A REVERSE RECORD

Add a name server record if needed.

Tip: Editing the Reverse Zone

After adding a forward zone, go back to the main menu and select the reverse zone for editing. There in the tab *Basics* activate the check box *Automatically Generate Records From* and select your forward zone. That way, all changes to the forward zone are automatically updated in the reverse zone.

19.4 Starting the BIND Name Server

On a openSUSE® Leap system, the name server BIND (*Berkeley Internet Name Domain*) comes preconfigured, so it can be started right after installation without any problems. Normally, if you already have an Internet connection and entered `127.0.0.1` as the name server address for `localhost` in `/var/run/netconfig/resolv.conf`, you already have a working name resolution without needing to know the DNS of the provider. BIND carries out name resolution via the root name server, a notably slower process. Normally, the DNS of the provider should be entered with its IP address in the configuration file `/etc/named.conf` under `forwarders` to

ensure effective and secure name resolution. If this works so far, the name server runs as a pure *caching-only* name server. Only when you configure its own zones it becomes a proper DNS. Find a simple example documented in `/usr/share/doc/packages/bind/config`.

Tip: Automatic Adaptation of the Name Server Information

Depending on the type of Internet connection or the network connection, the name server information can automatically be adapted to the current conditions. To do this, set the `NETCONFIG_DNS_POLICY` variable in the `/etc/sysconfig/network/config` file to `auto`.

However, do not set up an official domain until one is assigned to you by the responsible institution. Even if you have your own domain and it is managed by the provider, you are better off not using it, because BIND would otherwise not forward requests for this domain. The Web server at the provider, for example, would not be accessible for this domain.

To start the name server, enter the command **`systemctl start named`** as `root`. Check with **`systemctl status named`** whether named (as the name server process is called) has been started successfully. Test the name server immediately on the local system with the **`host`** or **`dig`** programs, which should return `localhost` as the default server with the address `127.0.0.1`. If this is not the case, `/var/run/netconfig/resolv.conf` probably contains an incorrect name server entry or the file does not exist. For the first test, enter **`host`** `127.0.0.1`, which should always work. If you get an error message, use **`systemctl status named`** to see whether the server is actually running. If the name server does not start or behaves unexpectedly, check the output of **`journalctl -e`**.

To use the name server of the provider (or one already running on your network) as the forwarder, enter the corresponding IP address or addresses in the `options` section under `forwarders`. The addresses included in *Example 19.1, "Forwarding Options in named.conf"* are examples only. Adjust these entries to your own setup.

EXAMPLE 19.1: FORWARDING OPTIONS IN NAMED.CONF

```
options {
        directory "/var/lib/named";
        forwarders { 10.11.12.13; 10.11.12.14; };
        listen-on { 127.0.0.1; 192.168.1.116; };
        allow-query { 127/8; 192.168/16 };
        notify no;
```

```
};
```

The `options` entry is followed by entries for the zone, `localhost`, and `0.0.127.in-addr.arpa`. The `type hint` entry under "." should always be present. The corresponding files do not need to be modified and should work as they are. Also make sure that each entry is closed with a ";" and that the curly braces are in the correct places. After changing the configuration file `/etc/named.conf` or the zone files, tell BIND to reread them with **`systemctl reload named`**. Achieve the same by stopping and restarting the name server with **`systemctl restart named`**. Stop the server at any time by entering **`systemctl stop named`**.

19.5 The /etc/named.conf Configuration File

All the settings for the BIND name server itself are stored in the `/etc/named.conf` file. However, the zone data for the domains to handle (consisting of the host names, IP addresses, and so on) are stored in separate files in the `/var/lib/named` directory. The details of this are described later.

`/etc/named.conf` is roughly divided into two areas. One is the `options` section for general settings and the other consists of `zone` entries for the individual domains. A `logging` section and `acl` (access control list) entries are optional. Comment lines begin with a `#` sign or `//`. A minimal `/etc/named.conf` is shown in *Example 19.2, "A Basic /etc/named.conf"*.

EXAMPLE 19.2: A BASIC /ETC/NAMED.CONF

```
options {
        directory "/var/lib/named";
        forwarders { 10.0.0.1; };
        notify no;
};

zone "localhost" in {
       type master;
       file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" in {
        type master;
        file "127.0.0.zone";
};

zone "." in {
```

```
        type hint;
        file "root.hint";
};
```

19.5.1 Important Configuration Options

directory "*FILENAME*";

Specifies the directory in which BIND can find the files containing the zone data. Usually, this is `/var/lib/named`.

forwarders { *IP-ADDRESS*; };

Specifies the name servers (mostly of the provider) to which DNS requests should be forwarded if they cannot be resolved directly. Replace *IP-ADDRESS* with an IP address like `192.168.1.116`.

forward first;

Causes DNS requests to be forwarded before an attempt is made to resolve them via the root name servers. Instead of `forward first`, `forward only` can be written to have all requests forwarded and none sent to the root name servers. This makes sense for firewall configurations.

listen-on port 53 { 127.0.0.1; *IP-ADDRESS*; };

Tells BIND on which network interfaces and port to accept client queries. `port 53` does not need to be specified explicitly, because `53` is the default port. Enter `127.0.0.1` to permit requests from the local host. If you omit this entry entirely, all interfaces are used by default.

listen-on-v6 port 53 {any; };

Tells BIND on which port it should listen for IPv6 client requests. The only alternative to `any` is `none`. As far as IPv6 is concerned, the server only accepts wild card addresses.

query-source address * port 53;

This entry is necessary if a firewall is blocking outgoing DNS requests. This tells BIND to post requests externally from port 53 and not from any of the high ports above 1024.

query-source-v6 address * port 53;

Tells BIND which port to use for IPv6 queries.

allow-query { 127.0.0.1; *NET*; };

Defines the networks from which clients can post DNS requests. Replace *NET* with address information like `192.168.2.0/24`. The `/24` at the end is an abbreviated expression for the netmask (in this case `255.255.255.0`).

allow-transfer ! *;;

Controls which hosts can request zone transfers. In the example, such requests are completely denied with `! *`. Without this entry, zone transfers can be requested from anywhere without restrictions.

statistics-interval 0;

In the absence of this entry, BIND generates several lines of statistical information per hour in the system's journal. Set it to 0 to suppress these statistics completely or set an interval in minutes.

cleaning-interval 720;

This option defines at which time intervals BIND clears its cache. This triggers an entry in the system's journal each time it occurs. The time specification is in minutes. The default is 60 minutes.

interface-interval 0;

BIND regularly searches the network interfaces for new or nonexistent interfaces. If this value is set to `0`, this is not done and BIND only listens at the interfaces detected at start-up. Otherwise, the interval can be defined in minutes. The default is sixty minutes.

notify no;

`no` prevents other name servers from being informed when changes are made to the zone data or when the name server is restarted.

For a list of available options, read the manual page **`man 5 named.conf`**.

19.5.2 Logging

What, how, and where logging takes place can be extensively configured in BIND. Normally, the default settings should be sufficient. *Example 19.3, "Entry to Disable Logging"*, shows the simplest form of such an entry and completely suppresses any logging.

EXAMPLE 19.3: ENTRY TO DISABLE LOGGING

```
logging {
        category default { null; };
```

```
};
```

19.5.3 Zone Entries

EXAMPLE 19.4: ZONE ENTRY FOR EXAMPLE.COM

```
zone "example.com" in {
      type master;
      file "example.com.zone";
      notify no;
};
```

After `zone`, specify the name of the domain to administer (`example.com`) followed by `in` and a block of relevant options enclosed in curly braces, as shown in *Example 19.4, "Zone Entry for example.com"*. To define a *slave zone*, switch the `type` to `slave` and specify a name server that administers this zone as `master` (which, in turn, may be a slave of another master), as shown in *Example 19.5, "Zone Entry for example.net"*.

EXAMPLE 19.5: ZONE ENTRY FOR EXAMPLE.NET

```
zone "example.net" in {
      type slave;
      file "slave/example.net.zone";
      masters { 10.0.0.1; };
};
```

The zone options:

type master;

By specifying `master`, tell BIND that the zone is handled by the local name server. This assumes that a zone file has been created in the correct format.

type slave;

This zone is transferred from another name server. It must be used together with `masters`.

type hint;

The zone `.` of the `hint` type is used to set the root name servers. This zone definition can be left as is.

file `example.com.zone` or file "slave/example.net.zone";

This entry specifies the file where zone data for the domain is located. This file is not required for a slave, because this data is pulled from another name server. To differentiate master and slave files, use the directory `slave` for the slave files.

masters { *SERVER_IP_ADDRESS*; };

This entry is only needed for slave zones. It specifies from which name server the zone file should be transferred.

allow-update {! *; };

This option controls external write access, which would allow clients to make a DNS entry —something not normally desirable for security reasons. Without this entry, zone updates are not allowed. The above entry achieves the same because `! *` effectively bans any such activity.

19.6 Zone Files

Two types of zone files are needed. One assigns IP addresses to host names and the other does the reverse: it supplies a host name for an IP address.

Tip: Using the Dot (Period, Fullstop) in Zone Files

The `"."` has an important meaning in the zone files. If host names are given without a final dot (`.`), the zone is appended. Complete host names specified with a full domain name must end with a dot (`.`) to avoid having the domain added to it again. A missing or wrongly placed "." is probably the most frequent cause of name server configuration errors.

The first case to consider is the zone file `example.com.zone`, responsible for the domain `example.com`, shown in *Example 19.6, "The /var/lib/named/example.com.zone File"*.

EXAMPLE 19.6: THE /VAR/LIB/NAMED/EXAMPLE.COM.ZONE FILE

```
$TTL 2D ❶
example.com. IN SOA      dns  root.example.com. ( ❷
             2003072441  ; serial ❸
             1D          ; refresh ❹
             2H          ; retry ❺
             1W          ; expiry ❻
```

```
                2D )        ; minimum 7

                IN NS       dns 8
                IN MX       10 mail dns 9
gate            IN A        192.168.5.1 10
                IN A        10.0.0.1
dns             IN A        192.168.1.116
mail            IN A        192.168.3.108
jupiter         IN A        192.168.2.100
venus           IN A        192.168.2.101
saturn          IN A        192.168.2.102
mercury         IN A        192.168.2.103
ntp             IN CNAME    dns 11
dns6            IN A6  0    2002:c0a8:174::
```

1. `$TTL` defines the default time to live that should apply to all the entries in this file. In this example, entries are valid for a period of two days (`2 D`).
2. This is where the SOA (start of authority) control record begins:
   - The name of the domain to administer is `example.com` in the first position. This ends with `"."`, because otherwise the zone would be appended a second time. Alternatively, `@` can be entered here, in which case the zone would be extracted from the corresponding entry in `/etc/named.conf`.
   - After `IN SOA` is the name of the name server in charge as master for this zone. The name is expanded from `dns` to `dns.example.com`, because it does not end with a `"."`.
   - An e-mail address of the person in charge of this name server follows. Because the `@` sign already has a special meaning, `"."` is entered here instead. For `root@example.com` the entry must read `root.example.com.`. The `"."` must be included at the end to prevent the zone from being added.
   - The `(` includes all lines up to `)` into the SOA record.
3. The `serial number` is a 10 digit number. It must be changed each time this file is changed. It is needed to inform the secondary name servers (slave servers) of changes. For this, a 10 digit number of the date and run number, written as YYYYMMDDNN, has become the customary format (YYYY = year, MM = month and DD = day. NN is a sequence number in case you update it more than once on the given day).
4. The `refresh rate` specifies the time interval at which the secondary name servers verify the zone `serial number`. In this case, one day.

5 The `retry rate` specifies the time interval at which a secondary name server, in case of error, attempts to contact the primary server again. Here, two hours.

6 The `expiration time` specifies the time frame after which a secondary name server discards the cached data if it has not regained contact to the primary server. Here, a week.

7 The last entry in the SOA record specifies the `negative caching TTL`—the time for which results of unresolved DNS queries from other servers may be cached.

8 The `IN NS` specifies the name server responsible for this domain. `dns` is extended to `dns.example.com` because it does not end with a `"."`. There can be several lines like this—one for the primary and one for each secondary name server. If `notify` is not set to `no` in `/etc/named.conf`, all the name servers listed here are informed of the changes made to the zone data.

9 The MX record specifies the mail server that accepts, processes, and forwards e-mails for the domain `example.com`. In this example, this is the host `mail.example.com`. The number in front of the host name is the preference value. If there are multiple MX entries, the mail server with the smallest value is taken first. If mail delivery to this server fails, the entry with the next-smallest value is used.

10 This and the following lines are the actual address records where one or more IP addresses are assigned to host names. The names are listed here without a `"."` because they do not include their domain, so `example.com` is added to all of them. Two IP addresses are assigned to the host `gate`, as it has two network cards. Wherever the host address is a traditional one (IPv4), the record is marked with `A`. If the address is an IPv6 address, the entry is marked with `AAAA`.

Note: IPv6 Syntax

The IPv6 record has a slightly different syntax than IPv4. Because of the fragmentation possibility, it is necessary to provide information about missed bits before the address. To fill up the IPv6 address with the needed number of “0”, add two colons at the correct place in the address.

```
pluto     AAAA 2345:00C1:CA11::1234:5678:9ABC:DEF0
pluto     AAAA 2345:00D2:DA11::1234:5678:9ABC:DEF0
```

11 The alias `ntp` can be used to address `dns` (`CNAME` means *canonical name*).

The pseudo domain `in-addr.arpa` is used for the reverse lookup of IP addresses into host names. It is appended to the network part of the address in reverse notation. So `192.168` is resolved into `168.192.in-addr.arpa`. See *Example 19.7, "Reverse Lookup"*.

EXAMPLE 19.7: REVERSE LOOKUP

```
$TTL 2D ❶
168.192.in-addr.arpa.   IN SOA dns.example.com. root.example.com. ( ❷
                        2003072441      ; serial
                        1D              ; refresh
                        2H              ; retry
                        1W              ; expiry
                        2D )            ; minimum

                        IN NS           dns.example.com. ❸

1.5                     IN PTR          gate.example.com. ❹
100.3                   IN PTR          www.example.com.
253.2                   IN PTR          cups.example.com.
```

❶ $TTL defines the standard TTL that applies to all entries here.

❷ The configuration file should activate reverse lookup for the network `192.168`. Given that the zone is called `168.192.in-addr.arpa`, it should not be added to the host names. Therefore, all host names are entered in their complete form—with their domain and with a `"."` at the end. The remaining entries correspond to those described for the previous `example.com` example.
See *Example 19.6, "The /var/lib/named/example.com.zone File"* for detail on the entries within this record.

❸ 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.

❹ This, and the following lines, are the pointer records hinting at the IP addresses on the respective hosts. Only the last part of the IP address is entered at the beginning of the line, without the `"."` at the end. Appending the zone to this (without the `.in-addr.arpa`) results in the complete IP address in reverse order.

Normally, zone transfers between different versions of BIND should be possible without any problems.

19.7 Dynamic Update of Zone Data

The term *dynamic update* refers to operations by which entries in the zone files of a master server are added, changed, or deleted. This mechanism is described in RFC 2136. Dynamic update is configured individually for each zone entry by adding an optional `allow-update` or `update-policy` rule. Zones to update dynamically should not be edited by hand.

Transmit the entries to update to the server with the command **`nsupdate`**. For the exact syntax of this command, check the manual page for nsupdate (**`man 8 nsupdate`**). For security reasons, any such update should be performed using TSIG keys as described in *Section 19.8, "Secure Transactions"*.

19.8 Secure Transactions

Secure transactions can be made with transaction signatures (TSIGs) based on shared secret keys (also called TSIG keys). This section describes how to generate and use such keys.

Secure transactions are needed for communication between different servers and for the dynamic update of zone data. Making the access control dependent on keys is much more secure than merely relying on IP addresses.

Generate a TSIG key with the following command (for details, see **`man`** `dnssec-keygen`):

```
tux > sudo dnssec-keygen -a hmac-md5 -b 128 -n HOST host1-host2
```

This creates two files with names similar to these:

```
Khost1-host2.+157+34265.private Khost1-host2.+157+34265.key
```

The key itself (a string like `ejIkuCyyGJwwuN3xAteKgg==`) is found in both files. To use it for transactions, the second file (`Khost1-host2.+157+34265.key`) must be transferred to the remote host, preferably in a secure way (using scp, for example). On the remote server, the key must be included in the `/etc/named.conf` file to enable a secure communication between `host1` and `host2`:

```
key host1-host2 {
 algorithm hmac-md5;
 secret "ejIkuCyyGJwwuN3xAteKgg==";
};
```

Warning: File Permissions of `/etc/named.conf`

Make sure that the permissions of `/etc/named.conf` are properly restricted. The default for this file is `0640`, with the owner being `root` and the group `named`. As an alternative, move the keys to an extra file with specially limited permissions, which is then included from `/etc/named.conf`. To include an external file, use:

```
include  "filename"
```

Replace `filename` with an absolute path to your file with keys.

To enable the server `host1` to use the key for `host2` (which has the address `10.1.2.3` in this example), the server's `/etc/named.conf` must include the following rule:

```
server 10.1.2.3 {
  keys { host1-host2. ;};
};
```

Analogous entries must be included in the configuration files of `host2`.

Add TSIG keys for any ACLs (access control lists, not to be confused with file system ACLs) that are defined for IP addresses and address ranges to enable transaction security. The corresponding entry could look like this:

```
allow-update { key host1-host2. ;};
```

This topic is discussed in more detail in the *BIND Administrator Reference Manual* under `update-policy`.

19.9 DNS Security

DNSSEC, or DNS security, is described in RFC 2535. The tools available for DNSSEC are discussed in the BIND Manual.

A zone considered secure must have one or several zone keys associated with it. These are generated with **`dnssec-keygen`**, as are the host keys. The DSA encryption algorithm is currently used to generate these keys. The public keys generated should be included in the corresponding zone file with an `$INCLUDE` rule.

With the command **`dnssec-signzone`**, you can create sets of generated keys (`keyset-` files), transfer them to the parent zone in a secure manner, and sign them. This generates the files to include for each zone in `/etc/named.conf`.

19.10 For More Information

For more information, see the *BIND Administrator Reference Manual* from the `bind-doc` package, which is installed under `/usr/share/doc/packages/bind/arm`. Consider additionally consulting the RFCs referenced by the manual and the manual pages included with BIND. `/usr/share/doc/packages/bind/README.SUSE` contains up-to-date information about BIND in openSUSE Leap.

20 DHCP

The purpose of the *Dynamic Host Configuration Protocol* (DHCP) is to assign network settings centrally (from a server) rather than configuring them locally on every workstation. A host configured to use DHCP does not have control over its own static address. It is enabled to configure itself completely and automatically according to directions from the server. If you use the NetworkManager on the client side, you do not need to configure the client. This is useful if you have changing environments and only one interface active at a time. Never use NetworkManager on a machine that runs a DHCP server.

One way to configure a DHCP server is to identify each client using the hardware address of its network card (which should usually be fixed), then supply that client with identical settings each time it connects to the server. DHCP can also be configured to assign addresses to each relevant client dynamically from an address pool set up for this purpose. In the latter case, the DHCP server tries to assign the same address to the client each time it receives a request, even over extended periods. This works only if the network does not have more clients than addresses.

DHCP makes life easier for system administrators. Any changes, even bigger ones, related to addresses and the network configuration in general can be implemented centrally by editing the server's configuration file. This is much more convenient than reconfiguring numerous workstations. It is also much easier to integrate machines, particularly new machines, into the network, because they can be given an IP address from the pool. Retrieving the appropriate network settings from a DHCP server is especially useful in case of laptops regularly used in different networks.

In this chapter, the DHCP server will run in the same subnet as the workstations, `192.168.2.0/24` with `192.168.2.1` as gateway. It has the fixed IP address `192.168.2.254` and serves two address ranges, `192.168.2.10` to `192.168.2.20` and `192.168.2.100` to `192.168.2.200`.

A DHCP server supplies not only the IP address and the netmask, but also the host name, domain name, gateway, and name server addresses for the client to use. In addition to that, DHCP allows several parameters to be configured in a centralized way, for example, a time server from which clients may poll the current time or even a print server.

20.1 Configuring a DHCP Server with YaST

To install a DHCP server, start YaST and select *Software* › *Software Management*. Choose *Filter* › *Patterns* and select *DHCP and DNS Server*. Confirm the installation of the dependent packages to finish the installation process.

Important: LDAP Support

The YaST DHCP module can be set up to store the server configuration locally (on the host that runs the DHCP server) or to have its configuration data managed by an LDAP server. To use LDAP, set up your LDAP environment before configuring the DHCP server.

For more information about LDAP, see *Book "Security 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* and then *Next*. See *Figure 20.1, "DHCP Server: Card Selection"*.

Note: DHCP and `firewalld`

Please note that the option *Open Firewall for Selected Interfaces* does not (yet) support `firewalld` in openSUSE Leap 15.1. To manually open the DHCP port, run

```
tux > sudo firewall-cmd --zone=public --permanent --add-service=dhcp
tux > sudo firewall-cmd --reload
```

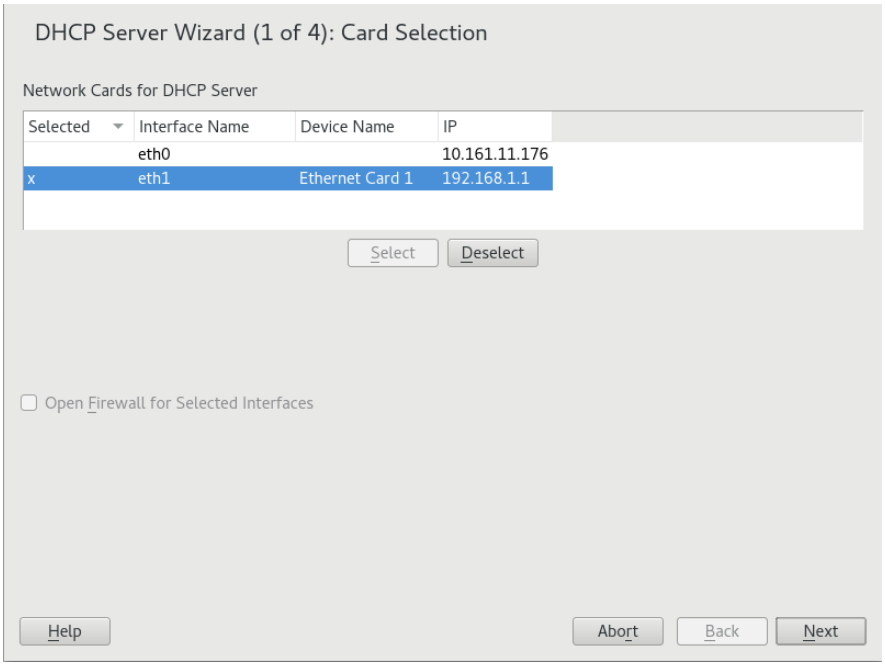


FIGURE 20.1: DHCP SERVER: CARD SELECTION

2. Use the check box to determine whether your DHCP settings should be automatically stored by an LDAP server. In the text boxes, provide the network specifics for all clients the DHCP server should manage. These specifics are the domain name, address of a time server, addresses of the primary and secondary name server, addresses of a print and a WINS server (for a mixed network with both Windows and Linux clients), gateway address, and lease time. See *Figure 20.2, "DHCP Server: Global Settings"*.

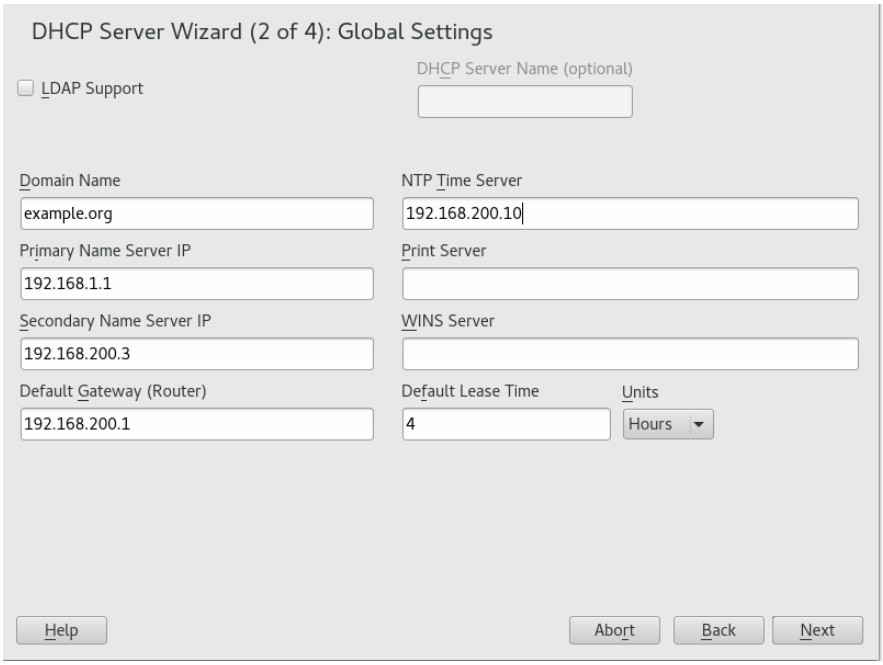


FIGURE 20.2: DHCP SERVER: GLOBAL SETTINGS

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"*.

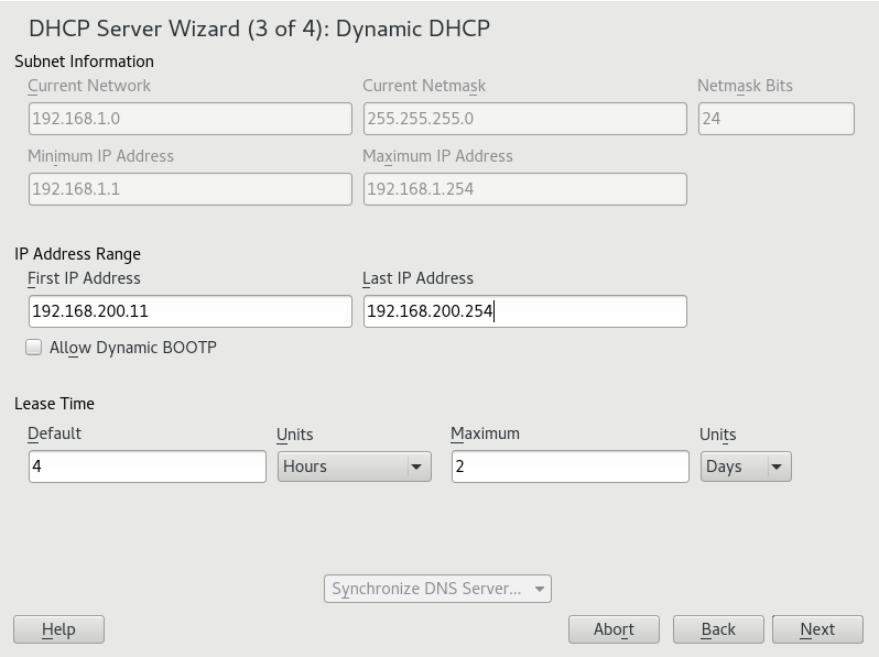


FIGURE 20.3: DHCP SERVER: DYNAMIC DHCP

4. Define how the DHCP server should be started. Specify whether to start the DHCP server automatically when the system is booted or manually when needed (for example, for testing purposes). Click *Finish* to complete the configuration of the server. See *Figure 20.4, "DHCP Server: Start-Up"*.

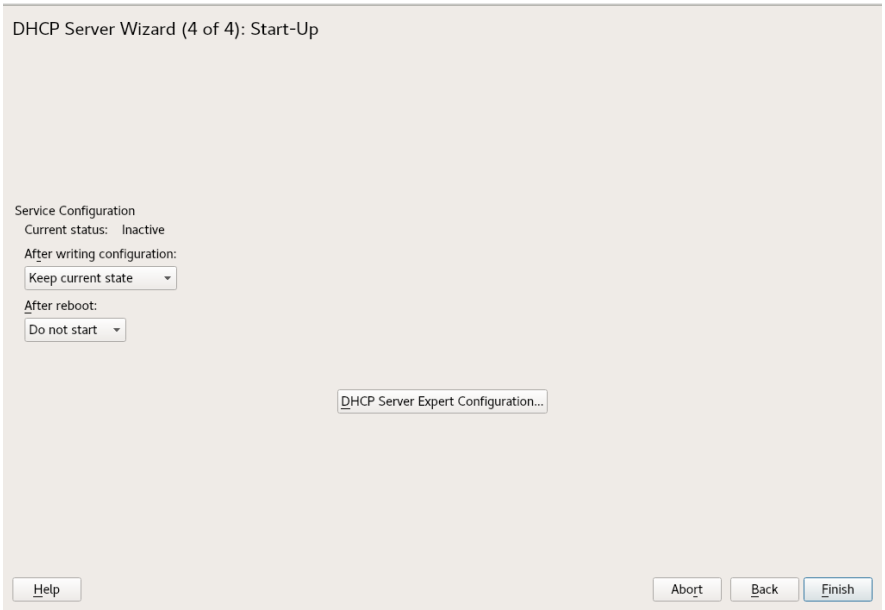


FIGURE 20.4: DHCP SERVER: START-UP

5. Instead of using dynamic DHCP in the way described in the preceding steps, you can also configure the server to assign addresses in quasi-static fashion. Use the text boxes provided in the lower part to specify a list of the clients to manage in this way. Specifically, provide the *Name* and the *IP Address* to give to such a client, the *Hardware Address*, and the *Network Type* (token ring or Ethernet). Modify the list of clients, which is shown in the upper part with *Add*, *Edit*, and *Delete from List*. See *Figure 20.5, "DHCP Server: Host Management"*.

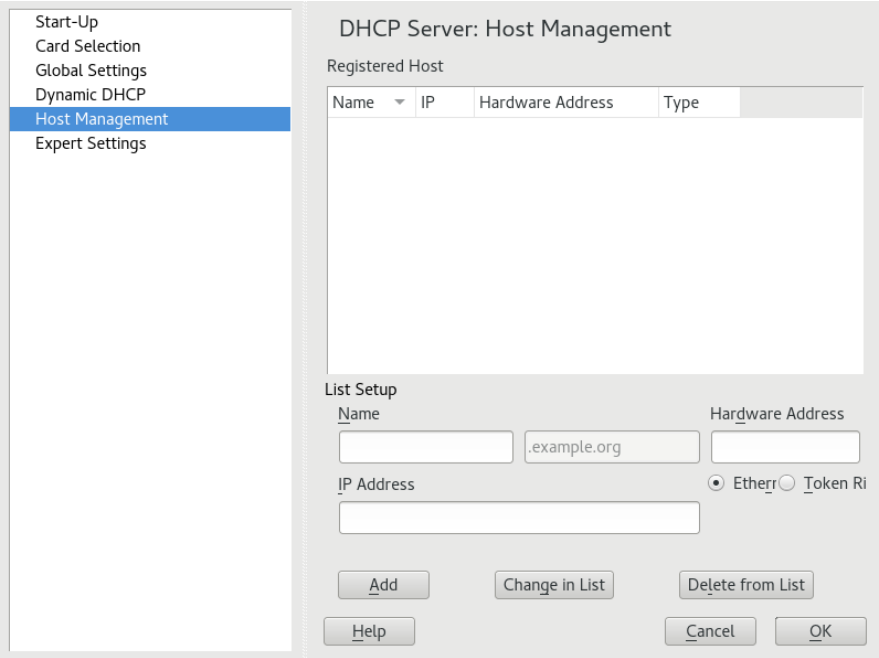


FIGURE 20.5: DHCP SERVER: HOST MANAGEMENT

20.1.2 DHCP Server Configuration (Expert)

In addition to the configuration method discussed earlier, there is also an expert configuration mode that allows you to change the DHCP server setup in every detail. Start the expert configuration by clicking *DHCP Server Expert Configuration* in the *Start-Up* dialog (see *Figure 20.4, "DHCP Server: Start-Up"*).

Chroot Environment and Declarations

In this first dialog, make the existing configuration editable by selecting *Start DHCP Server*. An important feature of the behavior of the DHCP server is its ability to run in a chroot environment, or chroot jail, to secure the server host. If the DHCP server should ever be compromised by an outside attack, the attacker will still be in the chroot jail, which prevents them from accessing the rest of the system. The lower part of the dialog displays

a tree view with the declarations that have already been defined. Modify these with *Add*, *Delete*, and *Edit*. Selecting *Advanced* takes you to additional expert dialogs. See *Figure 20.6, "DHCP Server: Chroot Jail and Declarations"*. After selecting *Add*, define the type of declaration to add. With *Advanced*, view the log file of the server, configure TSIG key management, and adjust the configuration of the firewall according to the setup of the DHCP server.

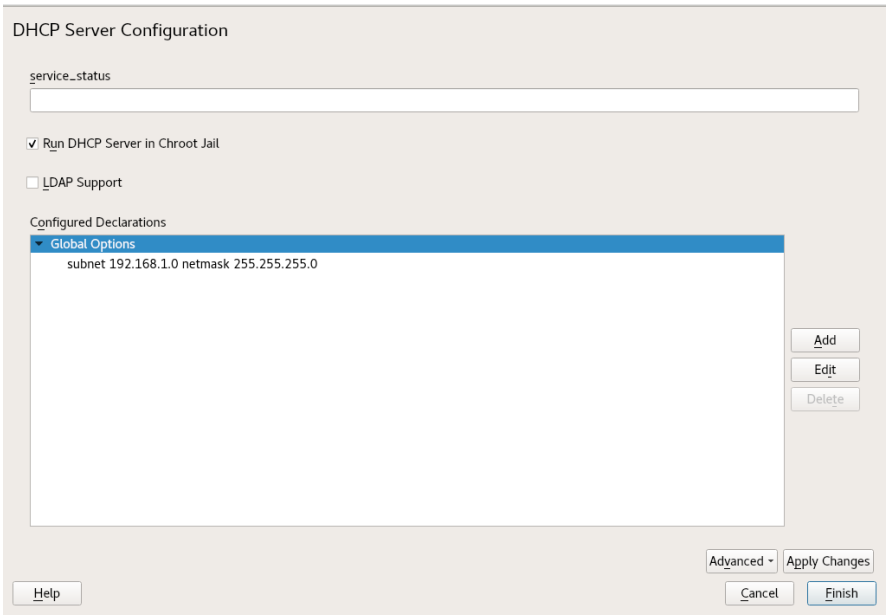


FIGURE 20.6: DHCP SERVER: CHROOT JAIL AND DECLARATIONS

Selecting the Declaration Type

The *Global Options* of the DHCP server are made up of several declarations. This dialog lets you set the declaration types *Subnet*, *Host*, *Shared Network*, *Group*, *Pool of Addresses*, and *Class*. This example shows the selection of a new subnet (see *Figure 20.7, "DHCP Server: Selecting a Declaration Type"*).

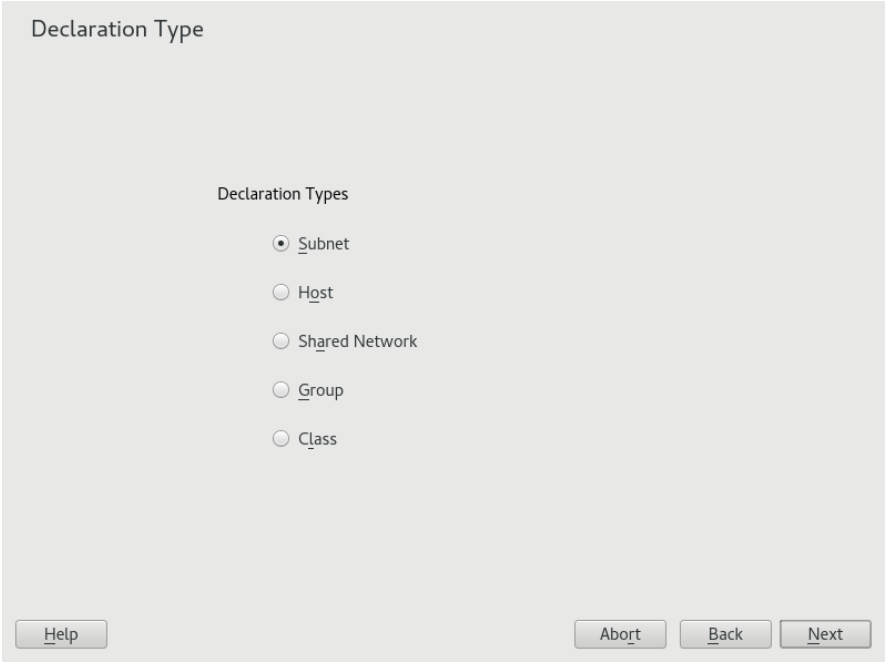


FIGURE 20.7: DHCP SERVER: SELECTING A DECLARATION TYPE

Subnet Configuration

This dialog allows you specify a new subnet with its IP address and netmask. In the middle part of the dialog, modify the DHCP server start options for the selected subnet using *Add*, *Edit*, and *Delete*. To set up dynamic DNS for the subnet, select *Dynamic DNS*.

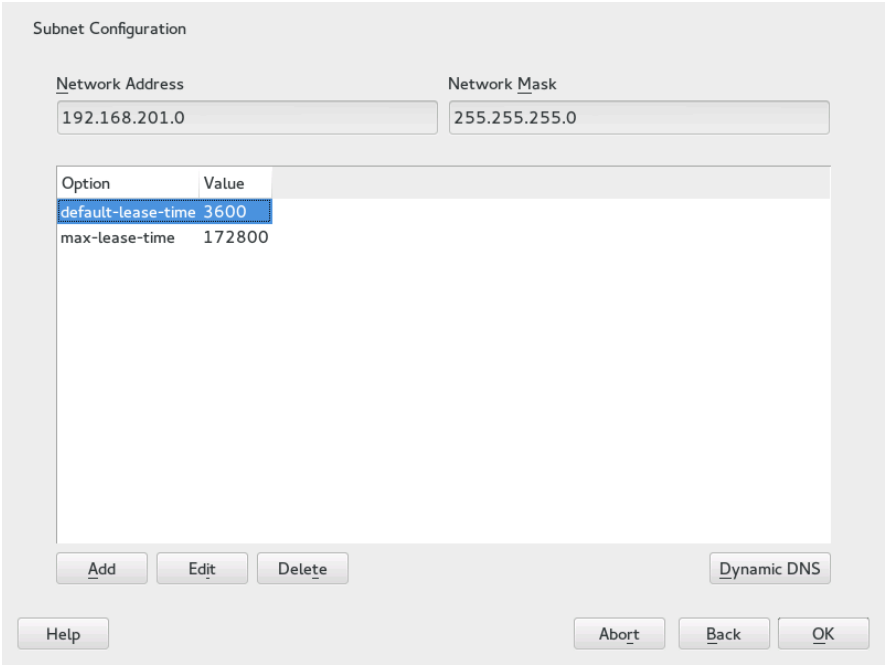


FIGURE 20.8: DHCP SERVER: CONFIGURING SUBNETS

TSIG Key Management

If you chose to configure dynamic DNS in the previous dialog, you can now configure the key management for a secure zone transfer. Selecting *OK* takes you to another dialog in which to configure the interface for dynamic DNS (see *Figure 20.10, "DHCP Server: Interface Configuration for Dynamic DNS"*).

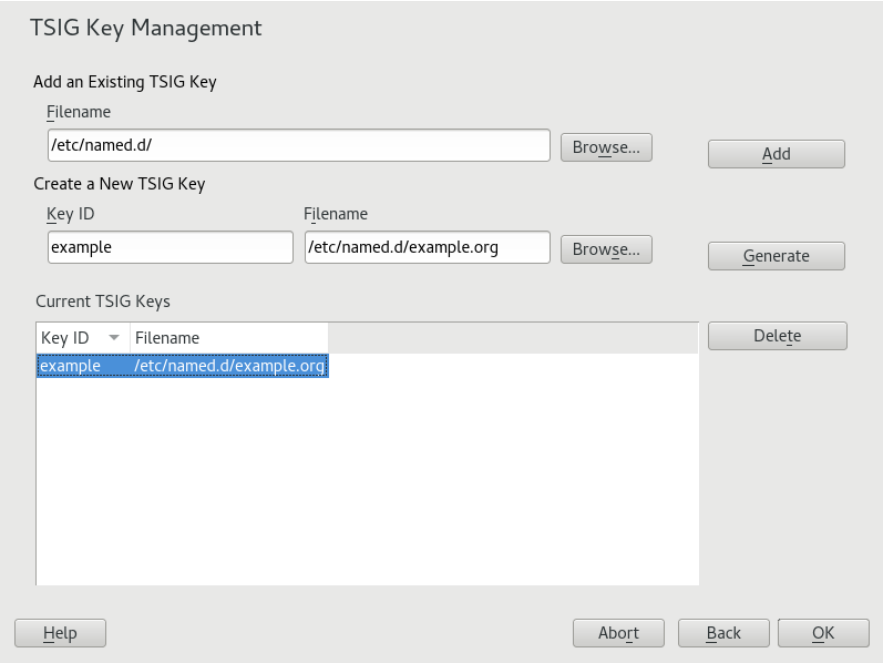


FIGURE 20.9: DHCP SERVER: TSIG CONFIGURATION

Dynamic DNS: Interface Configuration

You can now activate dynamic DNS for the subnet by selecting *Enable Dynamic DNS for This Subnet*. After doing so, use the drop-down box to activate the TSIG keys for forward and reverse zones, making sure that the keys are the same for the DNS and the DHCP server. With *Update Global Dynamic DNS Settings*, enable the automatic update and adjustment of the global DHCP server settings according to the dynamic DNS environment. Finally, define which forward and reverse zones should be updated per dynamic DNS, specifying the name of the primary name server for each of the two zones. Selecting *OK* returns to the subnet configuration dialog (see *Figure 20.8, "DHCP Server: Configuring Subnets"*). Selecting *OK* again returns to the original expert configuration dialog.

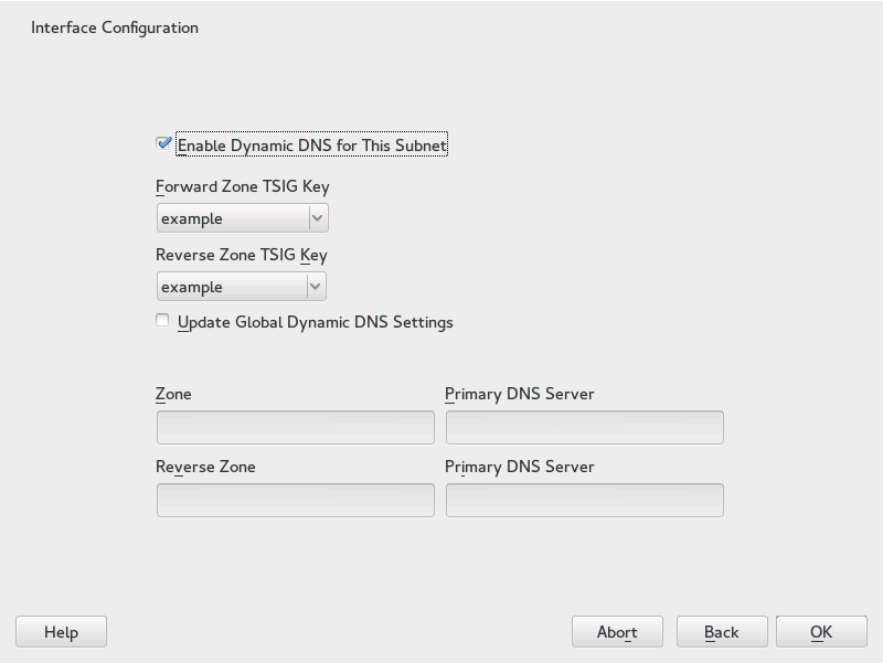


FIGURE 20.10: DHCP SERVER: INTERFACE CONFIGURATION FOR DYNAMIC DNS

Network Interface Configuration

To define the interfaces the DHCP server should listen to and to adjust the firewall configuration, select *Advanced* › *Interface Configuration* from the expert configuration dialog. From the list of interfaces displayed, select one or more that should be attended by the DHCP server. If clients in all subnets need to be able to communicate with the server and the server host also runs a firewall, adjust the firewall accordingly.

Note: DHCP and `firewalld`

Please note that the option *Open Firewall for Selected Interfaces* does not (yet) support `firewalld` in openSUSE Leap 15.1. To manually open the DHCP port, run

```
tux > sudo firewall-cmd --zone=public --permanent --add-service=dhcp
tux > sudo firewall-cmd --reload
```

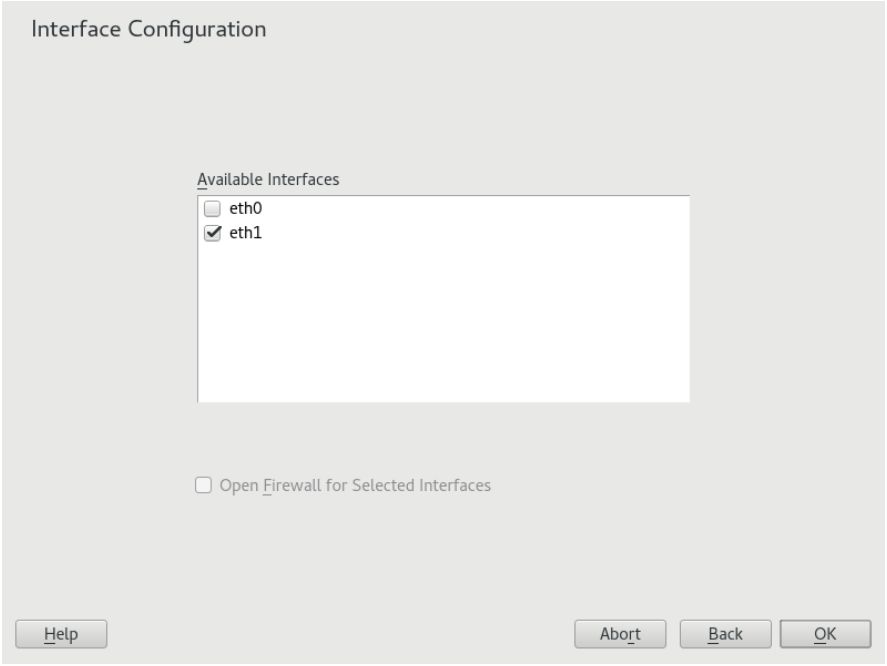


FIGURE 20.11: DHCP SERVER: NETWORK INTERFACE AND FIREWALL

After completing all configuration steps, close the dialog with *OK*. The server is now started with its new configuration.

20.2 DHCP Software Packages

Both the DHCP server and the DHCP clients are available for openSUSE Leap. The DHCP server available is `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 **`rcdhcpd`** `check-syntax` to perform a brief syntax check. If you encounter any unexpected problems with your configuration (the server aborts with an error or does not return `done` on start), you should be able to find out what has gone wrong by looking for information either in the main system log that can be queried with the command **`journalctl`** (see *Chapter 11, `journalctl`: Query the `systemd` Journal* for more information).

On a default openSUSE Leap system, the DHCP daemon is started in a chroot environment for security reasons. The configuration files must be copied to the chroot environment so the daemon can find them. Normally, there is no need to worry about this because the command **`systemctl start dhcpd`** automatically copies the files.

20.3.1 Clients with Fixed IP Addresses

DHCP can also be used to assign a predefined, static address to a specific client. Addresses assigned explicitly always take priority over dynamic addresses from the pool. A static address never expires in the way a dynamic address would, for example, if there were not enough addresses available and the server needed to redistribute them among clients.

To identify a client configured with a static address, dhcpd uses the hardware address (which is a globally unique, fixed numerical code consisting of six octet pairs) for the identification of all network devices (for example, `00:30:6E:08:EC:80`). If the respective lines, like the ones in

Example 20.2, "Additions to the Configuration File", are added to the configuration file of *Example 20.1, "The Configuration File /etc/dhcpd.conf"*, the DHCP daemon always assigns the same set of data to the corresponding client.

EXAMPLE 20.2: ADDITIONS TO THE CONFIGURATION FILE

```
host jupiter {
hardware ethernet 00:30:6E:08:EC:80;
fixed-address 192.168.2.100;
}
```

The name of the respective client (`host` *`HOSTNAME`*, here `jupiter`) is entered in the first line and the MAC address in the second line. On Linux hosts, find the MAC address with the command `ip link show` followed by the network device (for example, `eth0`). The output should contain something like

```
link/ether 00:30:6E:08:EC:80
```

In the preceding example, a client with a network card having the MAC address `00:30:6E:08:EC:80` is assigned the IP address `192.168.2.100` and the host name `jupiter` automatically. The type of hardware to enter is `ethernet` in nearly all cases, although `token-ring`, which is often found on IBM systems, is also supported.

20.3.2 The openSUSE Leap Version

To improve security, the openSUSE Leap version of the ISC's DHCP server comes with the non-root/chroot patch by Ari Edelkind applied. This enables dhcpd to run with the user ID `nobody` and run in a chroot environment (`/var/lib/dhcp`). To make this possible, the configuration file `dhcpd.conf` must be located in `/var/lib/dhcp/etc`. The init script automatically copies the file to this directory when starting.

Control the server's behavior regarding this feature by means of entries in the file `/etc/sysconfig/dhcpd`. To run dhcpd without the chroot environment, set the variable `DHCPD_RUN_CHROOTED` in `/etc/sysconfig/dhcpd` to "no".

To enable dhcpd to resolve host names even from within the chroot environment, some other configuration files must be copied as well:

- `/etc/localtime`
- `/etc/host.conf`

- `/etc/hosts`
- `/var/run/netconfig/resolv.conf`

These files are copied to `/var/lib/dhcp/etc/` when starting the init script. Take these copies into account for any changes that they require if they are dynamically modified by scripts like `/etc/ppp/ip-up`. However, there should be no need to worry about this if the configuration file only specifies IP addresses (instead of host names).

If your configuration includes additional files that should be copied into the chroot environment, set these under the variable `DHCPD_CONF_INCLUDE_FILES` in the file `/etc/sysconfig/dhcpd`. To ensure that the DHCP logging facility keeps working even after a restart of the syslog daemon, there is an additional entry `SYSLOGD_ADDITIONAL_SOCKET_DHCP` in the file `/etc/sysconfig/syslog`.

20.4 For More Information

More information about DHCP is available at the Web site of the *Internet Systems Consortium* (https://www.isc.org/blogs/category/dhcp/). Information is also available in the `dhcpd`, `dhcpd.conf`, `dhcpd.leases`, and `dhcp-options` man pages.

21 Samba

Using Samba, a Unix machine can be configured as a file and print server for macOS, Windows, and OS/2 machines. Samba has developed into a fully-fledged and rather complex product. Configure Samba with YaST, or by editing the configuration file manually.

21.1 Terminology

The following are some terms used in Samba documentation and in the YaST module.

SMB protocol

Samba uses the SMB (server message block) protocol that is based on the NetBIOS services. Microsoft released the protocol so other software manufacturers could establish connections to a Microsoft domain network. With Samba, the SMB protocol works on top of the TCP/IP protocol, so the TCP/IP protocol must be installed on all clients.

CIFS protocol

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. Common operating systems, such as Windows and macOS support the SMB protocol. The TCP/IP protocol must be installed on all computers. Samba provides a client for the different Unix flavors. For Linux, there is a kernel module for SMB that allows the integration of SMB resources on the Linux system level. You do not need to run any daemon for the Samba client.

Shares

SMB servers provide resources to the clients by means of shares. Shares are printers and directories with their subdirectories on the server. It is exported by means of a name and can be accessed by its name. The share name can be set to any name—it does not need to be the name of the export directory. A printer is also assigned a name. Clients can access the printer by its name.

DC

A domain controller (DC) is a server that handles accounts in a domain. For data replication, additional domain controllers are available in one domain.

21.2 Installing a Samba Server

To install a Samba server, start YaST and select *Software* › *Software Management*. Choose *View* › *Patterns* and select *File Server*. Confirm the installation of the required packages to finish the installation process.

21.3 Starting and Stopping Samba

You can start or stop the Samba server automatically (during boot) or manually. Starting and stopping policy is a part of the YaST Samba server configuration described in *Section 21.4.1, "Configuring a Samba Server with YaST"*.

From a command line, stop services required for Samba with **`systemctl stop smb nmb`** and start them with **`systemctl start nmb smb`**. The `smb` service cares about `winbind` if needed.

Tip: `winbind`

`winbind` is an independent service, and as such is also offered as an individual `samba-winbind` package.

21.4 Configuring a Samba Server

A Samba server in openSUSE® Leap can be configured in two different ways: with YaST or manually. Manual configuration offers a higher level of detail, but lacks the convenience of the YaST GUI.

21.4.1 Configuring a Samba Server with YaST

To configure a Samba server, start YaST and select *Network Services* › *Samba Server*.

21.4.1.1 Initial Samba Configuration

When starting the module for the first time, the *Samba Installation* dialog starts, prompting you to make a few basic decisions concerning administration of the server. At the end of the configuration, it prompts for the Samba administrator password (*Samba Root Password*). For later starts, the *Samba Configuration* dialog appears.

The *Samba Installation* dialog consists of two steps and optional detailed settings:

Workgroup or Domain Name

Select an existing name from *Workgroup or Domain Name* or enter a new one and click *Next*.

Samba Server Type

In the next step, specify whether your server should act as a primary domain controller (PDC), backup domain controller (BDC), or not act as a domain controller. Continue with *Next*.

If you do not want to proceed with a detailed server configuration, confirm with *OK*. Then in the final pop-up box, set the *Samba root Password*.

You can change all settings later in the *Samba Configuration* dialog with the *Start-Up*, *Shares*, *Identity*, *Trusted Domains*, and *LDAP Settings* tabs.

21.4.1.2 Enabling Current Versions of the SMB Protocol on the Server

On clients running current versions of openSUSE Leap or other recent Linux versions, the insecure SMB1 protocol is disabled by default. However, existing instances of Samba may be configured to only serve shares using the SMB1 version of the protocol. To interact with such clients, you need to configure Samba to serve shares using at least the SMB 2.1 protocol.

There are setups in which only SMB1 can be used, for example, because they rely on SMB1's/CIFS's Unix extensions. These extensions have not been ported to newer protocol versions. If you are in this situation, consider changing your setup or see *Section 21.5.2, "Mounting SMB1 Shares on Clients"*.

To do so, in the configuration file `/etc/samba/smb.conf`, set the global parameter `server max protocol = SMB2_10`. For a list of all possible values, see **`man smb.conf`**.

21.4.1.3 Advanced Samba Configuration

During the first start of the Samba server module the *Samba Configuration* dialog appears directly after the two initial steps described in *Section 21.4.1.1, "Initial Samba Configuration"*. Use it to adjust your Samba server configuration.

After editing your configuration, click *OK* to save your settings.

21.4.1.3.1 Starting the Server

In the *Start Up* tab, configure the start of the Samba server. To start the service every time your system boots, select *During Boot*. To activate manual start, choose *Manually*. More information about starting a Samba server is provided in *Section 21.3, "Starting and Stopping Samba"*.

In this tab, you can also open ports in your firewall. To do so, select *Open Port in Firewall*. If you have multiple network interfaces, select the network interface for Samba services by clicking *Firewall Details*, selecting the interfaces, and clicking *OK*.

21.4.1.3.2 Shares

In the *Shares* tab, determine the Samba shares to activate. There are some predefined shares, like homes and printers. Use *Toggle Status* to switch between *Active* and *Inactive*. Click *Add* to add new shares and *Delete* to delete the selected share.

Allow Users to Share Their Directories enables members of the group in *Permitted Group* to share directories they own with other users. For example, `users` for a local scope or `DOMAIN\Users` for a domain scope. The user also must make sure that the file system permissions allow access. With *Maximum Number of Shares*, limit the total amount of shares that may be created. To permit access to user shares without authentication, enable *Allow Guest Access*.

21.4.1.3.3 Identity

In the *Identity* tab, you can determine the domain with which the host is associated (*Base Settings*) and whether to use an alternative host name in the network (*NetBIOS Hostname*). It is also possible to use Microsoft Windows Internet Name Service (WINS) for name resolution. In this case, activate *Use WINS for Hostname Resolution* and decide whether to *Retrieve WINS server via DHCP*. To set expert global settings or set a user authentication source, for example LDAP instead of TDB database, click *Advanced Settings*.

21.4.1.3.4 Trusted Domains

To enable users from other domains to access your domain, make the appropriate settings in the *Trusted Domains* tab. To add a new domain, click *Add*. To remove the selected domain, click *Delete*.

21.4.1.3.5 LDAP Settings

In the tab *LDAP Settings*, you can determine the LDAP server to use for authentication. To test the connection to your LDAP server, click *Test Connection*. To set expert LDAP settings or use default values, click *Advanced Settings*.

For more information about LDAP configuration, see *Book "*Security 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$]

Using this approach, options of the shares can be set differently or globally in the `[global]` section, which makes the configuration file easier to understand.

21.4.2.1 The global Section

The following parameters of the `[global]` section should be modified to match the requirements of your network setup, so other machines can access your Samba server via SMB in a Windows environment.

`workgroup = WORKGROUP`

This line assigns the Samba server to a workgroup. Replace `WORKGROUP` with an appropriate workgroup of your networking environment. Your Samba server appears under its DNS name unless this name has been assigned to some other machine in the network. If the DNS name is not available, set the server name using `netbiosname=`*`MYNAME`*. For more details about this parameter, see the `smb.conf` man page.

`os level = 20`

This parameter triggers whether your Samba server tries to become LMB (local master browser) for its workgroup. Choose a very low value such as `2` to spare the existing Windows network from any interruptions caused by a misconfigured Samba server. More information about this topic can be found in the Network Browsing chapter of the Samba 3 Howto; for more information on the Samba 3 Howto, see *Section 21.9, "For More Information"*.

If no other SMB server is in your network (such as a Windows 2000 server) and you want the Samba server to keep a list of all systems present in the local environment, set the `os level` to a higher value (for example, `65`). Your Samba server is then chosen as LMB for your local network.

When changing this setting, consider carefully how this could affect an existing Windows network environment. First test the changes in an isolated network or at a noncritical time of day.

`wins support` **and** `wins server`

To integrate your Samba server into an existing Windows network with an active WINS server, enable the `wins server` option and set its value to the IP address of that WINS server.

If your Windows machines are connected to separate subnets and need to still be aware of each other, you have to set up a WINS server. To turn a Samba server into such a WINS server, set the option `wins support = Yes`. Make sure that only one Samba server of the network has this setting enabled. The options `wins server` and `wins support` must never be enabled at the same time in your `smb.conf` file.

21.4.2.2 Shares

The following examples illustrate how a CD-ROM drive and the user directories (`homes`) are made available to the SMB clients.

[cdrom]

To avoid having the CD-ROM drive accidentally made available, these lines are deactivated with comment marks (semicolons in this case). Remove the semicolons in the first column to share the CD-ROM drive with Samba.

EXAMPLE 21.1: A CD-ROM SHARE

```
[cdrom]
       comment = Linux CD-ROM
       path = /media/cdrom
```

```
        locking = No
```

`[cdrom]` and `comment`

The `[cdrom]` section entry is the name of the share that can be seen by all SMB clients on the network. An additional `comment` can be added to further describe the share.

`path = /media/cdrom`

`path` exports the directory `/media/cdrom`.

By means of a very restrictive default configuration, this kind of share is only made available to the users present on this system. If this share should be made available to everybody, add a line `guest ok = yes` to the configuration. This setting gives read permissions to anyone on the network. It is recommended to handle this parameter with great care. This applies even more to the use of this parameter in the `[global]` section.

`[homes]`

The `[homes]` share is of special importance here. If the user has a valid account and password for the Linux file server and their own home directory, they can be connected to it.

EXAMPLE 21.2: [HOMES] SHARE

```
[homes]
        comment = Home Directories
        valid users = %S
        browseable = No
        read only = No
        inherit acls = Yes
```

[homes]

As long as there is no other share using the share name of the user connecting to the SMB server, a share is dynamically generated using the `[homes]` share directives. The resulting name of the share is the user name.

`valid users = %S`

`%S` is replaced with the concrete name of the share when a connection has been successfully established. For a `[homes]` share, this is always the user name. As a consequence, access rights to a user's share are restricted exclusively to that user.

`browseable = No`

This setting makes the share invisible in the network environment.

`read only = No`

By default, Samba prohibits write access to any exported share by means of the `read only = Yes` parameter. To make a share writable, set the value `read only = No`, which is synonymous with `writable = Yes`.

`create mask = 0640`

Systems that are not based on MS Windows NT do not understand the concept of Unix permissions, so they cannot assign permissions when creating a file. The parameter `create mask` defines the access permissions assigned to newly created files. This only applies to writable shares. In effect, this setting means the owner has read and write permissions and the members of the owner's primary group have read permissions. `valid users = %S` prevents read access even if the group has read permissions. For the group to have read or write access, deactivate the line `valid users = %S`.

Warning: Do Not Share NFS Mounts with Samba

Sharing `NFS` mounts with Samba may result in data loss and is not supported. Install Samba directly on the file server or consider using alternatives such as `iSCSI`.

21.4.2.3 Security Levels

To improve security, each share access can be protected with a password. SMB offers the following ways of checking permissions:

User Level Security (`security = user`)

This variant introduces the concept of the user to SMB. Each user must register with the server with their own password. After registration, the server can grant access to individual exported shares dependent on user names.

ADS Level Security (`security = ADS`)

In this mode, Samba will act as a domain member in an Active Directory environment. To operate in this mode, the machine running Samba needs Kerberos installed and configured. You must join the machine using Samba to the ADS realm. This can be done using the YaST *Windows Domain Membership* module.

Domain Level Security (`security = domain`)

This mode will only work correctly if the machine has been joined into a Windows NT Domain. Samba will try to validate user name and password by passing it to a Windows NT Primary or Backup Domain Controller. The same way as a Windows NT Server would do. It expects the encrypted passwords parameter to be set to `yes`.

The selection of share, user, server, or domain level security applies to the entire server. It is not possible to offer individual shares of a server configuration with share level security and others with user level security. However, you can run a separate Samba server for each configured IP address on a system.

More information about this subject can be found in the Samba 3 HOWTO. For multiple servers on one system, pay attention to the options `interfaces` and `bind interfaces only`.

21.5 Configuring Clients

Clients can only access the Samba server via TCP/IP. NetBEUI and NetBIOS via IPX cannot be used with Samba.

21.5.1 Configuring a Samba Client with YaST

Configure a Samba client to access resources (files or printers) on the Samba or Windows server. Enter the NT or Active Directory domain or workgroup in the dialog *Network Services* › *Windows Domain Membership*. If you activate *Also Use SMB Information for Linux Authentication*, the user authentication runs over the Samba, NT or Kerberos server.

Click *Expert Settings* for advanced configuration options. For example, use the *Mount Server Directories* table to enable mounting server home directory automatically with authentication. This way users can access their home directories when hosted on CIFS. For details, see the `pam_mount` man page.

After completing all settings, confirm the dialog to finish the configuration.

21.5.2 Mounting SMB1 Shares on Clients

The first version of the SMB network protocol, SMB1, is an old and insecure protocol which has been deprecated by its originator, Microsoft. For security reasons, the **`mount`** command on openSUSE Leap will only mount SMB shares using newer protocol versions by default, namely SMB 2.1, SMB 3.0, or SMB 3.02.

However, this change only affects `mount` and mounting via `/etc/fstab`. SMB1 is still available by default when using the following:

- The `smbclient` tool.
- The Samba server software shipped with openSUSE.

There are setups in which this default setting will lead to connection failures, because only SMB1 can be used:

- Setups using an SMB server that does not support newer SMB protocol versions. Windows has offered SMB 2.1 support since Windows 7 and Windows Server 2008.
- Setups that rely on SMB1's/CIFS's Unix extensions. These extensions have not been ported to newer protocol versions.

Important: Decreased System Security

Following the instruction below makes it possible to exploit security issues. For more information about the issues, see https://blogs.technet.microsoft.com/filecab/2016/09/16/stop-using-smb1/.

As soon as possible, upgrade your server to allow for a more secure SMB version.

For information about enabling suitable protocol versions on openSUSE Leap, see *Section 21.4.1.2, "Enabling Current Versions of the SMB Protocol on the Server"*.

If you need to enable SMB1 shares on the current openSUSE Leap kernel, add the option `vers=1.0` to the `mount` command line you use:

```
root # mount -t smbfs IP_ADDRESS:/SHARE /MOUNT_POINT -o
 username=USER_ID,workgroup=WORKGROUP_NAME,vers=1.0
```

21.6 Samba as Login Server

In enterprise settings, it is often desirable to allow access only to users registered on a central instance. In a Windows-based network, this task is handled by a primary domain controller (PDC). You can use a Windows NT server configured as PDC, but this task can also be done with a Samba server. The entries that must be made in the `[global]` section of `smb.conf` are shown in *Example 21.3, "Global Section in smb.conf"*.

EXAMPLE 21.3: GLOBAL SECTION IN SMB.CONF

```
[global]
    workgroup = WORKGROUP
    domain logons = Yes
    domain master = Yes
```

It is necessary to prepare user accounts and passwords in an encryption format that conforms with Windows. Do this with the command **smbpasswd** `-a name`. Create the domain account for the computers, required by the Windows domain concept, with the following commands:

```
useradd hostname
smbpasswd -a -m hostname
```

With the **useradd** command, a dollar sign is added. The command **smbpasswd** inserts this automatically when the parameter `-m` is used. The commented configuration example (`/usr/share/doc/packages/samba/examples/smb.conf.SUSE`) contains settings that automate this task.

```
add machine script = /usr/sbin/useradd -g nogroup -c "NT Machine Account" \
-s /bin/false %m
```

To make sure that Samba can execute this script correctly, choose a Samba user with the required administrator permissions and add it to the `ntadmin` group. Then all users belonging to this Linux group can be assigned `Domain Admin` status with the command:

```
net groupmap add ntgroup="Domain Admins" unixgroup=ntadmin
```

21.7 Samba Server in the Network with Active Directory

If you run Linux servers and Windows servers together, you can build two independent authentication systems and networks or connect servers to one network with one central authentication system. Because Samba can cooperate with an active directory domain, you can join your openSUSE Leap server with an Active Directory (AD) domain.

To join an AD domain proceed as follows:

1. Log in as `root` and start YaST.

2. Start *Network Services* › *Windows Domain Membership*.

3. Enter the domain to join at *Domain or Workgroup* in the *Windows Domain Membership* screen.

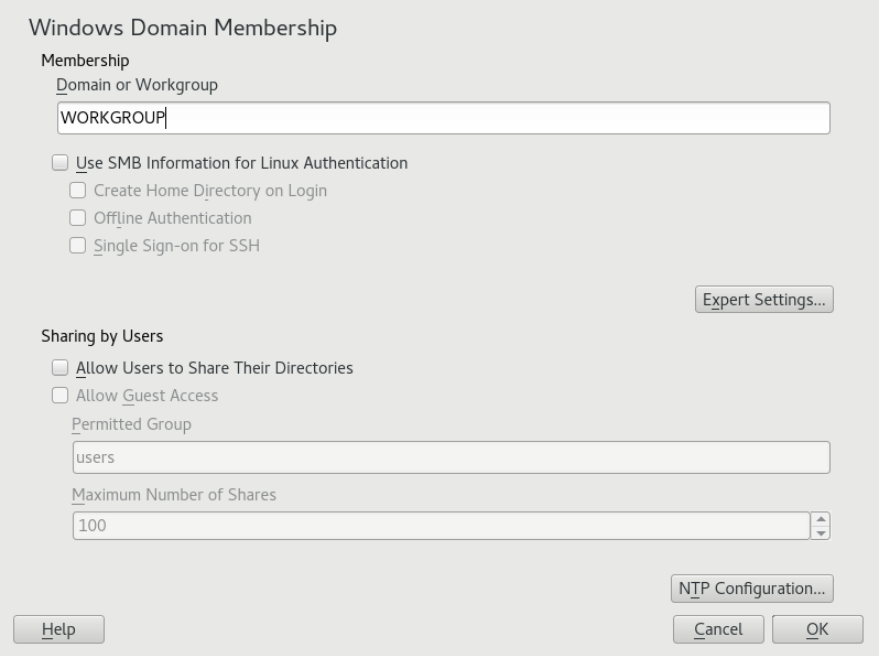


FIGURE 21.1: DETERMINING WINDOWS DOMAIN MEMBERSHIP

4. Check *Also Use SMB Information for Linux Authentication* to use the SMB source for Linux authentication on your server.

5. Click *OK* and confirm the domain join when prompted for it.

6. Provide the password for the Windows Administrator on the AD server and click *OK*. Your server is now set up to pull in all authentication data from the Active Directory domain controller.

Tip: Identity Mapping

In an environment with more than one Samba server, UIDs and GIDs will not be created consistently. The UIDs that get assigned to users will be dependent on the order in which they first log in, which results in UID conflicts across servers. To fix this, you need to use identity mapping. See https://www.samba.org/samba/docs/man/Samba-HOWTO-Collection/idmapper.html for more details.

21.8 Advanced Topics

This section introduces more advanced techniques to manage both the client and server part of the Samba suite.

21.8.1 Transparent File Compression on Btrfs

Samba allows clients to remotely manipulate file and directory compression flags for shares placed on the Btrfs file system. Windows Explorer provides the ability to flag files/directories for transparent compression via the *File* › *Properties* › *Advanced* dialog:

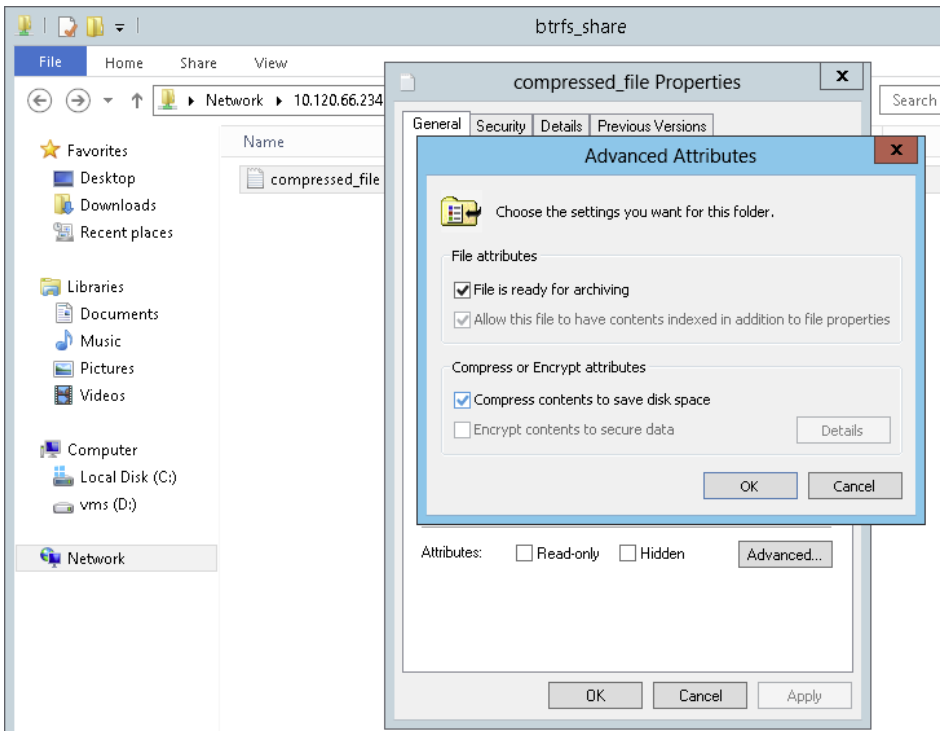


FIGURE 21.2: WINDOWS EXPLORER *ADVANCED ATTRIBUTES* DIALOG

Files flagged for compression are transparently compressed and decompressed by the underlying file system when accessed or modified. This normally results in storage capacity savings at the expense of extra CPU overhead when accessing the file. New files and directories inherit the compression flag from the parent directory, unless created with the FILE_NO_COMPRESSION option.

Windows Explorer presents compressed files and directories visually differently to those that are not compressed:

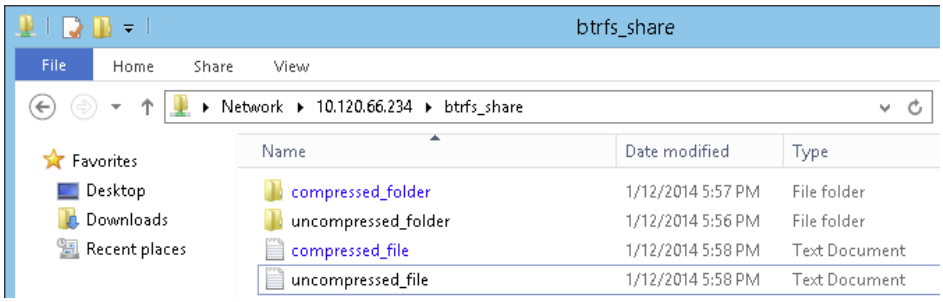


FIGURE 21.3: WINDOWS EXPLORER DIRECTORY LISTING WITH COMPRESSED FILES

You can enable Samba share compression either manually by adding

```
vfs objects = btrfs
```

to the share configuration in `/etc/samba/smb.conf`, or using YaST: *Network Services* › *Samba Server* › *Add*, and checking *Utilize Btrfs Features*.

21.8.2 Snapshots

Snapshots, also called Shadow Copies, are copies of the state of a file system subvolume at a certain point of time. Snapper is the tool to manage these snapshots in Linux. Snapshots are supported on the Btrfs file system or thin-provisioned LVM volumes. The Samba suite supports managing remote snapshots through the FSRVP protocol on both the server and client side.

21.8.2.1 Previous Versions

Snapshots on a Samba server can be exposed to remote Windows clients as file or directory previous versions.

To enable snapshots on a Samba server, the following conditions must be fulfilled:

- The SMB network share resides on a Btrfs subvolume.
- The SMB network share path has a related snapper configuration file. You can create the snapper file with

```
tux > sudo snapper -c <cfg_name> create-config /path/to/share
```

For more information on snapper, see *Chapter 3, System Recovery and Snapshot Management with Snapper*.

- The snapshot directory tree must allow access for relevant users. For more information, see the PERMISSIONS section of the vfs_snapper manual page (`man 8 vfs_snapper`).

To support remote snapshots, you need to modify the `/etc/samba/smb.conf` file. You can do it either with *YaST › Network Services › Samba Server*, or manually by enhancing the relevant share section with

```
vfs objects = snapper
```

Note that you need to restart the Samba service for manual `smb.conf` changes to take effect:

```
tux > sudo systemctl restart nmb smb
```

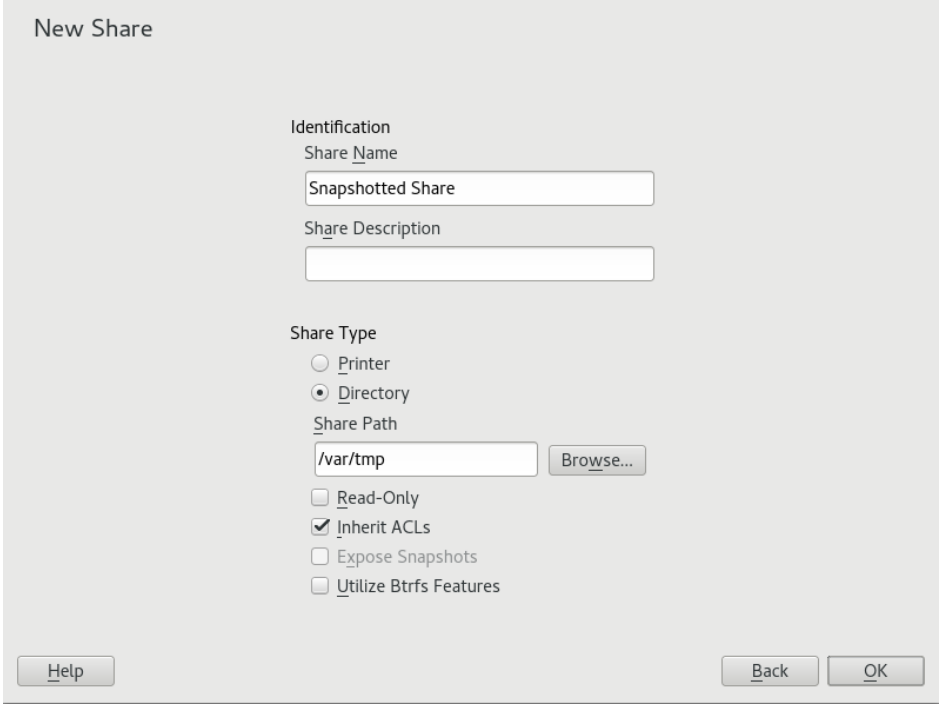


FIGURE 21.4: ADDING A NEW SAMBA SHARE WITH SNAPSHOTTING ENABLED

After being configured, snapshots created by snapper for the Samba share path can be accessed from Windows Explorer from a file or directory's *Previous Versions* tab.

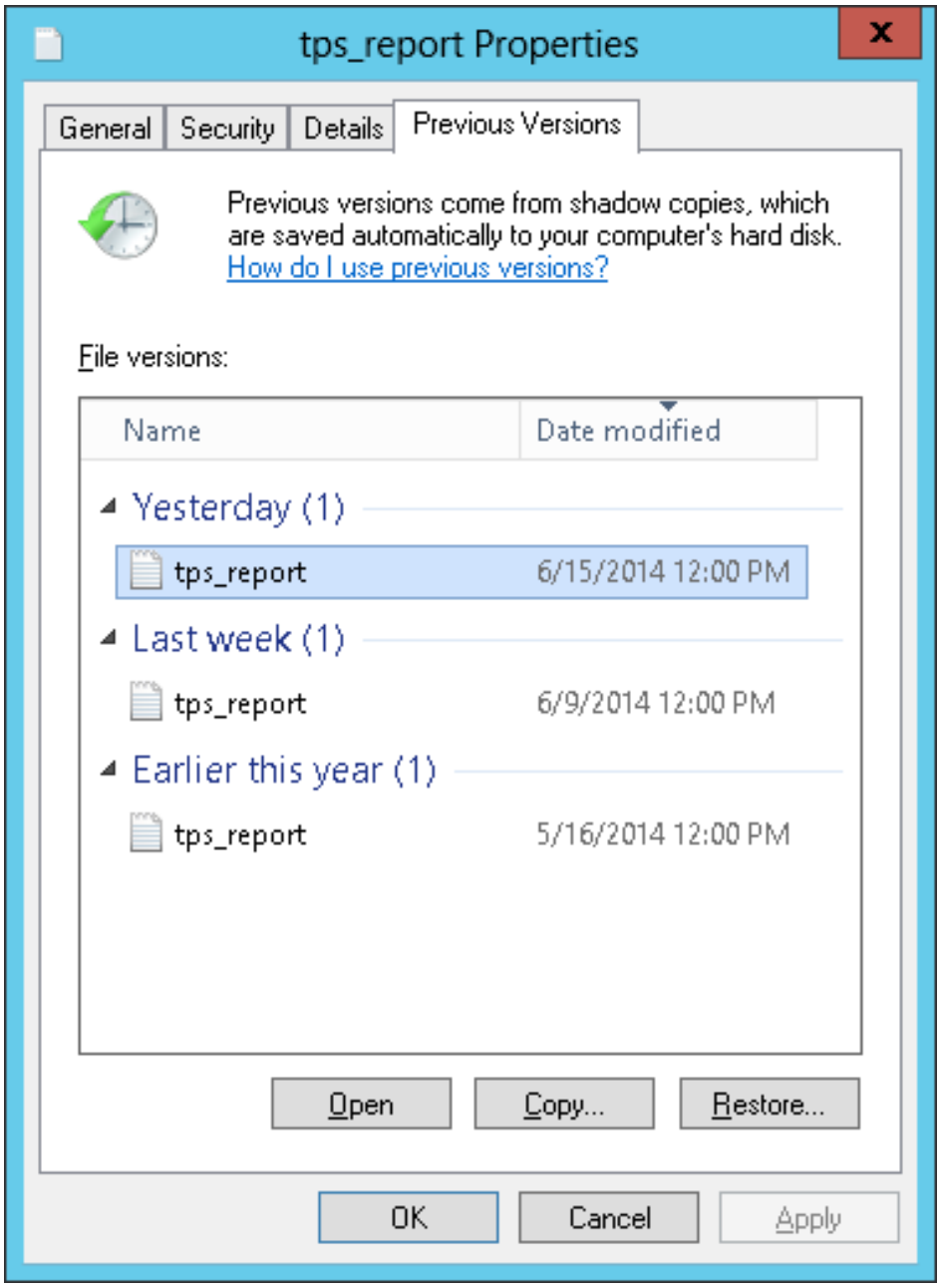


FIGURE 21.5: THE *PREVIOUS VERSIONS* TAB IN WINDOWS EXPLORER

21.8.2.2 Remote Share Snapshots

By default, snapshots can only be created and deleted on the Samba server locally, via the snapper command line utility, or using snapper's time line feature.

Samba can be configured to process share snapshot creation and deletion requests from remote hosts using the File Server Remote VSS Protocol (FSRVP).

In addition to the configuration and prerequisites documented in *Section 21.8.2.1, "Previous Versions"*, the following global configuration is required in `/etc/samba/smb.conf`:

```
[global]
rpc_daemon:fssd = fork
registry shares = yes
include = registry
```

FSRVP clients, including Samba's **`rpcclient`** and Windows Server 2012 **`DiskShadow.exe`**, can then instruct Samba to create or delete a snapshot for a given share, and expose the snapshot as a new share.

21.8.2.3 Managing Snapshots Remotely from Linux with **`rpcclient`**

The `samba-client` package contains an FSRVP client that can remotely request a Windows/Samba server to create and expose a snapshot of a given share. You can then use existing tools in openSUSE Leap to mount the exposed share and back up its files. Requests to the server are sent using the **`rpcclient`** binary.

EXAMPLE 21.4: USING **`rpcclient`** TO REQUEST A WINDOWS SERVER 2012 SHARE SNAPSHOT

Connect to `win-server.example.com` server as an administrator in an `EXAMPLE` domain:

```
root # rpcclient -U 'EXAMPLE\Administrator' ncacn_np:win-
server.example.com[ndr64,sign]
Enter EXAMPLE/Administrator's password:
```

Check that the SMB share is visible for **`rpcclient`**:

```
root # rpcclient $> netshareenum
netname: windows_server_2012_share
remark:
path:   C:\Shares\windows_server_2012_share
password:       (null)
```

Check that the SMB share supports snapshot creation:

```
root # rpcclient $> fss_is_path_sup windows_server_2012_share \
UNC \\WIN-SERVER\windows_server_2012_share\ supports shadow copy requests
```

Request the creation of a share snapshot:

```
root # rpcclient $> fss_create_expose backup ro windows_server_2012_share
13fe880e-e232-493d-87e9-402f21019fb6: shadow-copy set created
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
```

```
\\WIN-SERVER\windows_server_2012_share\ shadow-copy added to set
13fe880e-e232-493d-87e9-402f21019fb6: prepare completed in 0 secs
13fe880e-e232-493d-87e9-402f21019fb6: commit completed in 1 secs
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
share windows_server_2012_share@{1C26544E-8251-445F-BE89-D1E0A3938777} \
exposed as a snapshot of \\WIN-SERVER\windows_server_2012_share\
```

Confirm that the snapshot share is exposed by the server:

```
root # rpcclient $> netshareenum
netname: windows_server_2012_share
remark:
path:   C:\Shares\windows_server_2012_share
password:       (null)

netname: windows_server_2012_share@{1C26544E-8251-445F-BE89-D1E0A3938777}
remark: (null)
path:   \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy{F6E6507E-F537-11E3-9404-
B8AC6F927453}\Shares\windows_server_2012_share\
password:       (null)
```

Attempt to delete the snapshot share:

```
root # rpcclient $> fss_delete windows_server_2012_share \
13fe880e-e232-493d-87e9-402f21019fb6 1c26544e-8251-445f-be89-d1e0a3938777
13fe880e-e232-493d-87e9-402f21019fb6(1c26544e-8251-445f-be89-d1e0a3938777): \
\\WIN-SERVER\windows_server_2012_share\ shadow-copy deleted
```

Confirm that the snapshot share has been removed by the server:

```
root # rpcclient $> netshareenum
netname: windows_server_2012_share
remark:
path:   C:\Shares\windows_server_2012_share
password:       (null)
```

21.8.2.4 Managing Snapshots Remotely from Windows with `DiskShadow.exe`

You can manage snapshots of SMB shares on the Linux Samba server from the Windows environment acting as a client as well. Windows Server 2012 includes the **`DiskShadow.exe`** utility that can manage remote shares similar to the **`rpcclient`** described in *Section 21.8.2.3, "Managing Snapshots Remotely from Linux with `rpcclient`"*. Note that you need to carefully set up the Samba server first.

Following is an example procedure to set up the Samba server so that the Windows Server client can manage its share's snapshots. Note that *EXAMPLE* is the Active Directory domain used in the testing environment, `fsrvp-server.example.com` is the host name of the Samba server, and `/srv/smb` is the path to the SMB share.

PROCEDURE 21.1: DETAILED SAMBA SERVER CONFIGURATION

1. Join Active Directory domain via YaST. For more information, *Section 21.7, "Samba Server in the Network with Active Directory"*.
2. Ensure that the Active Domain DNS entry was correct:

```
fsrvp-server:~ # net -U 'Administrator' ads dns register \
fsrvp-server.example.com <IP address>
Successfully registered hostname with DNS
```

3. Create Btrfs subvolume at `/srv/smb`

```
fsrvp-server:~ # btrfs subvolume create /srv/smb
```

4. Create snapper configuration file for path `/srv/smb`

```
fsrvp-server:~ # snapper -c <snapper_config> create-config /srv/smb
```

5. Create new share with path `/srv/smb`, and YaST *Expose Snapshots* check box enabled. Make sure to add the following snippets to the global section of `/etc/samba/smb.conf` as mentioned in *Section 21.8.2.2, "Remote Share Snapshots"*:

```
[global]
 rpc_daemon:fssd = fork
 registry shares = yes
 include = registry
```

6. Restart Samba with **`systemctl restart nmb smb`**
7. Configure snapper permissions:

```
fsrvp-server:~ # snapper -c <snapper_config> set-config \
ALLOW_USERS="EXAMPLE\\\\Administrator EXAMPLE\\\\win-client$"
```

   Ensure that any ALLOW_USERS are also permitted traversal of the `.snapshots` subdirectory.

```
fsrvp-server:~ # snapper -c <snapper_config> set-config SYNC_ACL=yes
```

Important: Path Escaping

Be careful about the '\' escapes! Escape twice to ensure that the value stored in `/etc/snapper/configs/<snapper_config>` is escaped once.

"EXAMPLE\win-client$" corresponds to the Windows client computer account. Windows issues initial FSRVP requests while authenticated with this account.

8. Grant Windows client account necessary privileges:

```
fsrvp-server:~ # net -U 'Administrator' rpc rights grant \
"EXAMPLE\\win-client$" SeBackupPrivilege
Successfully granted rights.
```

The previous command is not needed for the "EXAMPLE\Administrator" user, which has privileges already granted.

PROCEDURE 21.2: WINDOWS CLIENT SETUP AND `DiskShadow.exe` IN ACTION

1. Boot Windows Server 2012 (example host name WIN-CLIENT).
2. Join the same Active Directory domain EXAMPLE as with the openSUSE Leap.
3. Reboot.
4. Open Powershell.
5. Start **`DiskShadow.exe`** and begin the backup procedure:

```
PS C:\Users\Administrator.EXAMPLE> diskshadow.exe
Microsoft DiskShadow version 1.0
Copyright (C) 2012 Microsoft Corporation
On computer:  WIN-CLIENT,  6/17/2014 3:53:54 PM

DISKSHADOW> begin backup
```

6. Specify that shadow copy persists across program exit, reset or reboot:

```
DISKSHADOW> set context PERSISTENT
```

7. Check whether the specified share supports snapshots, and create one:

```
DISKSHADOW> add volume \\fsrvp-server\sles_snapper
```

```
DISKSHADOW> create
Alias VSS_SHADOW_1 for shadow ID {de4ddca4-4978-4805-8776-cdf82d190a4a} set as \
 environment variable.
Alias VSS_SHADOW_SET for shadow set ID {c58e1452-c554-400e-a266-d11d5c837cb1} \
 set as environment variable.

Querying all shadow copies with the shadow copy set ID \
 {c58e1452-c554-400e-a266-d11d5c837cb1}

 * Shadow copy ID = {de4ddca4-4978-4805-8776-cdf82d190a4a}     %VSS_SHADOW_1%
    - Shadow copy set: {c58e1452-c554-400e-a266-d11d5c837cb1}  %VSS_SHADOW_SET%
    - Original count of shadow copies = 1
    - Original volume name: \\FSRVP-SERVER\SLES_SNAPPER\ \
      [volume not on this machine]
    - Creation time: 6/17/2014 3:54:43 PM
    - Shadow copy device name:
      \\FSRVP-SERVER\SLES_SNAPPER@{31afd84a-44a7-41be-b9b0-751898756faa}
    - Originating machine: FSRVP-SERVER
    - Service machine: win-client.example.com
    - Not exposed
    - Provider ID: {89300202-3cec-4981-9171-19f59559e0f2}
    - Attributes:  No_Auto_Release Persistent FileShare

Number of shadow copies listed: 1
```

8. Finish the backup procedure:

```
DISKSHADOW> end backup
```

9. After the snapshot was created, try to delete it and verify the deletion:

```
DISKSHADOW> delete shadows volume \\FSRVP-SERVER\SLES_SNAPPER\
Deleting shadow copy {de4ddca4-4978-4805-8776-cdf82d190a4a} on volume \
 \\FSRVP-SERVER\SLES_SNAPPER\ from provider \
{89300202-3cec-4981-9171-19f59559e0f2} [Attributes: 0x04000009]...

Number of shadow copies deleted: 1

DISKSHADOW> list shadows all

Querying all shadow copies on the computer ...
No shadow copies found in system.
```

21.9 For More Information

- **Man Pages:** To see a list of all man pages installed with the package `samba`, run **`apropos samba`**. Open any of the man pages with **`man`** *`NAME_OF_MAN_PAGE`*.
- **SUSE-specific README file:** The package `samba-client` contains the file `/usr/share/doc/packages/samba/README.SUSE`.
- **Additional Packaged Documentation:** Install the package `samba-doc` with **`zypper install samba-doc`**.
  This documentation installs into `/usr/share/doc/packages/samba`. It contains an HTML version of the man pages and a library of example configurations (such as `smb.conf.SUSE`).
- **Online Documentation:** The Samba wiki contains extensive *User Documentation* at https://wiki.samba.org/index.php/User_Documentation.

22 Sharing File Systems with NFS

The *Network File System* (*NFS*) is a protocol that allows access to files on a server ima a manner very similar to accessing local files.

22.1 Overview

The *Network File System* (NFS) is a standardized, well-proven and widely supported network protocol that allows files to be shared between separate hosts.

The *Network Information Service* (NIS) can be used to have a centralized user management in the network. Combining NFS and NIS allows using file and directory permissions for access control in the network. NFS with NIS makes a network transparent to the user.

In the default configuration, NFS completely trusts the network and thus any machine that is connected to a trusted network. Any user with administrator privileges on any computer with physical access to any network the NFS server trusts can access any files that the server makes available.

Often, this level of security is perfectly satisfactory, such as when the network that is trusted is truly private, often localized to a single cabinet or machine room, and no unauthorized access is possible. In other cases the need to trust a whole subnet as a unit is restrictive and there is a need for more fine-grained trust. To meet the need in these cases, NFS supports various security levels using the *Kerberos* infrastructure. Kerberos requires NFSv4, which is used by default. For details, see *Book "*Security Guide*", Chapter 6 "Network Authentication with Kerberos"*.

The following are terms used in the YaST module.

Exports

A directory *exported* by an NFS server, which clients can integrate 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` (ID-to-name mapping for NFSv4, needed for certain scenarios only), `statd` (file locking), and `mountd` (mount requests).

NFSv3

NFSv3 is the version 3 implementation, the “old” stateless NFS that supports client authentication.

NFSv4

NFSv4 is the new version 4 implementation that supports secure user authentication via kerberos. NFSv4 requires one single port only and thus is better suited for environments behind a firewall than NFSv3.

The protocol is specified as http://tools.ietf.org/html/rfc3530.

pNFS

Parallel NFS, a protocol extension of NFSv4. Any pNFS clients can directly access the data on an NFS server.

Important: Need for DNS

In principle, all exports can be made using IP addresses only. To avoid time-outs, you need a working DNS system. DNS is necessary at least for logging purposes, because the `mountd` daemon does reverse lookups.

22.2 Installing NFS Server

The NFS server is not part of the default installation. To install the NFS server using YaST, choose *Software* › *Software Management*, select *Patterns*, and enable the *File Server* option in the *Server Functions* section. Click *Accept* to install the required packages.

Like NIS, NFS is a client/server system. However, a machine can be both—it can supply file systems over the network (export) and mount file systems from other hosts (import).

Note: Mounting NFS Volumes Locally on the Exporting Server

Mounting NFS volumes locally on the exporting server is not supported on openSUSE Leap.

22.3 Configuring NFS Server

Configuring an NFS server can be done either through YaST or manually. For authentication, NFS can also be combined with Kerberos.

22.3.1 Exporting File Systems with YaST

With YaST, turn a host in your network into an NFS server—a server that exports directories and files to all hosts granted access to it or to all members of a group. Thus, the server can also provide applications without installing the applications locally on every host.

To set up such a server, proceed as follows:

PROCEDURE 22.1: SETTING UP AN NFS SERVER

1. Start YaST and select *Network Services* › *NFS Server*; see *Figure 22.1, "NFS Server Configuration Tool"*. You may be prompted to install additional software.

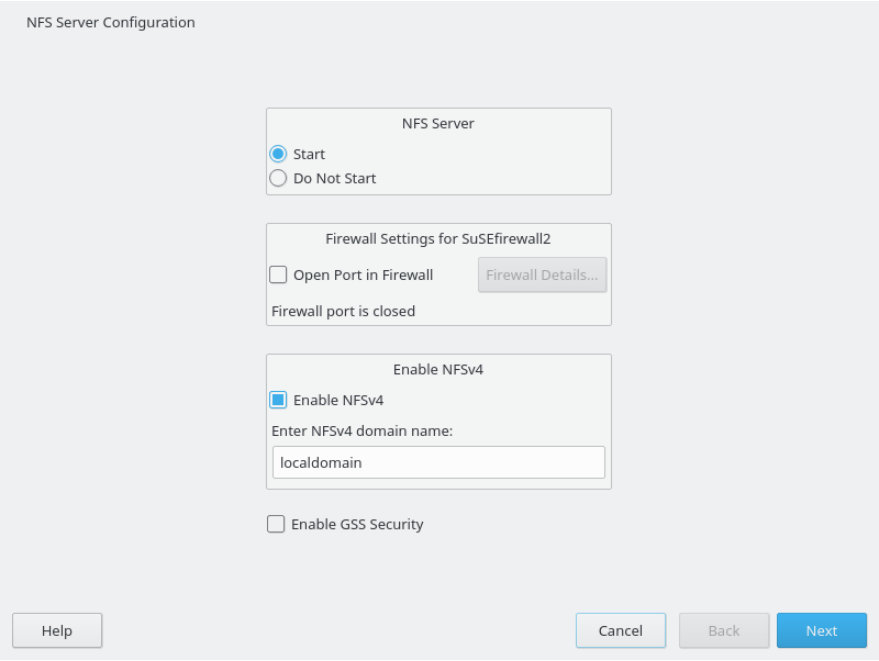


FIGURE 22.1: NFS SERVER CONFIGURATION TOOL

2. 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. For information about enabling NFSv2, see *Note: NFSv2*.

    - If NFSv4 is selected, additionally enter the appropriate NFSv4 domain name. This parameter is used by the `idmapd` daemon that is required for Kerberos setups or if clients cannot work with numeric user names. Leave it as `localdomain` (the default) if you do not run `idmapd` or do not have any special requirements. For more information on the `idmapd` daemon see `/etc/idmapd.conf`.

5. Click *Enable GSS Security* if you need secure access to the server. A prerequisite for this is to have Kerberos installed on your domain and to have both the server and the clients kerberized. Click *Next* to proceed with the next configuration dialog.

6. Click *Add Directory* in the upper half of the dialog to export your directory.

7. If you have not configured the allowed hosts already, another dialog for entering the client information and options pops up automatically. Enter the host wild card (usually you can leave the default settings as they are).
   There are four possible types of host wild cards that can be set for each host: a single host (name or IP address), netgroups, wild cards (such as `*` indicating all machines can access the server), and IP networks.
   For more information about these options, see the `exports` man page.

8. Click *Finish* to complete the configuration.

22.3.2 Exporting File Systems Manually

The configuration files for the NFS export service are `/etc/exports` and `/etc/sysconfig/nfs`. In addition to these files, `/etc/idmapd.conf` is needed for the NFSv4 server configuration with kerberized NFS or if the clients cannot work with numeric user names.

To start or restart the services, run the command **`systemctl restart nfsserver`**. This also restarts the RPC portmapper that is required by the NFS server.

To make sure the NFS server always starts at boot time, run **`sudo systemctl enable nfsserver`**.

Note: NFSv4

NFSv4 is the latest version of NFS protocol available on openSUSE Leap. Configuring directories for export with NFSv4 is now the same as with NFSv3.

On openSUSE prior to Leap, the bind mount in `/etc/exports` was mandatory. It is still supported, but now deprecated.

`/etc/exports`

The `/etc/exports` file contains a list of entries. Each entry indicates a directory that is shared and how it is shared. A typical entry in `/etc/exports` consists of:

```
/SHARED/DIRECTORY   HOST(OPTION_LIST)
```

For example:

```
/export/data   192.168.1.2(rw,sync)
```

Here the IP address `192.168.1.2` is used to identify the allowed client. You can also use the name of the host, a wild card indicating a set of hosts (`*.abc.com`, `*`, etc.), or netgroups (`@my-hosts`).

For a detailed explanation of all options and their meaning, refer to the man page of `/etc/exports` (**`man exports`**).

In case you have modified `/etc/exports` while the NFS server was running, you need to restart it for the changes to become active: **`sudo systemctl restart nfsserver`**.

`/etc/sysconfig/nfs`

The `/etc/sysconfig/nfs` file contains a few parameters that determine NFSv4 server daemon behavior. It is important to set the parameter `NFS4_SUPPORT` to `yes` (default). `NFS4_SUPPORT` determines whether the NFS server supports NFSv4 exports and clients.

In case you have modified `/etc/sysconfig/nfs` while the NFS server was running, you need to restart it for the changes to become active: **`sudo systemctl restart nfsserver`**.

Tip: Mount Options

On openSUSE prior to Leap, the `--bind` mount in `/etc/exports` was mandatory. It is still supported, but now deprecated. Configuring directories for export with NFSv4 is now the same as with NFSv3.

Note: NFSv2

If NFS clients still depend on NFSv2, enable it on the server in `/etc/sysconfig/nfs` by setting:

```
NFSD_OPTIONS="-V2"
```

```
MOUNTD_OPTIONS="-V2"
```

After restarting the service, check whether version 2 is available with the command:

```
tux > cat /proc/fs/nfsd/versions
+2 +3 +4 +4.1 -4.2
```

`/etc/idmapd.conf`

The `idmapd` daemon is only required if Kerberos authentication is used, or if clients cannot work with numeric user names. Linux clients can work with numeric user names since Linux kernel 2.6.39. The `idmapd` daemon does the name-to-ID mapping for NFSv4 requests to the server and replies to the client.

If required, `idmapd` needs to run on the NFSv4 server. Name-to-ID mapping on the client will be done by **`nfsidmap`** provided by the package `nfs-client`.

Make sure that there is a uniform way in which user names and IDs (UIDs) are assigned to users across machines that might probably be sharing file systems using NFS. This can be achieved by using NIS, LDAP, or any uniform domain authentication mechanism in your domain.

The parameter `Domain` must be set the same for both, client and server in the `/etc/idmapd.conf` file. If you are not sure, leave the domain as `localdomain` in the server and client files. A sample configuration file looks like the following:

```
[General]
Verbosity = 0
Pipefs-Directory = /var/lib/nfs/rpc_pipefs
Domain = localdomain

[Mapping]
Nobody-User = nobody
Nobody-Group = nobody
```

To start the `idmapd` daemon, run **`systemctl start nfs-idmapd`**. In case you have modified `/etc/idmapd.conf` while the daemon was running, you need to restart it for the changes to become active: **`systemctl start nfs-idmapd`**.

For more information, see the man pages of `idmapd` and `idmapd.conf` (`man idmapd` and `man idmapd.conf`).

22.3.3 NFS with Kerberos

To use Kerberos authentication for NFS, Generic Security Services (GSS) must be enabled. Select *Enable GSS Security* in the initial YaST NFS Server dialog. You must have a working Kerberos server to use this feature. YaST does not set up the server but only uses the provided functionality. To use Kerberos authentication in addition to the YaST configuration, complete at least the following steps before running the NFS configuration:

1. Make sure that both the server and the client are in the same Kerberos domain. They must access the same KDC (Key Distribution Center) server and share their `krb5.keytab` file (the default location on any machine is `/etc/krb5.keytab`). For more information about Kerberos, see *Book "Security Guide", Chapter 6 "Network Authentication with Kerberos"*.
2. Start the gssd service on the client with **`systemctl start rpc-gssd.service`**.
3. Start the svcgssd service on the server with **`systemctl start rpc-svcgssd.service`**.

Kerberos authentication also requires the `idmapd` daemon to run on the server. For more information refer to `/etc/idmapd.conf`.

For more information about configuring kerberized NFS, refer to the links in *Section 22.5, "For More Information"*.

22.4 Configuring Clients

To configure your host as an NFS client, you do not need to install additional software. All needed packages are installed by default.

22.4.1 Importing File Systems with YaST

Authorized users can mount NFS directories from an NFS server into the local file tree using the YaST NFS client module. Proceed as follows:

PROCEDURE 22.2: IMPORTING NFS DIRECTORIES

1. Start the YaST NFS client module.
2. Click *Add* in the *NFS Shares* tab. Enter the host name of the NFS server, the directory to import, and the mount point at which to mount this directory locally.

3. When using NFSv4, select *Enable NFSv4* in the *NFS Settings* tab. Additionally, the *NFSv4 Domain Name* must contain the same value as used by the NFSv4 server. The default domain is `localdomain`.
4. To use Kerberos authentication for NFS, GSS security must be enabled. Select *Enable GSS Security*.
5. Enable *Open Port in Firewall* in the *NFS Settings* tab if you use a Firewall and want to allow access to the service from remote computers. The firewall status is displayed next to the check box.
6. Click *OK* to save your changes.

The configuration is written to `/etc/fstab` and the specified file systems are mounted. When you start the YaST configuration client at a later time, it also reads the existing configuration from this file.

Tip: NFS as a Root File System

On (diskless) systems, where the root partition is mounted via network as an NFS share, you need to be careful when configuring the network device with which the NFS share is accessible.

When shutting down or rebooting the system, the default processing order is to turn off network connections, then unmount the root partition. With NFS root, this order causes problems as the root partition cannot be cleanly unmounted as the network connection to the NFS share is already not activated. To prevent the system from deactivating the relevant network device, open the network device configuration tab as described in *Section 13.4.1.2.5, "Activating the Network Device"* and choose *On NFSroot* in the *Device Activation* pane.

22.4.2 Importing File Systems Manually

The prerequisite for importing file systems manually from an NFS server is a running RPC port mapper. The `nfs` service takes care to start it properly; thus, start it by entering **`systemctl start nfs`** as `root`. Then remote file systems can be mounted in the file system like local partitions using **`mount`**:

```
tux > sudo mount HOST:REMOTE-PATHLOCAL-PATH
```

To import user directories from the `nfs.example.com` machine, for example, use:

```
tux > sudo mount nfs.example.com:/home /home
```

22.4.2.1 Using the Automount Service

The autofs daemon can be used to mount remote file systems automatically. Add the following entry to the `/etc/auto.master` file:

```
/nfsmounts /etc/auto.nfs
```

Now the `/nfsmounts` directory acts as the root for all the NFS mounts on the client if the `auto.nfs` file is filled appropriately. The name `auto.nfs` is chosen for the sake of convenience —you can choose any name. In `auto.nfs` add entries for all the NFS mounts as follows:

```
localdata -fstype=nfs server1:/data
nfs4mount -fstype=nfs4 server2:/
```

Activate the settings with **`systemctl start autofs`** as `root`. In this example, `/nfsmounts/localdata`, the `/data` directory of `server1`, is mounted with NFS and `/nfsmounts/nfs4mount` from `server2` is mounted with NFSv4.

If the `/etc/auto.master` file is edited while the service autofs is running, the automounter must be restarted for the changes to take effect with **`systemctl restart autofs`**.

22.4.2.2 Manually Editing `/etc/fstab`

A typical NFSv3 mount entry in `/etc/fstab` looks like this:

```
nfs.example.com:/data /local/path nfs rw,noauto 0 0
```

For NFSv4 mounts, use `nfs4` instead of `nfs` in the third column:

```
nfs.example.com:/data /local/pathv4 nfs4 rw,noauto 0 0
```

The `noauto` option prevents the file system from being mounted automatically at start-up. If you want to mount the respective file system manually, it is possible to shorten the mount command specifying the mount point only:

```
tux > sudo mount /local/path
```

Note: Mounting at Start-Up

If you do not enter the `noauto` option, the init scripts of the system will handle the mount of those file systems at start-up.

22.4.3 Parallel NFS (pNFS)

NFS is one of the oldest protocols, developed in the '80s. As such, NFS is usually sufficient if you want to share small files. However, when you want to transfer big files or many clients want to access data, an NFS server becomes a bottleneck and has a significant impact on the system performance. This is because of files quickly getting bigger, whereas the relative speed of your Ethernet has not fully kept up.

When you request a file from a regular NFS server, the server looks up the file metadata, collects all the data and transfers it over the network to your client. However, the performance bottleneck becomes apparent no matter how small or big the files are:

- With small files most of the time is spent collecting the metadata.
- With big files most of the time is spent on transferring the data from server to client.

pNFS, or parallel NFS, overcomes this limitation as it separates the file system metadata from the location of the data. As such, pNFS requires two types of servers:

- A *metadata* or *control server* that handles all the non-data traffic
- One or more *storage server(s)* that hold(s) the data

The metadata and the storage servers form a single, logical NFS server. When a client wants to read or write, the metadata server tells the NFSv4 client which storage server to use to access the file chunks. The client can access the data directly on the server.

openSUSE Leap supports pNFS on the client side only.

22.4.3.1 Configuring pNFS Client With YaST

Proceed as described in *Procedure 22.2, "Importing NFS Directories"*, but click the *pNFS (v4.1)* 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:

```
tux > sudo mount -t nfs4 -o minorversion=1 MDS_SERVER MOUNTPOINT
```

To help with debugging, change the value in the `/proc` file system:

```
tux > sudo echo 32767 > /proc/sys/sunrpc/nfsd_debug
tux > sudo echo 32767 > /proc/sys/sunrpc/nfs_debug
```

22.5 For More Information

In addition to the man pages of **`exports`**, **`nfs`**, and **`mount`**, information about configuring an NFS server and client is available in `/usr/share/doc/packages/nfsidmap/README`. For further documentation online refer to the following Web sites:

- Find the detailed technical documentation online at SourceForge (http://nfs.sourceforge.net/).
- For instructions for setting up kerberized NFS, refer to NFS Version 4 Open Source Reference Implementation (http://www.citi.umich.edu/projects/nfsv4/linux/krb5-setup.html).
- If you have questions on NFSv4, refer to the Linux NFSv4 FAQ (http://www.citi.umich.edu/projects/nfsv4/linux/faq/).

23 On-Demand Mounting with Autofs

`autofs` is a program that automatically mounts specified directories on an on-demand basis. It is based on a kernel module for high efficiency, and can manage both local directories and network shares. These automatic mount points are mounted only when they are accessed, and unmounted after a certain period of inactivity. This on-demand behavior saves bandwidth and results in better performance than static mounts managed by `/etc/fstab`. While `autofs` is a control script, **`automount`** is the command (daemon) that does the actual auto-mounting.

23.1 Installation

`autofs` is not installed on openSUSE Leap by default. To use its auto-mounting capabilities, first install it with

```
tux > sudo zypper install autofs
```

23.2 Configuration

You need to configure `autofs` manually by editing its configuration files with a text editor, such as **`vim`**. There are two basic steps to configure `autofs`—the *master* map file, and specific map files.

23.2.1 The Master Map File

The default master configuration file for `autofs` is `/etc/auto.master`. You can change its location by changing the value of the `DEFAULT_MASTER_MAP_NAME` option in `/etc/sysconfig/autofs`. Here is the content of the default one for openSUSE Leap:

```
#
# Sample auto.master file
# This is an automounter map and it has the following format
```

```
# key [ -mount-options-separated-by-comma ] location
# For details of the format look at autofs(5). (1)
#
#/misc  /etc/auto.misc (2)
#/net -hosts
#
# Include /etc/auto.master.d/*.autofs (3)
#
#+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 (4)
```

(1) The `autofs` manual page (**`man 5 autofs`**) offers a lot of valuable information on the format of the automounter maps.

(2) Although commented out (#) by default, this is an example of a simple automounter mapping syntax.

(3) 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.

(4) `+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
```

Make sure that map files are not marked as executable. You can remove the executable bits by executing **chmod -x** *MAP_FILE*.

mount point

Specifies where to mount the source location. This can be either a single directory name (so-called *indirect* mount) to be added to the base mount point specified in `auto.master`, or the full path of the mount point (direct mount, see *Section 23.2.1.1, "Direct Mounts"*).

options

Specifies optional comma-separated list of mount options for the relevant entries. If `auto.master` contains options for this map file as well, theses are appended.

location

Specifies from where the file system is to be mounted. It is usually an NFS or SMB volume in the usual notation `host_name:path_name`. If the file system to be mounted begins with a '/' (such as local `/dev` entries or smbfs shares), a colon symbol ':' needs to be prefixed, such as `:/dev/sda1`.

23.3 Operation and Debugging

This section introduces information on how to control the `autofs` service operation, and how to view more debugging information when tuning the automounter operation.

23.3.1 Controlling the `autofs` Service

The operation of the `autofs` service is controlled by `systemd`. The general syntax of the **systemctl** command for `autofs` is

```
tux > sudo systemctl SUB_COMMAND autofs
```

where *SUB_COMMAND* is one of:

enable

Starts the automounter daemon at boot.

start

Starts the automounter daemon.

stop

Stops the automounter daemon. Automatic mount points are not accessible.

status

Prints the current status of the `autofs` service together with a part of a relevant log file.

restart

Stops and starts the automounter, terminating all running daemons and starting new ones.

reload

Checks the current `auto.master` map, restarts those daemons whose entries have changed, and starts new ones for new entries.

23.3.2 Debugging the Automounter Problems

If you experience problems when mounting directories with `autofs`, it is useful to run the **`automount`** daemon manually and watch its output messages:

1. Stop `autofs`.

   ```
   tux > sudo systemctl stop autofs
   ```

2. From one terminal, run **`automount`** manually in the foreground, producing verbose output.

   ```
   tux > sudo automount -f -v
   ```

3. From another terminal, try to mount the auto-mounting file systems by accessing the mount points (for example by **`cd`** or **`ls`**).

4. Check the output of **`automount`** from the first terminal for more information why the mount failed, or why it was not even attempted.

23.4 Auto-Mounting an NFS Share

The following procedure illustrates how to configure `autofs` to auto-mount an NFS share available on your network. It uses the information mentioned above, and assumes you are familiar with NFS exports. For more information on NFS, see *Chapter 22, Sharing File Systems with NFS*.

1. Edit the master map file `/etc/auto.master`:

   ```
   tux > sudo vim /etc/auto.master
   ```

   Add a new entry for the new NFS mount at the end of `/etc/auto.master`:

   ```
   /nfs      /etc/auto.nfs      --timeout=10
   ```

   It tells `autofs` that the base mount point is `/nfs`, the NFS shares are specified in the `/etc/auto.nfs` map, and that all shares in this map will be automatically unmounted after 10 seconds of inactivity.

2. Create a new map file for NFS shares:

   ```
   tux > sudo vim /etc/auto.nfs
   ```

   `/etc/auto.nfs` normally contains a separate line for each NFS share. Its format is described in *Section 23.2.2, "Map Files"*. Add the line describing the mount point and the NFS share network address:

   ```
   export      jupiter.com:/home/geeko/doc/export
   ```

   The above line means that the `/home/geeko/doc/export` directory on the `jupiter.com` host will be auto-mounted to the `/nfs/export` directory on the local host (`/nfs` is taken from the `auto.master` map) when requested. The `/nfs/export` directory will be created automatically by `autofs`.

3. Optionally comment out the related line in `/etc/fstab` if you previously mounted the same NFS share statically. The line should look similar to this:

   ```
   #jupiter.com:/home/geeko/doc/export /nfs/export nfs defaults 0 0
   ```

4. Reload `autofs` and check if it works:

   ```
   tux > sudo systemctl restart autofs
   ```

```
# ls -l /nfs/export
total 20
drwxr-xr-x  5 1001 users 4096 Jan 14  2017 .images/
drwxr-xr-x 10 1001 users 4096 Aug 16  2017 .profiled/
drwxr-xr-x  3 1001 users 4096 Aug 30  2017 .tmp/
drwxr-xr-x  4 1001 users 4096 Apr 25 08:56 manual/
```

If you can see the list of files on the remote share, then `autofs` is functioning.

23.5 Advanced Topics

This section describes topics that are beyond the basic introduction to `autofs`—auto-mounting of NFS shares that are available on your network, using wild cards in map files, and information specific to the CIFS file system.

23.5.1 `/net` Mount Point

This helper mount point is useful if you use a lot of NFS shares. `/net` auto-mounts all NFS shares on your local network on demand. The entry is already present in the `auto.master` file, so all you need to do is uncomment it and restart `autofs`:

```
/net      -hosts
```

```
tux > sudo systemctl restart autofs
```

For example, if you have a server named `jupiter` with an NFS share called `/export`, you can mount it by typing

```
tux > sudo cd /net/jupiter/export
```

on the command line.

23.5.2 Using Wild Cards to Auto-Mount Subdirectories

If you have a directory with subdirectories that you need to auto-mount individually—the typical case is the `/home` directory with individual users' home directories inside— `autofs` offers a clever solution for that.

In case of home directories, add the following line in `auto.master`:

```
/home      /etc/auto.home
```

Now you need to add the correct mapping to the `/etc/auto.home` file, so that the users' home directories are mounted automatically. One solution is to create separate entries for each directory:

```
wilber      jupiter.com:/home/wilber
penguin      jupiter.com:/home/penguin
tux      jupiter.com:/home/tux
[...]
```

This is very awkward as you need to manage the list of users inside `auto.home`. You can use the asterisk '*' instead of the mount point, and the ampersand '&' instead of the directory to be mounted:

```
*      jupiter:/home/&
```

23.5.3 Auto-Mounting CIFS File System

If you want to auto-mount an SMB/CIFS share (see *Chapter 21, Samba* for more information on the SMB/CIFS protocol), you need to modify the syntax of the map file. Add `-fstype=cifs` in the option field, and prefix the share location with a colon ':'.

```
mount point      -fstype=cifs      ://jupiter.com/export
```

24 The Apache HTTP Server

According to the survey from http://www.netcraft.com/, the Apache HTTP Server (Apache) is the world's most widely-used Web server. Developed by the Apache Software Foundation (http://www.apache.org/), it is available for most operating systems. openSUSE® Leap includes Apache version 2.4. In this chapter, learn how to install, configure and set up a Web server; how to use SSL, CGI, and additional modules; and how to troubleshoot Apache.

24.1 Quick Start

With this section, quickly set up and start Apache. You must be `root` to install and configure Apache.

24.1.1 Requirements

Make sure the following requirements are met before trying to set up the Apache Web server:

1. The machine's network is configured properly. For more information about this topic, refer to *Chapter 13, Basic Networking*.
2. The machine's exact system time is maintained by synchronizing with a time server. This is necessary because parts of the HTTP protocol depend on the correct time. See *Chapter 18, Time Synchronization with NTP* to learn more about this topic.
3. The latest security updates are installed. If in doubt, run a YaST Online Update.
4. The default Web server port (`80`) is opened in the firewall. For this, configure `firewalld` to allow the service `http` in the public zone. See *Book "Security Guide", Chapter 16 "Masquerading and Firewalls", Section 16.4.1 "Configuring the Firewall on the Command Line"* for details.

24.1.2 Installation

Apache on openSUSE Leap is not installed by default. To install it with a standard, predefined configuration that runs “out of the box”, proceed as follows:

PROCEDURE 24.1: INSTALLING APACHE WITH THE DEFAULT CONFIGURATION

1. Start YaST and select *Software* › *Software Management*.
2. Choose *View* › *Patterns* and select *Web and LAMP Server*.
3. Confirm the installation of the dependent packages to finish the installation process.

24.1.3 Start

You can start Apache automatically at boot time or start it manually.

To make sure that Apache is automatically started during boot in the targets `multi-user.target` and `graphical.target`, execute the following command:

```
tux > sudo systemctl enable apache2
```

For more information about the `systemd` targets in openSUSE Leap and a description of the YaST *Services Manager*, refer to *Section 10.4, “Managing Services with YaST”*.

To manually start Apache using the shell, run **`systemctl start apache2`**.

PROCEDURE 24.2: CHECKING IF APACHE IS RUNNING

If you do not receive error messages when starting Apache, this usually indicates that the Web server is running. To test this:

1. Start a browser and open http://localhost/.
   If Apache is up and running, you get a test page stating “It works!”.
2. If you do not see this page, refer to *Section 24.9, “Troubleshooting”*.

Now that the Web server is running, you can add your own documents, adjust the configuration according to your needs, or add functionality by installing modules.

24.2 Configuring Apache

openSUSE Leap offers two configuration options:

- *Configuring Apache Manually*
- *Configuring Apache with YaST*

Manual configuration offers a higher level of detail, but lacks the convenience of the YaST GUI.

Important: Reload or Restart Apache after Configuration Changes

Most configuration changes require a reload (some also a restart) of Apache to take effect. Manually reload Apache with **`systemctl reload apache2`** or use one of the restart options as described in *Section 24.3, "Starting and Stopping Apache"*.

If you configure Apache with YaST, this can be taken care of automatically if you set *HTTP Service* to *Enabled* as described in *Section 24.2.3.2, "HTTP Server Configuration"*.

24.2.1 Apache Configuration Files

This section gives an overview of the Apache configuration files. If you use YaST for configuration, you do not need to touch these files—however, the information might be useful for you to switch to manual configuration later on.

Apache configuration files can be found in two different locations:

- `/etc/sysconfig/apache2`
- `/etc/apache2/`

24.2.1.1 `/etc/sysconfig/apache2`

`/etc/sysconfig/apache2` controls some global settings of Apache, like modules to load, additional configuration files to include, flags with which the server should be started, and flags that should be added to the command line. Every configuration option in this file is extensively documented and therefore not mentioned here. For a general-purpose Web server, the settings in `/etc/sysconfig/apache2` should be sufficient for any configuration needs.

24.2.1.2 /etc/apache2/

`/etc/apache2/` hosts all configuration files for Apache. In the following, the purpose of each file is explained. Each file includes several configuration options (also called *directives*). Every configuration option in these files is extensively documented and therefore not mentioned here.

The Apache configuration files are organized as follows:

```
/etc/apache2/
     |
     |- charset.conv
     |- conf.d/
     |   |
     |   |- *.conf
     |
     |- default-server.conf
     |- errors.conf
     |- httpd.conf
     |- listen.conf
     |- magic
     |- mime.types
     |- mod_*.conf
     |- server-tuning.conf
     |- ssl.*
     |- ssl-global.conf
     |- sysconfig.d
     |   |
     |   |- global.conf
     |   |- include.conf
     |   |- loadmodule.conf . .
     |
     |- uid.conf
     |- vhosts.d
     |   |- *.conf
```

APACHE CONFIGURATION FILES IN /ETC/APACHE2/

`charset.conv`

Specifies which character sets to use for different languages. Do not edit this file.

`conf.d/*.conf`

Configuration files added by other modules. These configuration files can be included into your virtual host configuration where needed. See `vhosts.d/vhost.template` for examples. By doing so, you can provide different module sets for different virtual hosts.

`default-server.conf`

Global configuration for all virtual hosts with reasonable defaults. Instead of changing the values, overwrite them with a virtual host configuration.

`errors.conf`

Defines how Apache responds to errors. To customize these messages for all virtual hosts, edit this file. Otherwise overwrite these directives in your virtual host configurations.

`httpd.conf`

The main Apache server configuration file. Avoid changing this file. It primarily contains include statements and global settings. Overwrite global settings in the pertinent configuration files listed here. Change host-specific settings (such as document root) in your virtual host configuration.

`listen.conf`

Binds Apache to specific IP addresses and ports. Name-based virtual hosting is also configured here. For details, see *Section 24.2.2.1.1, "Name-Based Virtual Hosts"*.

`magic`

Data for the mime_magic module that helps Apache automatically determine the MIME type of an unknown file. Do not change this file.

`mime.types`

MIME types known by the system (this actually is a link to `/etc/mime.types`). Do not edit this file. If you need to add MIME types not listed here, add them to `mod_mime-defaults.conf`.

`mod_*.conf`

Configuration files for the modules that are installed by default. Refer to *Section 24.4, "Installing, Activating, and Configuring Modules"* for details. Note that configuration files for optional modules reside in the directory `conf.d`.

`server-tuning.conf`

Contains configuration directives for the different MPMs (see *Section 24.4.4, "Multiprocessing Modules"*) and general configuration options that control Apache's performance. Properly test your Web server when making changes here.

`ssl-global.conf` and `ssl.*`

Global SSL configuration and SSL certificate data. Refer to *Section 24.6, "Setting Up a Secure Web Server with SSL"* for details.

`sysconfig.d/*.conf`

Configuration files automatically generated from `/etc/sysconfig/apache2`. Do not change any of these files—edit `/etc/sysconfig/apache2` instead. Do not put other configuration files in this directory.

`uid.conf`

Specifies under which user and group ID Apache runs. Do not change this file.

`vhosts.d/*.conf`

Your virtual host configuration should be located here. The directory contains template files for virtual hosts with and without SSL. Every file in this directory ending with `.conf` is automatically included in the Apache configuration. Refer to *Section 24.2.2.1, "Virtual Host Configuration"* for details.

24.2.2 Configuring Apache Manually

Configuring Apache manually involves editing plain text configuration files as user `root`.

24.2.2.1 Virtual Host Configuration

The term *virtual host* refers to Apache's ability to serve multiple universal resource identifiers (URIs) from the same physical machine. This means that several domains, such as www.example.com and www.example.net, are run by a single Web server on one physical machine.

It is common practice to use virtual hosts to save administrative effort (only a single Web server needs to be maintained) and hardware expenses (each domain does not require a dedicated server). Virtual hosts can be name based, IP based, or port based.

To list all existing virtual hosts, use the command **`apache2ctl`** `-S`. This outputs a list showing the default server and all virtual hosts together with their IP addresses and listening ports. Furthermore, the list also contains an entry for each virtual host showing its location in the configuration files.

Virtual hosts can be configured via YaST as described in *Section 24.2.3.1.4, "Virtual Hosts"* or by manually editing a configuration file. By default, Apache in openSUSE Leap is prepared for one configuration file per virtual host in `/etc/apache2/vhosts.d/`. All files in this directory with the extension `.conf` are automatically included to the configuration. A basic template for a virtual host is provided in this directory (`vhost.template` or `vhost-ssl.template` for a virtual host with SSL support).

Tip: Always Create a Virtual Host Configuration

It is recommended to always create a virtual host configuration file, even if your Web server only hosts one domain. By doing so, you not only have the domain-specific configuration in one file, but you can always fall back to a working basic configuration by simply moving, deleting, or renaming the configuration file for the virtual host. For the same reason, you should also create separate configuration files for each virtual host.

When using name-based virtual hosts it is recommended to set up a default configuration that will be used when a domain name does not match a virtual host configuration. The default virtual host is the one whose configuration is loaded first. Since the order of the configuration files is determined by file name, start the file name of the default virtual host configuration with an underscore character (`_`) to make sure it is loaded first (for example: `_default_vhost.conf`).

The `<VirtualHost> </VirtualHost>` block holds the information that applies to a particular domain. When Apache receives a client request for a defined virtual host, it uses the directives enclosed in this section. Almost all directives can be used in a virtual host context. See http://httpd.apache.org/docs/2.4/mod/quickreference.html for further information about Apache's configuration directives.

24.2.2.1.1 Name-Based Virtual Hosts

With name-based virtual hosts, more than one Web site is served per IP address. Apache uses the host field in the HTTP header that is sent by the client to connect the request to a matching `ServerName` entry of one of the virtual host declarations. If no matching `ServerName` is found, the first specified virtual host is used as a default.

The first step is to create a `<VirtualHost>` block for each different name-based host that you want to serve. Inside each `<VirtualHost>` block, you will need at minimum a `ServerName` directive to designate which host is served and a `DocumentRoot` directive to show where in the file system the content for that host resides.

EXAMPLE 24.1: BASIC EXAMPLES OF NAME-BASED `VirtualHost` ENTRIES

```
<VirtualHost *:80>
# This first-listed virtual host is also the default for *:80
ServerName www.example.com
ServerAlias example.com
DocumentRoot /srv/www/htdocs/domain
```

```
</VirtualHost>

<VirtualHost *:80>
ServerName other.example.com
DocumentRoot /srv/www/htdocs/otherdomain
</VirtualHost>
```

The opening `VirtualHost` tag takes the IP address (or fully qualified domain name) as an argument in a name-based virtual host configuration. A port number directive is optional.

The wild card * is also allowed as a substitute for the IP address. When using IPv6 addresses, the address must be included in square brackets.

EXAMPLE 24.2: NAME-BASED `VirtualHost` DIRECTIVES

```
<VirtualHost 192.168.3.100:80>
  ...
</VirtualHost>

<VirtualHost 192.168.3.100>
  ...
</VirtualHost>

<VirtualHost *:80>
  ...
</VirtualHost>

<VirtualHost *>
  ...
</VirtualHost>

<VirtualHost [2002:c0a8:364::]>
  ...
</VirtualHost>
```

24.2.2.1.2 IP-Based Virtual Hosts

This alternative virtual host configuration requires the setup of multiple IP addresses for a machine. One instance of Apache hosts several domains, each of which is assigned a different IP.

The physical server must have one IP address for each IP-based virtual host. If the machine does not have multiple network cards, virtual network interfaces (IP aliasing) can also be used.

The following example shows Apache running on a machine with the IP `192.168.3.100`, hosting two domains on the additional IP addresses `192.168.3.101` and `192.168.3.102`. A separate `VirtualHost` block is needed for every virtual server.

EXAMPLE 24.3: IP-BASED `VirtualHost` DIRECTIVES

```
<VirtualHost 192.168.3.101>
  ...
</VirtualHost>

<VirtualHost 192.168.3.102>
  ...
</VirtualHost>
```

Here, `VirtualHost` directives are only specified for interfaces other than `192.168.3.100`. When a `Listen` directive is also configured for `192.168.3.100`, a separate IP-based virtual host must be created to answer HTTP requests to that interface—otherwise the directives found in the default server configuration (`/etc/apache2/default-server.conf`) are applied.

24.2.2.1.3 Basic Virtual Host Configuration

At least the following directives should be in each virtual host configuration to set up a virtual host. See `/etc/apache2/vhosts.d/vhost.template` for more options.

`ServerName`
: The fully qualified domain name under which the host should be addressed.

`DocumentRoot`
: Path to the directory from which Apache should serve files for this host. For security reasons, access to the entire file system is forbidden by default, so you must explicitly unlock this directory within a `Directory` container.

`ServerAdmin`
: E-mail address of the server administrator. This address is, for example, shown on error pages Apache creates.

`ErrorLog`
: The error log file for this virtual host. Although it is not necessary to create separate error log files for each virtual host, it is common practice to do so, because it makes the debugging of errors much easier. `/var/log/apache2/` is the default directory for Apache's log files.

`CustomLog`

The access log file for this virtual host. Although it is not necessary to create separate access log files for each virtual host, it is common practice to do so, because it allows the separate analysis of access statistics for each host. `/var/log/apache2/` is the default directory for Apache's log files.

As mentioned above, access to the whole file system is forbidden by default for security reasons. Therefore, explicitly unlock the directories in which you have placed the files Apache should serve—for example the `DocumentRoot`:

```
<Directory "/srv/www/www.example.com/htdocs">
  Require all granted
</Directory>
```

Note: `Require all granted`

In previous versions of Apache, the statement `Require all granted` was expressed as:

```
Order allow,deny
Allow from all
```

This old syntax is still supported by the `mod_access_compat` module.

The complete configuration file looks like this:

EXAMPLE 24.4: BASIC `VirtualHost` CONFIGURATION

```
<VirtualHost 192.168.3.100>
  ServerName www.example.com
  DocumentRoot /srv/www/www.example.com/htdocs
  ServerAdmin webmaster@example.com
  ErrorLog /var/log/apache2/www.example.com_log
  CustomLog /var/log/apache2/www.example.com-access_log common
  <Directory "/srv/www/www.example.com/htdocs">
  Require all granted
  </Directory>
</VirtualHost>
```

24.2.3 Configuring Apache with YaST

To configure your Web server with YaST, start YaST and select *Network Services* › *HTTP Server*. When starting the module for the first time, the *HTTP Server Wizard* starts, prompting you to make a few basic decisions concerning administration of the server. After having finished the wizard, the *HTTP Server Configuration* dialog starts each time you call the *HTTP Server* module. For more information, see *Section 24.2.3.2, "HTTP Server Configuration"*.

24.2.3.1 HTTP Server Wizard

The HTTP Server Wizard consists of five steps. In the last step of the dialog, you may enter the expert configuration mode to make even more specific settings.

24.2.3.1.1 Network Device Selection

Here, specify the network interfaces and ports Apache uses to listen for incoming requests. You can select any combination of existing network interfaces and their respective IP addresses. Ports from all three ranges (well-known ports, registered ports, and dynamic or private ports) that are not reserved by other services can be used. The default setting is to listen on all network interfaces (IP addresses) on port `80`.

Check *Open Port In Firewall* to open the ports in the firewall that the Web server listens on. This is necessary to make the Web server available on the network, which can be a LAN, WAN, or the public Internet. Keeping the port closed is only useful in test situations where no external access to the Web server is necessary. If you have multiple network interfaces, click *Firewall Details* to specify on which interface(s) the port(s) should be opened.

Click *Next* to continue with the configuration.

24.2.3.1.2 Modules

The *Modules* configuration option allows for the activation or deactivation of the script languages that the Web server should support. For the activation or deactivation of other modules, refer to *Section 24.2.3.2.2, "Server Modules"*. Click *Next* to advance to the next dialog.

24.2.3.1.3 Default Host

This option pertains to the default Web server. As explained in *Section 24.2.2.1, "Virtual Host Configuration"*, Apache can serve multiple virtual hosts from a single physical machine. The first declared virtual host in the configuration file is commonly called the *default host*. Each virtual host inherits the default host's configuration.

To edit the host settings (also called *directives*), select the appropriate entry in the table then click *Edit*. To add new directives, click *Add*. To delete a directive, select it and click *Delete*.

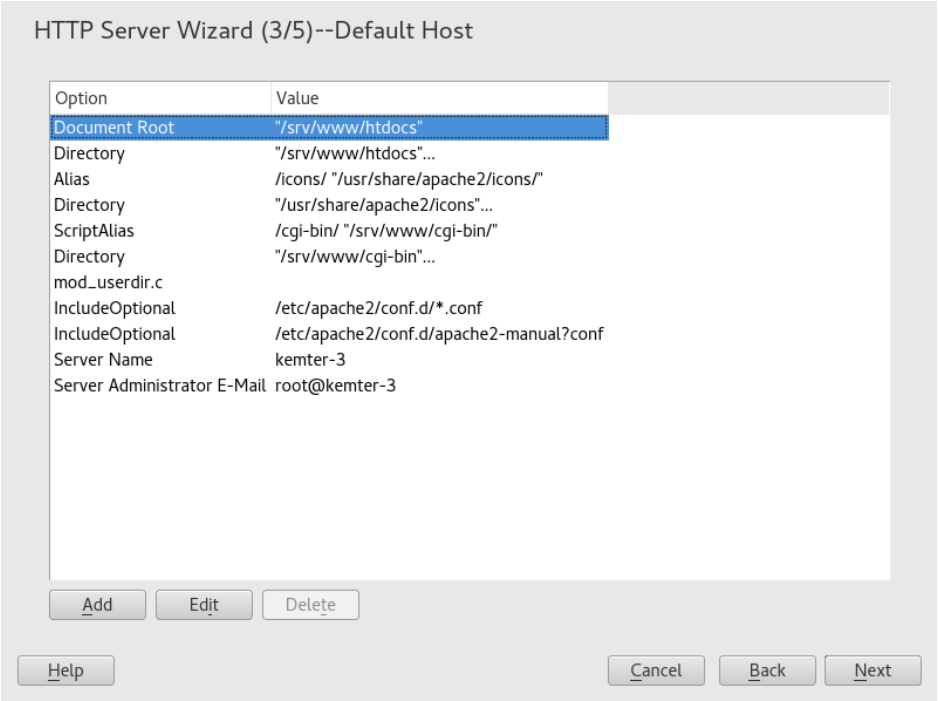


FIGURE 24.1: HTTP SERVER WIZARD: DEFAULT HOST

Here is list of the default settings of the server:

`Document Root`

Path to the directory from which Apache serves files for this host. `/srv/www/htdocs` is the default location.

`Alias`

Using `Alias` directives, URLs can be mapped to physical file system locations. This means that a certain path even outside the `Document Root` in the file system can be accessed via a URL aliasing that path.

The default openSUSE Leap `Alias /icons` points to `/usr/share/apache2/icons` for the Apache icons displayed in the directory index view.

`ScriptAlias`

Similar to the `Alias` directive, the `ScriptAlias` directive maps a URL to a file system location. The difference is that `ScriptAlias` designates the target directory as a CGI location, meaning that CGI scripts should be executed in that location.

`Directory`

With `Directory` settings, you can enclose a group of configuration options that will only apply to the specified directory.

Access and display options for the directories `/srv/www/htdocs`, `/usr/share/apache2/icons` and `/srv/www/cgi-bin` are configured here. It should not be necessary to change the defaults.

`Include`

With include, additional configuration files can be specified. Two `Include` directives are already preconfigured: `/etc/apache2/conf.d/` is the directory containing the configuration files that come with external modules. With this directive, all files in this directory ending in `.conf` are included. With the second directive, `/etc/apache2/conf.d/apache2-manual.conf`, the `apache2-manual` configuration file is included.

`Server Name`

This specifies the default URL used by clients to contact the Web server. Use a fully qualified domain name (FQDN) to reach the Web server at `http://FQDN/` or its IP address. You cannot choose an arbitrary name here—the server must be "known" under this name.

`Server Administrator E-Mail`

E-mail address of the server administrator. This address is, for example, shown on error pages Apache creates.

After finishing with the *Default Host* step, click *Next* to continue with the configuration.

24.2.3.1.4 Virtual Hosts

In this step, the wizard displays a list of already configured virtual hosts (see *Section 24.2.2.1, "Virtual Host Configuration"*). If you have not made manual changes prior to starting the YaST HTTP wizard, no virtual host is present.

To add a host, click *Add* to open a dialog in which to enter basic information about the host, such as *Server Name*, *Server Contents Root* (`DocumentRoot`), and the *Administrator E-Mail*. *Server Resolution* is used to determine how a host is identified (name based or IP based). Specify the name or IP address with *Change Virtual Host ID*

Clicking *Next* advances to the second part of the virtual host configuration dialog.

In part two of the virtual host configuration you can specify whether to enable CGI scripts and which directory to use for these scripts. It is also possible to enable SSL. If you do so, you must specify the path to the certificate as well. See *Section 24.6.2, "Configuring Apache with SSL"* for details on SSL and certificates. With the *Directory Index* option, you can specify which file to display when the client requests a directory (by default, `index.html`). Add one or more file names (space-separated) to change this. With *Enable Public HTML*, the content of the users public directories (`~USER/public_html/`) is made available on the server under `http://www.example.com/~USER`.

Important: Creating Virtual Hosts

It is not possible to add virtual hosts at will. If using name-based virtual hosts, each host name must be resolved on the network. If using IP-based virtual hosts, you can assign only one host to each IP address available.

24.2.3.1.5 Summary

This is the final step of the wizard. Here, determine how and when the Apache server is started: when booting or manually. Also see a short summary of the configuration made so far. If you are satisfied with your settings, click *Finish* to complete configuration. To change something, click *Back* until you have reached the desired dialog. Clicking *HTTP Server Expert Configuration* opens the dialog described in *Section 24.2.3.2, "HTTP Server Configuration"*.

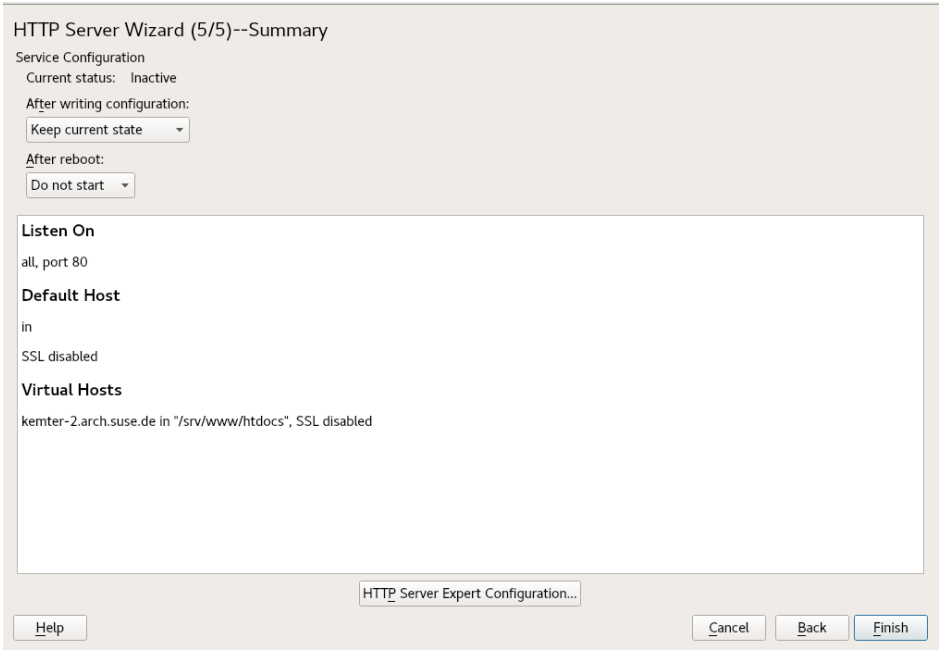


FIGURE 24.2: HTTP SERVER WIZARD: SUMMARY

24.2.3.2 HTTP Server Configuration

The *HTTP Server Configuration* dialog also lets you make even more adjustments to the configuration than the wizard (which only runs if you configure your Web server for the first time). It consists of four tabs described in the following. No configuration option you change here is effective immediately—you always must confirm your changes with *Finish* to make them effective. Clicking *Abort* leaves the configuration module and discards your changes.

24.2.3.2.1 Listen Ports and Addresses

In *HTTP Service*, select whether Apache should be running (*Enabled*) or stopped (*Disabled*). In *Listen on Ports*, *Add*, *Edit*, or *Delete* addresses and ports on which the server should be available. The default is to listen on all interfaces on port `80`. You should always check *Open Port In Firewall*, because otherwise the Web server is not reachable from outside. Keeping the port closed is only useful in test situations where no external access to the Web server is necessary. If you have multiple network interfaces, click *Firewall Details* to specify on which interface(s) the port(s) should be opened.

With *Log Files*, watch either the access log file or the error log file. This is useful if you want to test your configuration. The log file opens in a separate window from which you can also restart or reload the Web server. For details, see *Section 24.3, "Starting and Stopping Apache"*. These commands are effective immediately and their log messages are also displayed immediately.

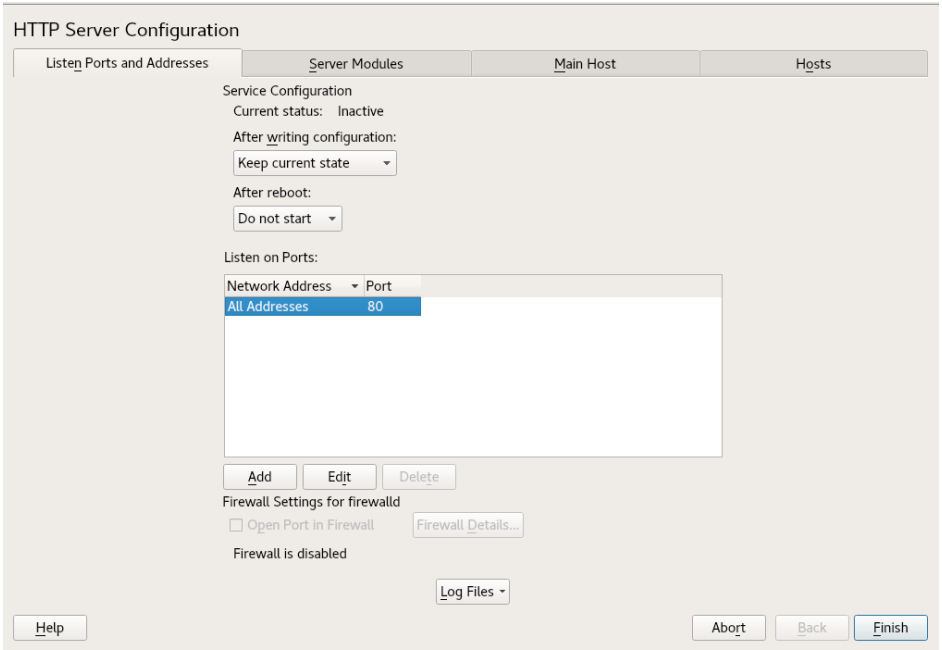


FIGURE 24.3: HTTP SERVER CONFIGURATION: LISTEN PORTS AND ADDRESSES

24.2.3.2.2 Server Modules

You can change the status (enabled or disabled) of Apache2 modules by clicking *Toggle Status*. Click *Add Module* to add a new module that is already installed but not yet listed. Learn more about modules in *Section 24.4, "Installing, Activating, and Configuring Modules"*.

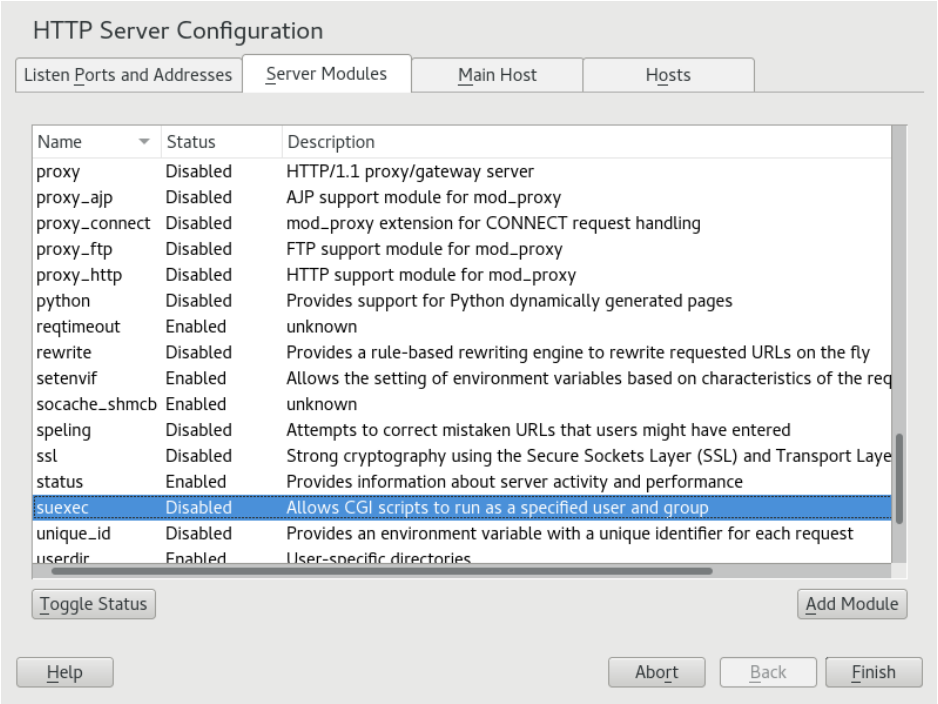


FIGURE 24.4: HTTP SERVER CONFIGURATION: SERVER MODULES

24.2.3.2.3 Main Host or Hosts

These dialogs are identical to the ones already described. Refer to *Section 24.2.3.1.3, "Default Host"* and *Section 24.2.3.1.4, "Virtual Hosts"*.

24.3 Starting and Stopping Apache

If configured with YaST as described in *Section 24.2.3, "Configuring Apache with YaST"*, Apache is started at boot time in the `multi-user.target` and `graphical.target`. You can change this behavior using YaST's *Services Manager* or with the **`systemctl`** command line tool (**`systemctl enable`** or **`systemctl disable`**).

To start, stop, or manipulate Apache on a running system, use either the **`systemctl`** or the **`apachectl`** commands as described below.

For general information about **`systemctl`** commands, refer to *Section 10.2.1, "Managing Services in a Running System"*.

`systemctl status apache2`

Checks if Apache is started.

`systemctl start apache2`

Starts Apache if it is not already running.

`systemctl stop apache2`

Stops Apache by terminating the parent process.

`systemctl restart apache2`

Stops and then restarts Apache. Starts the Web server if it was not running before.

`systemctl try-restart apache2`

Stops then restarts Apache only if it is already running.

`systemctl reload apache2`

Stops the Web server by advising all forked Apache processes to first finish their requests before shutting down. As each process dies, it is replaced by a newly started one, resulting in a complete "restart" of Apache.

Tip: Restarting Apache in Production Environments

This command allows activating changes in the Apache configuration without causing connection break-offs.

`systemctl stop apache2`

Stops the Web server after a defined period of time configured with `GracefulShutdownTimeout` to ensure that existing requests can be finished.

`apachectl configtest`

Checks the syntax of the configuration files without affecting a running Web server. Because this check is forced every time the server is started, reloaded, or restarted, it is usually not necessary to run the test explicitly (if a configuration error is found, the Web server is not started, reloaded, or restarted).

`apachectl status` and **`apachectl fullstatus`**

Dumps a short or full status screen, respectively. Requires the module `mod_status` to be enabled and a text-based browser (such as **`links`** or **`w3m`**) to be installed. In addition to that, `STATUS` must be added to `APACHE_SERVER_FLAGS` in the file `/etc/sysconfig/apache2`.

Tip: Additional Flags

If you specify additional flags to the commands, these are passed through to the Web server.

24.4 Installing, Activating, and Configuring Modules

The Apache software is built in a modular fashion: all functionality except some core tasks are handled by modules. This has progressed so far that even HTTP is processed by a module (`http_core`).

Apache modules can be compiled into the Apache binary at build time or be dynamically loaded at runtime. Refer to *Section 24.4.2, "Activation and Deactivation"* for details of how to load modules dynamically.

Apache modules can be divided into four different categories:

Base Modules

Base modules are compiled into Apache by default. Apache in openSUSE Leap has only `mod_so` (needed to load other modules) and `http_core` compiled in. All others are available as shared objects: rather than being included in the server binary itself, they can be included at runtime.

Extension Modules

In general, modules labeled as extensions are included in the Apache software package, but are usually not compiled into the server statically. In openSUSE Leap, they are available as shared objects that can be loaded into Apache at runtime.

External Modules

Modules labeled external are not included in the official Apache distribution. However, openSUSE Leap provides several of them.

Multiprocessing Modules (MPMs)

MPMs are responsible for accepting and handling requests to the Web server, representing the core of the Web server software.

24.4.1 Module Installation

If you have done a default installation as described in *Section 24.1.2, "Installation"*, the following modules are already installed: all base and extension modules, the multiprocessing module Prefork MPM, and the external module `mod_python`.

You can install additional external modules by starting YaST and choosing *Software* › *Software Management*. Now choose *View* › *Search* and search for `apache`. Among other packages, the results list contains all available external Apache modules.

24.4.2 Activation and Deactivation

Activate or deactivate particular modules either manually or with YaST. In YaST, script language modules (PHP 5 and Python) need to be enabled or disabled with the module configuration described in *Section 24.2.3.1, "HTTP Server Wizard"*. All other modules can be enabled or disabled as described in *Section 24.2.3.2.2, "Server Modules"*.

If you prefer to activate or deactivate the modules manually, use the commands **`a2enmod`** *`MODULE`* or **`a2dismod`** *`MODULE`*, respectively. **`a2enmod -l`** outputs a list of all currently active modules.

Important: Including Configuration Files for External Modules

If you have activated external modules manually, make sure to load their configuration files in all virtual host configurations. Configuration files for external modules are located under `/etc/apache2/conf.d/` and are loaded in `/etc/apache2/default-server.conf` by default. For more fine-grained control you can comment out the inclusion in `/etc/apache2/default-server.conf` and add it to specific virtual hosts only. See `/etc/apache2/vhosts.d/vhost.template` for examples.

24.4.3 Base and Extension Modules

All base and extension modules are described in detail in the Apache documentation. Only a brief description of the most important modules is available here. Refer to http://httpd.apache.org/docs/2.4/mod/ to learn details about each module.

`mod_actions`

Provides methods to execute a script whenever a certain MIME type (such as `application/pdf`), a file with a specific extension (like `.rpm`), or a certain request method (such as `GET`) is requested. This module is enabled by default.

`mod_alias`

Provides `Alias` and `Redirect` directives with which you can map a URL to a specific directory (`Alias`) or redirect a requested URL to another location. This module is enabled by default.

`mod_auth*`

The authentication modules provide different authentication methods: basic authentication with `mod_auth_basic` or digest authentication with `mod_auth_digest`. `mod_auth_basic` and `mod_auth_digest` must be combined with an authentication provider module, `mod_authn_*` (for example, `mod_authn_file` for text file–based authentication) and with an authorization module `mod_authz_*` (for example, `mod_authz_user` for user authorization).

More information about this topic is available in the *Authentication HOWTO* at http://httpd.apache.org/docs/2.4/howto/auth.html.

`mod_autoindex`

Autoindex generates directory listings when no index file (for example, `index.html`) is present. The look and feel of these indexes is configurable. This module is enabled by default. However, directory listings are disabled by default via the `Options` directive—overwrite this setting in your virtual host configuration. The default configuration file for this module is located at `/etc/apache2/mod_autoindex-defaults.conf`.

`mod_cgi`

`mod_cgi` is needed to execute CGI scripts. This module is enabled by default.

`mod_deflate`

Using this module, Apache can be configured to compress given file types on the fly before delivering them.

`mod_dir`

`mod_dir` provides the `DirectoryIndex` directive with which you can configure which files are automatically delivered when a directory is requested (`index.html` by default). It also provides an automatic redirect to the correct URL when a directory request does not contain a trailing slash. This module is enabled by default.

`mod_env`

Controls the environment that is passed to CGI scripts or SSI pages. Environment variables can be set or unset or passed from the shell that invoked the `httpd` process. This module is enabled by default.

`mod_expires`

With `mod_expires`, you can control how often proxy and browser caches refresh your documents by sending an `Expires` header. This module is enabled by default.

`mod_http2`

With `mod_http2`, Apache gains support for the HTTP/2 protocol. It can be enabled by specifying `Protocols h2 http/1.1` in a `VirtualHost`.

`mod_include`

`mod_include` lets you use Server Side Includes (SSI), which provide a basic functionality to generate HTML pages dynamically. This module is enabled by default.

`mod_info`

Provides a comprehensive overview of the server configuration under http://localhost/server-info/. For security reasons, you should always limit access to this URL. By default only `localhost` is allowed to access this URL. `mod_info` is configured at `/etc/apache2/mod_info.conf`.

`mod_log_config`

With this module, you can configure the look of the Apache log files. This module is enabled by default.

`mod_mime`

The mime module ensures that a file is delivered with the correct MIME header based on the file name's extension (for example `text/html` for HTML documents). This module is enabled by default.

`mod_negotiation`

Necessary for content negotiation. See http://httpd.apache.org/docs/2.4/content-negotiation.html for more information. This module is enabled by default.

`mod_rewrite`

Provides the functionality of `mod_alias`, but offers more features and flexibility. With `mod_rewrite`, you can redirect URLs based on multiple rules, request headers, and more.

`mod_setenvif`

Sets environment variables based on details of the client's request, such as the browser string the client sends, or the client's IP address. This module is enabled by default.

`mod_spelling`

`mod_spelling` attempts to automatically correct typographical errors in URLs, such as capitalization errors.

`mod_ssl`

Enables encrypted connections between Web server and clients. See *Section 24.6, "Setting Up a Secure Web Server with SSL"* for details. This module is enabled by default.

`mod_status`

Provides information on server activity and performance under http://localhost/server-status/. For security reasons, you should always limit access to this URL. By default, only `localhost` is allowed to access this URL. `mod_status` is configured at `/etc/apache2/mod_status.conf`.

`mod_suexec`

`mod_suexec` lets you run CGI scripts under a different user and group. This module is enabled by default.

`mod_userdir`

Enables user-specific directories available under *`~USER/`*. The `UserDir` directive must be specified in the configuration. This module is enabled by default.

24.4.4 Multiprocessing Modules

openSUSE Leap provides two different multiprocessing modules (MPMs) for use with Apache:

- *Prefork MPM*
- *Worker MPM*

24.4.4.1 Prefork MPM

The prefork MPM implements a non-threaded, preforking Web server. It makes the Web server behave similarly to Apache version 1.x. In this version it isolates each request and handles it by forking a separate child process. Thus problematic requests cannot affect others, avoiding a lockup of the Web server.

While providing stability with this process-based approach, the prefork MPM consumes more system resources than its counterpart, the worker MPM. The prefork MPM is considered the default MPM for Unix-based operating systems.

Important: MPMs in This Document

This document assumes Apache is used with the prefork MPM.

24.4.4.2 Worker MPM

The worker MPM provides a multi-threaded Web server. A thread is a “lighter” form of a process. The advantage of a thread over a process is its lower resource consumption. Instead of only forking child processes, the worker MPM serves requests by using threads with server processes. The preforked child processes are multi-threaded. This approach makes Apache perform better by consuming fewer system resources than the prefork MPM.

One major disadvantage is the stability of the worker MPM: if a thread becomes corrupt, all threads of a process can be affected. In the worst case, this may result in a server crash. Especially when using the Common Gateway Interface (CGI) with Apache under heavy load, internal server errors might occur because of threads being unable to communicate with system resources. Another argument against using the worker MPM with Apache is that not all available Apache modules are thread-safe and thus cannot be used with the worker MPM.

Warning: Using PHP Modules with MPMs

Not all available PHP modules are thread-safe. Using the worker MPM with `mod_php` is strongly discouraged.

24.4.5 External Modules

Find a list of all external modules shipped with openSUSE Leap here. Find the module's documentation in the listed directory.

`mod_apparmor`

Adds support to Apache to provide AppArmor confinement to individual CGI scripts handled by modules like `mod_php5`.

Package Name: `apache2-mod_apparmor`

More Information: *Book* "Security Guide"

`mod_php5`

PHP is a server-side, cross-platform HTML embedded scripting language.

Package Name: `apache2-mod_php5`

Configuration File: `/etc/apache2/conf.d/php5.conf`

More Information: `/usr/share/doc/packages/apache2-mod_php5`

`mod_python`

`mod_python` allows embedding Python within the Apache HTTP server for a considerable boost in performance and added flexibility in designing Web-based applications.

Package Name: `apache2-mod_python`

More Information: `/usr/share/doc/packages/apache2-mod_python`

`mod_security`

`mod_security` provides a Web application firewall to protect Web applications from a range of attacks. It also enables HTTP traffic monitoring and real-time analysis.

Package Name: `apache2-mod_security2`

Configuration File: `/etc/apache2/conf.d/mod_security2.conf`

More Information: `/usr/share/doc/packages/apache2-mod_security2`

Documentation: http://modsecurity.org/documentation/

24.4.6 Compilation

Apache can be extended by advanced users by writing custom modules. To develop modules for Apache or compile third-party modules, the package `apache2-devel` is required along with the corresponding development tools. `apache2-devel` also contains the **`apxs2`** tools, which are necessary for compiling additional modules for Apache.

`apxs2` enables the compilation and installation of modules from source code (including the required changes to the configuration files), which creates *dynamic shared objects* (DSOs) that can be loaded into Apache at runtime.

The **apxs2** binaries are located under /usr/sbin:

- /usr/sbin/apxs2—suitable for building an extension module that works with any MPM. The installation location is /usr/lib64/apache2.
- /usr/sbin/apxs2-prefork—suitable for prefork MPM modules. The installation location is /usr/lib64/apache2-prefork.
- /usr/sbin/apxs2-worker—suitable for worker MPM modules. The installation location is /usr/lib64/apache2-worker.

Install and activate a module from source code with the following commands:

```
tux > sudo cd /path/to/module/source
tux > sudo apxs2 -cia MODULE.c
```

where -c compiles the module, -i installs it, and -a activates it. Other options of **apxs2** are described in the apxs2(1) man page.

24.5 Enabling CGI Scripts

Apache's Common Gateway Interface (CGI) lets you create dynamic content with programs or scripts usually called CGI scripts. CGI scripts can be written in any programming language. Usually, script languages such as PHP are used.

To enable Apache to deliver content created by CGI scripts, mod_cgi needs to be activated. mod_alias is also needed. Both modules are enabled by default. Refer to *Section 24.4.2, "Activation and Deactivation"* for details on activating modules.

Warning: CGI Security

Allowing the server to execute CGI scripts is a potential security hole. Refer to *Section 24.8, "Avoiding Security Problems"* for additional information.

24.5.1 Apache Configuration

In openSUSE Leap, the execution of CGI scripts is only allowed in the directory /srv/www/cgi-bin/. This location is already configured to execute CGI scripts. If you have created a virtual host configuration (see *Section 24.2.2.1, "Virtual Host Configuration"*) and want to place your scripts in a host-specific directory, you must unlock and configure this directory.

EXAMPLE 24.5: VIRTUALHOST CGI CONFIGURATION

```
ScriptAlias /cgi-bin/ "/srv/www/www.example.com/cgi-bin/" (1)

<Directory "/srv/www/www.example.com/cgi-bin/">
 Options +ExecCGI (2)
 AddHandler cgi-script .cgi .pl (3)
 Require all granted (4)
</Directory>
```

(1) Tells Apache to handle all files within this directory as CGI scripts.

(2) Enables CGI script execution

(3) Tells the server to treat files with the extensions .pl and .cgi as CGI scripts. Adjust according to your needs.

(4) The `Require` directive controls the default access state. In this case, access is granted to the specified directory without limitation. For more information on authentication and authorization, see http://httpd.apache.org/docs/2.4/howto/auth.html.

24.5.2 Running an Example Script

CGI programming differs from "regular" programming in that the CGI programs and scripts must be preceded by a MIME-Type header such as `Content-type: text/html`. This header is sent to the client, so it understands what kind of content it receives. Secondly, the script's output must be something the client, usually a Web browser, understands—HTML usually, or plain text or images, for example.

A simple test script available under `/usr/share/doc/packages/apache2/test-cgi` is part of the Apache package. It outputs the content of some environment variables as plain text. Copy this script to either `/srv/www/cgi-bin/` or the script directory of your virtual host (`/srv/www/www.example.com/cgi-bin/`) and name it `test.cgi`. Edit the file to have `#!/bin/sh` as the first line.

Files accessible by the Web server should be owned by the user `root`. For additional information see *Section 24.8, "Avoiding Security Problems"*. Because the Web server runs with a different user, the CGI scripts must be world-executable and world-readable. Change into the CGI directory and use the command **`chmod 755 test.cgi`** to apply the proper permissions.

Now call `http://localhost/cgi-bin/test.cgi` or `http://www.example.com/cgi-bin/test.cgi`. You should see the "CGI/1.0 test script report".

24.5.3 CGI Troubleshooting

If you do not see the output of the test program but an error message instead, check the following:

CGI TROUBLESHOOTING

- *Have you reloaded the server after having changed the configuration?* If not, reload with `systemctl reload apache2`.
- *If you have configured your custom CGI directory, is it configured properly?* If in doubt, try the script within the default CGI directory `/srv/www/cgi-bin/` and call it with `http://localhost/cgi-bin/test.cgi`.
- *Are the file permissions correct?* Change into the CGI directory and execute `ls -l test.cgi`. The output should start with

```
-rwxr-xr-x  1 root root
```

- Make sure that the script does not contain programming errors. If you have not changed `test.cgi`, this should not be the case, but if you are using your own programs, always make sure that they do not contain programming errors.

24.6 Setting Up a Secure Web Server with SSL

Whenever sensitive data, such as credit card information, is transferred between Web server and client, it is desirable to have a secure, encrypted connection with authentication. `mod_ssl` provides strong encryption using the secure sockets layer (SSL) and transport layer security (TLS) protocols for HTTP communication between a client and the Web server. Using TLS/SSL, a private connection between Web server and client is established. Data integrity is ensured and client and server can authenticate each other.

For this purpose, the server sends an SSL certificate that holds information proving the server's valid identity before any request to a URL is answered. In turn, this guarantees that the server is the uniquely correct end point for the communication. Additionally, the certificate generates an encrypted connection between client and server that can transport information without the risk of exposing sensitive, plain-text content.

`mod_ssl` does not implement the TLS/SSL protocols itself, but acts as an interface between Apache and an SSL library. In openSUSE Leap, the OpenSSL library is used. OpenSSL is automatically installed with Apache.

The most visible effect of using `mod_ssl` with Apache is that URLs are prefixed with `https://` instead of `http://`.

24.6.1 Creating an SSL Certificate

To use TLS/SSL with the Web server, you need to create an SSL certificate. This certificate is needed for the authorization between Web server and client, so that each party can clearly identify the other party. To ensure the integrity of the certificate, it must be signed by a party every user trusts.

There are three types of certificates you can create: a "dummy" certificate for testing purposes only, a self-signed certificate for a defined circle of users that trust you, and a certificate signed by an independent, publicly-known certificate authority (CA).

Creating a certificate is a two step process. First, a private key for the certificate authority is generated then the server certificate is signed with this key.

Tip: For More Information

To learn more about concepts and definitions of TLS/SSL, refer to http://httpd.apache.org/docs/2.4/ssl/ssl_intro.html.

24.6.1.1 Creating a "Dummy" Certificate

To generate a dummy certificate, call the script **`/usr/bin/gensslcert`**. It creates or overwrites the files listed below. Use **`gensslcert`**'s optional switches to fine-tune the certificate. Call **`/usr/bin/gensslcert`** `-h` for more information.

- `/etc/apache2/ssl.crt/ca.crt`
- `/etc/apache2/ssl.crt/server.crt`
- `/etc/apache2/ssl.key/server.key`
- `/etc/apache2/ssl.csr/server.csr`

A copy of `ca.crt` is also placed at `/srv/www/htdocs/CA.crt` for download.

Important: For Testing Purposes Only

A dummy certificate should never be used on a production system. Only use it for testing purposes.

24.6.1.2 Creating a Self-Signed Certificate

If you are setting up a secure Web server for an intranet or for a defined circle of users, it is probably sufficient if you sign a certificate with your own certificate authority (CA). Note that visitors to such a site will see a warning like “this is an untrusted site”, as Web browsers do not recognize self-signed certificates.

Important: Self-Signed Certificates

Only use a self-signed certificate on a Web server that is accessed by people who know and trust you as a certificate authority. It is not recommended to use such a certificate for a public shop, for example.

First you need to generate a certificate signing request (CSR). You are going to use **`openssl`**, with `PEM` as the certificate format. During this step, you will be asked for a passphrase, and to answer several questions. Remember the passphrase you enter as you will need it in the future.

```
tux > sudo openssl req -new > new.cert.csr
Generating a 1024 bit RSA private key
..++++++
.........++++++
writing new private key to 'privkey.pem'
Enter PEM pass phrase:❶
Verifying - Enter PEM pass phrase:❷
-----
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]:❸
State or Province Name (full name) [Some-State]:❹
Locality Name (eg, city) []:❺
```

```
Organization Name (eg, company) [Internet Widgits Pty Ltd]: 6
Organizational Unit Name (eg, section) []: 7
Common Name (for example server FQDN, or YOUR name) []: 8
Email Address []: 9

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: 10
An optional company name []: 11
```

1. Fill in your passphrase,
2. ...fill it in once more (and remember it).
3. Fill in your 2 letter country code, such as `GB` or `CZ`.
4. Fill in the name of the state where you live.
5. Fill in the city name, such as `Prague`.
6. Fill in the name of the organization you work for.
7. Fill in your organization unit, or leave blank if you have none.
8. Fill in either the domain name of the server, or your first and last name.
9. Fill in your work e-mail address.
10. Leave the challenge password empty, otherwise you will need to enter it every time you restart the Apache Web server.
11. Fill in the optional company name, or leave blank.

Now you can generate the certificate. You are going to use **`openssl`** again, and the format of the certificate is the default `PEM`.

PROCEDURE 24.3: GENERATING THE CERTIFICATE

1. Export the private part of the key to `new.cert.key`. You will be prompted for the passphrase you entered when creating the certificate signing request (CSR).

   ```
   tux > sudo openssl rsa -in privkey.pem -out new.cert.key
   ```

2. Generate the public part of the certificate according to the information you filled out in the signing request. The `-days` option specifies the length of time before the certificate expires. You can revoke a certificate, or replace one before it expires.

   ```
   tux > sudo openssl x509 -in new.cert.csr -out new.cert.cert -req \
   -signkey new.cert.key -days 365
   ```

3. Copy the certificate files to the relevant directories, so that the Apache server can read them. Make sure that the private key `/etc/apache2/ssl.key/server.key` is not world-readable, while the public PEM certificate `/etc/apache2/ssl.crt/server.crt` is.

```
tux > sudo cp new.cert.cert /etc/apache2/ssl.crt/server.crt
tux > sudo cp new.cert.key /etc/apache2/ssl.key/server.key
```

Tip: Public Certificate Location

The last step is to copy the public certificate file from `/etc/apache2/ssl.crt/server.crt` to a location where your users can access it to incorporate it into the list of known and trusted CAs in their Web browsers. Otherwise a browser complains that the certificate was issued by an unknown authority.

24.6.1.3 Getting an Officially Signed Certificate

There are several official certificate authorities that sign your certificates. The certificate is signed by a trustworthy third party, so can be fully trusted. Publicly operating secure Web servers usually have an officially signed certificate. A list of the most used Certificate Authorities (CAs) is available at https://en.wikipedia.org/wiki/Certificate_authority#Providers.

When requesting an officially signed certificate, you do not send a certificate to the CA. Instead, issue a Certificate Signing Request (CSR). To create a CSR, run the following command:

```
tux > openssl req -new -newkey rsa:2048 -nodes -keyout newkey.pem -out newreq.pem
```

You are asked to enter a distinguished name. This requires you to answer a few questions, such as country name or organization name. Enter valid data—everything you enter here later shows up in the certificate and is checked. You do not need to answer every question. If one does not apply to you or you want to leave it blank, use ".". Common name is the name of the CA itself—choose a significant name, such as *`My company`* CA. Last, a challenge password and an alternative company name must be entered.

Find the CSR in the directory from which you called the script. The file is named `newreq.pem`.

24.6.2 Configuring Apache with SSL

The default port for TLS/SSL requests on the Web server side is 443. There is no conflict between a "regular" Apache listening on port 80 and an TLS/SSL-enabled Apache listening on port 443. In fact, HTTP and HTTPS can be run with the same Apache instance. Usually separate virtual hosts are used to dispatch requests to port 80 and port 443 to separate virtual servers.

Important: Firewall Configuration

Do not forget to open the firewall for SSL-enabled Apache on port 443. This can be done with `firewalld` as described in *Book "Security Guide", Chapter 16 "Masquerading and Firewalls", Section 16.4.1 "Configuring the Firewall on the Command Line"*.

The SSL module is enabled by default in the global server configuration. In case it has been disabled on your host, activate it with the following command: **`a2enmod ssl`**. To finally enable SSL, the server needs to be started with the flag "SSL". To do so, call **`a2enflag SSL`** (case-sensitive!). If you have chosen to encrypt your server certificate with a password, you should also increase the value for `APACHE_TIMEOUT` in `/etc/sysconfig/apache2`, so you have enough time to enter the passphrase when Apache starts. Restart the server to make these changes active. A reload is not sufficient.

The virtual host configuration directory contains a template `/etc/apache2/vhosts.d/vhost-ssl.template` with SSL-specific directives that are extensively documented. Refer to *Section 24.2.2.1, "Virtual Host Configuration"* for the general virtual host configuration.

To get started, copy the template to `/etc/apache2/vhosts.d/MYSSL-HOST.conf` and edit it. Adjusting the values for the following directives should be sufficient:

- `DocumentRoot`
- `ServerName`
- `ServerAdmin`
- `ErrorLog`
- `TransferLog`

24.6.2.1 Name-Based Virtual Hosts and SSL

By default it is not possible to run multiple SSL-enabled virtual hosts on a server with only one IP address. Name-based virtual hosting requires that Apache knows which server name has been requested. The problem with SSL connections is, that such a request can only be read after the SSL connection has already been established (by using the default virtual host). As a result, users will receive a warning message stating that the certificate does not match the server name.

openSUSE Leap comes with an extension to the SSL protocol called Server Name Indication (SNI) addresses this issue by sending the name of the virtual domain as part of the SSL negotiation. This enables the server to "switch" to the correct virtual domain early and present the browser the correct certificate.

SNI is enabled by default on openSUSE Leap. To enable Name-Based Virtual Hosts for SSL, configure the server as described in *Section 24.2.2.1.1, "Name-Based Virtual Hosts"* (note that you need to use port `443` rather than port `80` with SSL).

Important: SNI Browser Support

SNI must also be supported on the client side. However, SNI is supported by most browsers, except for certain older browsers. For more information, see https://en.wikipedia.org/wiki/Server_Name_Indication#Support.

To configure handling of non-SNI capable browsers, use the directive `SSLStrictSNIVHostCheck`. When set to `on` in the server configuration, non-SNI capable browser will be rejected for all virtual hosts. When set to `on` within a `VirtualHost` directive, access to this particular host will be rejected.

When set to `off` in the server configuration, the server will behave as if not having SNI support. SSL requests will be handled by the *first* virtual host defined (for port 443).

24.7 Running Multiple Apache Instances on the Same Server

As of openSUSE® Leap 42.1, you can run multiple Apache instances on the same server. This has several advantages over running multiple virtual hosts (see *Section 24.2.2.1, "Virtual Host Configuration"*):

- When a virtual host needs to be disabled for some time, you need to change the Web server configuration and restart it so that the change takes effect.
- In case of problems with one virtual host, you need to restart all of them.

You can run the default Apache instance as usual:

```
tux > sudo systemctl start apache2
```

It reads the default `/etc/sysconfig/apache2` file. If the file is not present, or it is present but it does not set the `APACHE_HTTPD_CONF` variable, it reads `/etc/apache2/httpd.conf`.

To activate another Apache instance, run:

```
tux > sudo systemctl start apache2@INSTANCE_NAME
```

For example:

```
tux > sudo systemctl start apache2@example_web.org
```

By default, the instance uses `/etc/apache2@example_web.org/httpd.conf` as a main configuration file, which can be overwritten by setting `APACHE_HTTPD_CONF` in `/etc/sysconfig/apache2@example_web.org`.

An example to set up an additional instance of Apache follows. Note that you need to execute all the commands as `root`.

PROCEDURE 24.4: CONFIGURING AN ADDITIONAL APACHE INSTANCE

1. Create a new configuration file based on `/etc/sysconfig/apache2`, for example `/etc/sysconfig/apache2@example_web.org`:

   ```
   tux > sudo cp /etc/sysconfig/apache2 /etc/sysconfig/apache2@example_web.org
   ```

2. Edit the file `/etc/sysconfig/apache2@example_web.org` and change the line containing

```
APACHE_HTTPD_CONF
```

to

```
APACHE_HTTPD_CONF="/etc/apache2/httpd@example_web.org.conf"
```

3. Create the file `/etc/apache2/httpd@example_web.org.conf` based on `/etc/apache2/httpd.conf`.

```
tux > sudo cp /etc/apache2/httpd.conf /etc/apache2/httpd@example_web.org.conf
```

4. Edit `/etc/apache2/httpd@example_web.org.conf` and change

```
Include /etc/apache2/listen.conf
```

to

```
Include /etc/apache2/listen@example_web.org.conf
```

Review all the directives and change them to fit your needs. You will probably want to change

```
Include /etc/apache2/global.conf
```

and create new `global@example_web.org.conf` for each instance. We suggest to change

```
ErrorLog /var/log/apache2/error_log
```

to

```
ErrorLog /var/log/apache2/error@example_web.org_log
```

to have separate logs for each instance.

5. Create `/etc/apache2/listen@example_web.org.conf` based on `/etc/apache2/listen.conf`.

```
tux > sudo cp /etc/apache2/listen.conf /etc/apache2/listen@example_web.org.conf
```

6. Edit `/etc/apache2/listen@example_web.org.conf` and change

```
Listen 80
```

to the port number you want the new instance to run on, for example 82:

```
Listen 82
```

To run the new Apache instance over a secured protocol (see *Section 24.6, "Setting Up a Secure Web Server with SSL"*), change also the line

```
Listen 443
```

for example to

```
Listen 445
```

7. Start the new Apache instance:

```
tux > sudo systemctl start apache2@example_web.org
```

8. Check if the server is running by pointing your Web browser at `http://server_name:82`. If you previously changed the name of the error log file for the new instance, you can check it:

```
tux > sudo tail -f /var/log/apache2/error@example_web.org_log
```

Here are several points to consider when setting up more Apache instances on the same server:

- The file `/etc/sysconfig/apache2@INSTANCE_NAME` can include the same variables as `/etc/sysconfig/apache2`, including module loading and MPM setting.
- The default Apache instance does not need to be running while other instances run.
- The Apache helper utilities **`a2enmod`**, **`a2dismod`** and **`apachectl`** operate on the default Apache instance if not specified otherwise with the `HTTPD_INSTANCE` environment variable. The following example

```
tux > sudo export HTTPD_INSTANCE=example_web.org
tux > sudo a2enmod access_compat
tux > sudo a2enmod status
tux > sudo apachectl start
```

will add `access_compat` and `status` modules to the `APACHE_MODULES` variable of `/etc/sysconfig/apache2@example_web.org`, and then start the `example_web.org` instance.

24.8 Avoiding Security Problems

A Web server exposed to the public Internet requires an ongoing administrative effort. It is inevitable that security issues appear, both related to the software and to accidental misconfiguration. Here are some tips for how to deal with them.

24.8.1 Up-to-Date Software

If there are vulnerabilities found in the Apache software, a security advisory will be issued by SUSE. It contains instructions for fixing the vulnerabilities, which in turn should be applied when possible. The SUSE security announcements are available from the following locations:

- **Web Page.** 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 PHP, SSI, or any other programming language can essentially run arbitrary commands and therefore present a general security issue. Scripts that will be executed from the server should only be installed from sources the server administrator trusts—allowing users to run their own scripts is generally not a good idea. It is also recommended to do security audits for all scripts.

To make the administration of scripts as easy as possible, it is common practice to limit the execution of CGI scripts to specific directories instead of globally allowing them. The directives `ScriptAlias` and `Option ExecCGI` are used for configuration. The openSUSE Leap default configuration does not allow execution of CGI scripts from everywhere.

All CGI scripts run as the same user, so different scripts can potentially conflict with each other. The module suEXEC lets you run CGI scripts under a different user and group.

24.8.5 User Directories

When enabling user directories (with `mod_userdir` or `mod_rewrite`) you should strongly consider not allowing `.htaccess` files, which would allow users to overwrite security settings. At least you should limit the user's engagement by using the directive `AllowOverRide`. In openSUSE Leap, `.htaccess` files are enabled by default, but the user is not allowed to overwrite any `Option` directives when using `mod_userdir` (see the `/etc/apache2/mod_userdir.conf` configuration file).

24.9 Troubleshooting

If Apache does not start, the Web page is not accessible, or users cannot connect to the Web server, it is important to find the cause of the problem. Here are some typical places to look for error explanations and important things to check:

Output of the `apache2.service` subcommand:

Instead of starting and stopping the Web server with the binary `/usr/sbin/apache2ctl`, rather use the **`systemctl`** commands instead (described in *Section 24.3, "Starting and Stopping Apache"*). **`systemctl status apache2`** is verbose about errors, and it even provides tips and hints for fixing configuration errors.

Log Files and Verbosity

In case of both fatal and nonfatal errors, check the Apache log files for causes, mainly the error log file located at `/var/log/apache2/error_log` by default. Additionally, you can control the verbosity of the logged messages with the `LogLevel` directive if more detail is needed in the log files.

Tip: A Simple Test

Watch the Apache log messages with the command **`tail -F /var/log/apache2/`***`MY_ERROR_LOG`*. Then run **`systemctl restart apache2`**. Now, try to connect with a browser and check the output.

Firewall and Ports

A common mistake is to not open the ports for Apache in the firewall configuration of the server. If you configure Apache with YaST, there is a separate option available to take care of this specific issue (see *Section 24.2.3, "Configuring Apache with YaST"*). If you are configuring Apache manually, open firewall ports for HTTP and HTTPS via YaST's firewall module.

If the error cannot be tracked down with any of these, check the online Apache bug database at http://httpd.apache.org/bug_report.html. Additionally, the Apache user community can be reached via a mailing list available at http://httpd.apache.org/userslist.html.

24.10 For More Information

The package `apache2-doc` contains the complete Apache manual in various localizations for local installation and reference. It is not installed by default—the quickest way to install it is to use the command **`zypper in apache2-doc`**. having been installed, the Apache manual is available at http://localhost/manual/. You may also access it on the Web at http://httpd.apache.org/docs-2.4/. SUSE-specific configuration hints are available in the directory `/usr/share/doc/packages/apache2/README.*`.

24.10.1 Apache 2.4

For a list of new features in Apache 2.4, refer to http://httpd.apache.org/docs/2.4/new_features_2_4.html. Information about upgrading from version 2.2 to 2.4 is available at http://httpd.apache.org/docs-2.4/upgrading.html.

24.10.2 Apache Modules

More information about external Apache modules that are briefly described in *Section 24.4.5, "External Modules"* is available at the following locations:

`mod_apparmor`
: http://en.opensuse.org/SDB:AppArmor

`mod_php5`
: http://www.php.net/manual/en/install.unix.apache2.php

`mod_python`
: http://www.modpython.org/

`mod_security`
: http://modsecurity.org/

24.10.3 Development

More information about developing Apache modules or about getting involved in the Apache Web server project are available at the following locations:

Apache Developer Information
: http://httpd.apache.org/dev/

Apache Developer Documentation
: http://httpd.apache.org/docs/2.4/developer/

25 Setting Up an FTP Server with YaST

Using the YaST *FTP Server* module, you can configure your machine to function as an FTP (File Transfer Protocol) server. Anonymous and/or authenticated users can connect to your machine and download files using the FTP protocol. Depending on the configuration, they can also upload files to the FTP server. YaST uses vsftpd (Very Secure FTP Daemon).

If the YaST FTP Server module is not available in your system, install the `yast2-ftp-server` package.

To configure the FTP server using YaST, follow these steps:

1. Open the YaST control center and choose *Network Services* › *FTP Server* or run the **`yast2 ftp-server`** command as `root`.
2. If there is not any FTP server installed in your system, you will be asked which server to install when the YaST FTP Server module starts. Choose the vsftpd server and confirm the dialog.
3. In the *Start-Up* dialog, configure the options for starting of the FTP server. For more information, see *Section 25.1, "Starting the FTP Server"*.
   In the *General* dialog, configure FTP directories, welcome message, file creation masks and other parameters. For more information, see *Section 25.2, "FTP General Settings"*.
   In the *Performance* dialog, set the parameters that affect the load on the FTP server. For more information, see *Section 25.3, "FTP Performance Settings"*.
   In the *Authentication* dialog, set whether the FTP server should be available for anonymous and/or authenticated users. For more information, see *Section 25.4, "Authentication"*.
   In the *Expert Settings* dialog, configure the operation mode of the FTP server, SSL connections and firewall settings. For more information, see *Section 25.5, "Expert Settings"*.
4. Click *Finish* to save the configurations.

25.1 Starting the FTP Server

In the *Service Start* frame of the *FTP Start-Up* dialog set the way the FTP server is started up. You can choose between starting the server automatically during the system boot and starting it manually. If the FTP server should be started only after an FTP connection request, choose *Via socket*.

The current status of the FTP server is shown in the *Switch On and Off* frame of the *FTP Start-Up* dialog. Start the FTP server by clicking *Start FTP Now*. To stop the server, click *Stop FTP Now*. After having changed the settings of the server click *Save Settings and Restart FTP Now*. Your configurations will be saved by leaving the configuration module with *Finish*.

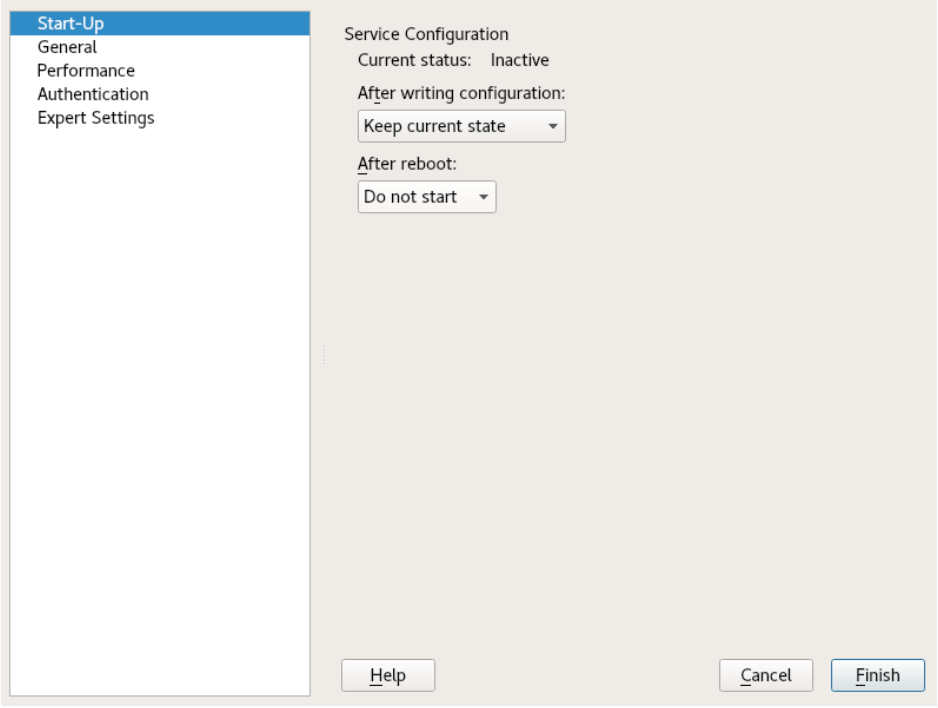


FIGURE 25.1: FTP SERVER CONFIGURATION — START-UP

25.2 FTP General Settings

In the *General Settings* frame of the *FTP General Settings* dialog you can set the *Welcome message* which is shown after connecting to the FTP server.

If you check the *Chroot Everyone* option, all local users will be placed in a chroot jail in their home directory after login. This option has security implications, especially if the users have upload permission or shell access, so be careful enabling this option.

If you check the *Verbose Logging* option, all FTP requests and responses are logged.

You can limit permissions of files created by anonymous and/or authenticated users with umask. Set the file creation mask for anonymous users in *Umask for Anonymous* and the file creation mask for authenticated users in *Umask for Authenticated Users*. The masks should be entered as octal numbers with a leading zero. For more information about umask, see the umask man page (`man 1p umask`).

In the *FTP Directories* frame set the directories used for anonymous and authorized users. With *Browse*, you can select a directory to be used from the local file system. The default FTP directory for anonymous users is `/srv/ftp`. Note that vsftpd does not allow this directory to be writable for all users. The subdirectory `upload` with write permissions for anonymous users is created instead.

25.3 FTP Performance Settings

In the *Performance* dialog set the parameters which affect the load on the FTP server. *Max Idle Time* is the maximum time (in minutes) the remote client may spend between FTP commands. In case of longer inactivity, the remote client is disconnected. *Max Clients for One IP* determines the maximum number of clients which can be connected from a single IP address. *Max Clients* determines the maximum number of clients which may be connected. Any additional clients will get an error message.

The maximum data transfer rate (in KB/s) is set in *Local Max Rate* for local authenticated users, and in *Anonymous Max Rate* for anonymous clients respectively. The default value for the rate settings is `0`, which means unlimited data transfer rate.

25.4 Authentication

In the *Enable/Disable Anonymous and Local Users* frame of the *Authentication* dialog, you can set which users are allowed to access your FTP server. You can choose between the following options: granting access to anonymous users only, to authenticated users only (with accounts on the system) or to both types of users.

To allow users to upload files to the FTP server, check *Enable Upload* in the *Uploading* frame of the *Authentication* dialog. Here you can allow uploading or creating directories even for anonymous users by checking the respective box.

Note: vsftp—Allowing File Upload for Anonymous Users

If a vsftpd server is used and you want anonymous users to be able to upload files or create directories, a subdirectory with writing permissions for all users needs to be created in the anonymous FTP directory.

25.5 Expert Settings

An FTP server can run in active or in passive mode. By default the server runs in passive mode. To switch into active mode, deselect the *Enable Passive Mode* option in the *Expert Settings* dialog. You can also change the range of ports on the server used for the data stream by tweaking the *Min Port for Pas. Mode* and *Max Port for Pas. Mode* options.

If you want encrypted communication between clients and the server, you can *Enable SSL*. Check the versions of the protocol to be supported and specify the DSA certificate to be used for SSL encrypted connections.

If your system is protected by a firewall, check *Open Port in Firewall* to enable a connection to the FTP server.

25.6 For More Information

For more information about the FTP server read the manual pages of **`vsftpd`** and **`vsftpd.conf`**.

26 Squid Caching Proxy Server

Squid is a widely-used caching proxy server for Linux and Unix platforms. This means that it stores requested Internet objects, such as data on a Web or FTP server, on a machine that is closer to the requesting workstation than the server. It can be set up in multiple hierarchies to assure optimal response times and low bandwidth usage, even in modes that are transparent to end users.

Squid acts as a caching proxy server. It redirects object requests from clients (in this case, from Web browsers) to the server. When the requested objects arrive from the server, it delivers the objects to the client and keeps a copy of them in the hard disk cache. An advantage of caching is that several clients requesting the same object can be served from the hard disk cache. This enables clients to receive the data much faster than from the Internet. This procedure also reduces the network traffic.

Along with actual caching, Squid offers a wide range of features:

- Distributing load over intercommunicating hierarchies of proxy servers
- Defining strict access control lists for all clients accessing the proxy server
- Allowing or denying access to specific Web pages using other applications
- Generating statistics about frequently-visited Web pages for the assessment of surfing habits

Squid is not a generic proxy server. It normally proxies only HTTP connections. It supports the protocols FTP, Gopher, SSL, and WAIS, but it does not support other Internet protocols, such as the news protocol, or video conferencing protocols. Because Squid only supports the UDP protocol to provide communication between different caches, many multimedia programs are not supported.

26.1 Some Facts about Proxy Servers

As a caching proxy server, Squid can be used in several ways. When combined with a firewall, it can help with security. Multiple proxies can be used together. It can also determine what types of objects should be cached and for how long.

26.1.1 Squid and Security

It is possible to use Squid together with a firewall to secure internal networks from the outside. The firewall denies all clients access to external services except Squid. All Web connections must be established by the proxy server. With this configuration, Squid completely controls Web access.

If the firewall configuration includes a demilitarized zone (DMZ), the proxy server should operate within this zone. *Section 26.6, "Configuring a Transparent Proxy"* describes how to implement a *transparent* proxy. This simplifies the configuration of the clients, because in this case, they do not need any information about the proxy server.

26.1.2 Multiple Caches

Several instances of Squid can be configured to exchange objects between them. This reduces the total system load and increases the chances of retrieving an object from the local network. It is also possible to configure cache hierarchies, so a cache can forward object requests to sibling caches or to a parent cache—causing it to request objects from another cache in the local network, or directly from the source.

Choosing the appropriate topology for the cache hierarchy is very important, because it is not desirable to increase the overall traffic on the network. For a very large network, it would make sense to configure a proxy server for every subnet and connect them to a parent proxy server, which in turn is connected to the caching proxy server of the ISP.

All this communication is handled by ICP (Internet cache protocol) running on top of the UDP protocol. Data transfers between caches are handled using HTTP (hypertext transmission protocol) based on TCP.

To find the most appropriate server from which to request objects, a cache sends an ICP request to all sibling proxies. The sibling proxies answer these requests via ICP responses. If the object was detected, they use the code `HIT`, if not, they use `MISS`.

If multiple `HIT` responses were found, the proxy server decides from which server to download, depending on factors such as which cache sent the fastest answer or which one is closer. If no satisfactory responses are received, the request is sent to the parent cache.

Note: How Squid Avoids Duplication of Objects

To avoid duplication of objects in different caches in the network, other ICP protocols are used, such as CARP (cache array routing protocol) or HTCP (hypertext cache protocol). The more objects maintained in the network, the greater the possibility of finding the desired object.

26.1.3 Caching Internet Objects

Many objects available in the network are not static, such as dynamically generated pages and TLS/SSL-encrypted content. Objects like these are not cached because they change each time they are accessed.

To determine how long objects should remain in the cache, objects are assigned one of several states. Web and proxy servers find out the status of an object by adding headers to these objects, such as “Last modified” or “Expires” and the corresponding date. Other headers specifying that objects must not be cached can be used as well.

Objects in the cache are normally replaced, because of a lack of free disk space, using algorithms such as LRU (last recently used). This means that the proxy expunges those objects that have not been requested for the longest time.

26.2 System Requirements

System requirements largely depend on the maximum network load that the system must bear. Therefore, examine load peaks, as during those times, load might be more than four times the day's average. When in doubt, slightly overestimate the system's requirements. Having Squid working close to the limit of its capabilities can lead to a severe loss in quality of service. The following sections point to system factors in order of significance:

1. RAM size
2. CPU speed/physical CPU cores
3. Size of the disk cache
4. Hard disks/SSDs and their architecture

26.2.1 RAM

The amount of memory (RAM) required by Squid directly correlates with the number of objects in the cache. Random access memory is much faster than a hard disk/SSD. Therefore, it is very important to have sufficient memory for the Squid process, because system performance is dramatically reduced if the swap disk is used.

Squid also stores cache object references and frequently requested objects in the main memory to speed up retrieval of this data. In addition to that, there is other data that Squid needs to keep in memory, such as a table with all the IP addresses handled, an exact domain name cache, the most frequently requested objects, access control lists, buffers, and more.

26.2.2 CPU

Squid is tuned to work best with lower processor core counts (4–8 physical cores), with each providing high performance. Technologies providing virtual cores such as hyperthreading can hurt performance.

To make the best use of multiple CPU cores, it is necessary to set up multiple worker threads writing to different caching devices. By default, multi-core support is mostly disabled.

26.2.3 Size of the Disk Cache

In a small cache, the probability of a `HIT` (finding the requested object already located there) is small, because the cache is easily filled and less requested objects are replaced by newer ones. If, for example, 1 GB is available for the cache and the users use up only 10 MB per day surfing, it would take more than one hundred days to fill the cache.

The easiest way to determine the necessary cache size is to consider the maximum transfer rate of the connection. With a 1 Mbit/s connection, the maximum transfer rate is 128 KB/s. If all this traffic ended up in the cache, in one hour it would add up to 460 MB. Assuming that all this traffic is generated in only eight working hours, it would reach 3.6 GB in one day. Because the connection is normally not used to its upper volume limit, it can be assumed that the total data volume handled by the cache is approximately 2 GB. Hence, in this example, 2 GB of disk space is required for Squid to keep one day's worth of browsing data cached.

26.2.4 Hard Disk/SSD Architecture

Speed plays an important role in the caching process, so this factor deserves special attention. For hard disks/SSDs, this parameter is described as *random seek time* or *random read performance*, measured in milliseconds. Because the data blocks that Squid reads from or writes to the hard disk/SSD tend to be small, the seek time/read performance of the hard disk/SSD is more important than its data throughput.

For use as a proxy server, hard disks with high rotation speeds or SSDs are the best choice. When using hard disks, it can be better to use multiple smaller hard disks, each with a single cache directory to avoid excessive read times.

Using a RAID system allows increasing reliability at expense of speed. However, for performance reasons, avoid (software) RAID5 and similar settings.

File system choice is usually not decisive. However, using the mount option `noatime` can improve performance—Squid provides its own time stamps and thus does not need the file system to track access times.

26.3 Basic Usage of Squid

If not already installed, install the package `squid` . `squid` is not among the packages installed by default on openSUSE® Leap.

Squid is already preconfigured in openSUSE Leap, you can start it directly after the installation. To ensure a smooth start-up, the network should be configured in a way that at least one name server and the Internet can be reached. Problems can arise if a dial-up connection is used with a dynamic DNS configuration. In this case, at least the name server should be specified, because Squid does not start if it does not detect a DNS server in `/var/run/netconfig/resolv.conf`.

26.3.1 Starting Squid

To start Squid, use:

```
tux > sudo systemctl start squid
```

If you want Squid to start when the system boots up, enable the service with **`systemctl enable squid`**.

26.3.2 Checking Whether Squid Is Working

To check whether Squid is running, choose one of the following ways:

- Using **`systemctl`**:

  ```
  tux > systemctl status squid
  ```

  The output of this command should indicate that Squid is `loaded` and `active (running)`.

- Using Squid itself:

  ```
  tux > sudo squid -k check | echo $?
  ```

  The output of this command should be `0`, but may contain additional warnings or messages.

To test the functionality of Squid on the local system, choose one of the following ways:

- To test, you can use **`squidclient`**, a command-line tool that can output the response to a Web request, similar to **`wget`** or **`curl`**.
  Unlike those tools, **`squidclient`** will automatically connect to the default proxy setup of Squid, `localhost:3128`. However, if you changed the configuration of Squid, you need to configure **`squidclient`** to use different settings using command line options. For more information, see **`squidclient --help`**.

  EXAMPLE 26.1: A REQUEST WITH **`squidclient`**

  ```
  tux > squidclient http://www.example.org
  HTTP/1.1 200 OK
  Cache-Control: max-age=604800
  Content-Type: text/html
  Date: Fri, 22 Jun 2016 12:00:00 GMT
  Expires: Fri, 29 Jun 2016 12:00:00 GMT
  Last-Modified: Fri, 09 Aug 2013 23:54:35 GMT
  Server: ECS (iad/182A)
  Vary: Accept-Encoding
  X-Cache: HIT
  x-ec-custom-error: 1
  Content-Length: 1270
  X-Cache: MISS from moon (1)
  X-Cache-Lookup: MISS from moon:3128
  Via: 1.1 moon (squid/3.5.16) (2)
  Connection: close
  ```

```
<!doctype html>
<html>
<head>
    <title>Example Domain</title>
[...]
</body>
</html>
```

The output shown in *Example 26.1, "A Request With* **squidclient***"* can be split into two parts:

1. The protocol headers of the response: the lines before the blank line.
2. The actual content of the response: the lines after the blank line.

To verify that Squid is used, refer to the selected header lines:

1. The value of the header `X-Cache` tells you that the requested document was not in the Squid cache (`MISS`) of the computer `moon`.
   The example above contains two `X-Cache` lines. You can ignore the first `X-Cache` header. It is produced by the internal caching software of the originating Web server.
2. The value of the header `Via` tells you the HTTP version, the name of the computer, and the version of Squid in use.

- Using a browser: Set up `localhost` as the proxy and `3128` as the port. You can then load a page and check the response headers in the *Network* panel of the browser's *Inspector* or *Developer Tools*. The headers should be reproduced similarly to the way shown in *Example 26.1, "A Request With* **squidclient***"*.

To allow users from the local system and other systems to access Squid and the Internet, change the entry in the configuration files `/etc/squid/squid.conf` from `http_access deny all` to `http_access allow all`. However, in doing so, consider that Squid is made completely accessible to anyone by this action. Therefore, define ACLs (access control lists) that control access to the proxy server. After modifying the configuration file, Squid must be reloaded or restarted. For more information on ACLs, see *Section 26.5.2, "Options for Access Controls"*.

If Squid quits after a short period of time even though it was started successfully, check whether there is a faulty name server entry or whether the `/var/run/netconfig/resolv.conf` file is missing. Squid logs the cause of a start-up failure in the file `/var/log/squid/cache.log`.

26.3.3 Stopping, Reloading, and Restarting Squid

To reload Squid, choose one of the following ways:

- Using **systemctl**:

```
tux > sudo systemctl reload squid
```

or

```
tux > sudo systemctl restart squid
```

- Using YaST:
  In the Squid module, click the *Save Settings and Restart Squid Now* button.

To stop Squid, choose one of the following ways:

- Using **systemctl**:

```
tux > sudo systemctl stop squid
```

- Using YaST
  In the Squid module click the *Stop Squid Now*. button.

Shutting down Squid can take a while, because Squid waits up to half a minute before dropping the connections to the clients and writing its data to the disk (see `shutdown_lifetime` option in `/etc/squid/squid.conf`),

Warning: Terminating Squid

Terminating Squid with **kill** or **killall** can damage the cache. To be able to restart Squid, damaged caches must be deleted.

26.3.4 Removing Squid

Removing Squid from the system does not remove the cache hierarchy and log files. To remove these, delete the `/var/cache/squid` directory manually.

26.3.5 Local DNS Server

Setting up a local DNS server makes sense even if it does not manage its own domain. It then simply acts as a caching-only name server and is also able to resolve DNS requests via the root name servers without requiring any special configuration (see *Section 19.4, "Starting the BIND Name Server"*). How this can be done depends on whether you chose dynamic DNS during the configuration of the Internet connection.

Dynamic DNS

Normally, with dynamic DNS, the DNS server is set by the provider during the establishment of the Internet connection and the local `/var/run/netconfig/resolv.conf` file is adjusted automatically. This behavior is controlled in the `/etc/sysconfig/network/config` file with the `NETCONFIG_DNS_POLICY` sysconfig variable. Set `NETCONFIG_DNS_POLICY` to `""` with the YaST sysconfig editor.

Then, add the local DNS server in the `/var/run/netconfig/resolv.conf` file with the IP address `127.0.0.1` for `localhost`. This way, Squid can always find the local name server when it starts.

To make the provider's name server accessible, specify it in the configuration file `/etc/named.conf` under `forwarders` along with its IP address. With dynamic DNS, this can be achieved automatically when establishing the connection by setting the sysconfig variable `NETCONFIG_DNS_POLICY` to `auto`.

Static DNS

With static DNS, no automatic DNS adjustments take place while establishing a connection, so there is no need to change any sysconfig variables. However, you must specify the local DNS server in the file `/var/run/netconfig/resolv.conf` as described in *Dynamic DNS*. Additionally, the provider's static name server must be specified manually in the `/etc/named.conf` file under `forwarders` along with its IP address.

Tip: DNS and Firewall

If you have a firewall running, make sure DNS requests can pass it.

26.4 The YaST Squid Module

The YaST Squid module contains the following tabs:

Start-Up

Specifies how Squid is started and which Firewall port is open on which interfaces.

HTTP Ports

Define all ports where Squid will listen for HTTP requests from clients.

Refresh Patterns

Defines how Squid treats objects in the cache.

Cache Settings

Defines settings in regard to cache memory, maximum and minimum object size, and more.

Cache Directory

Defines the top-level directory where Squid stores all cache swap files.

Access Control

Controls the access to the Squid server via ACL groups.

Logging and Timeout

Define paths to access, cache, and cache store log files in addition with connection timeouts and client lifetime.

Miscellaneous

Sets language and mail address of administrator.

26.5 The Squid Configuration File

All Squid proxy server settings are made in the `/etc/squid/squid.conf` file. To start Squid for the first time, no changes are necessary in this file, but external clients are initially denied access. The proxy is available for `localhost`. The default port is `3128`. The preinstalled configuration file `/etc/squid/squid.conf` provides detailed information about the options and many examples.

Many entries are commented and therefore begin with the comment character `#`. The relevant specifications can be found at the end of the line. The given values usually correlate with the default values, so removing the comment signs without changing any of the parameters usually

has no effect. If possible, leave the commented lines as they are and insert the options along with the modified values in the line below. This way, the default values may easily be recovered and compared with the changes.

Tip: Adapting the Configuration File After an Update

If you have updated from an earlier Squid version, it is recommended to edit the new `/etc/squid/squid.conf` and only apply the changes made in the previous file.

Sometimes, Squid options are added, removed, or modified. Therefore, if you try to use the old `squid.conf`, Squid might stop working properly.

26.5.1 General Configuration Options

The following is a list of a selection of configuration options for Squid. It is not exhaustive. The Squid package contains a full, lightly documented list of options in `/etc/squid/squid.conf.documented`.

`http_port` *`PORT`*

This is the port on which Squid listens for client requests. The default port is `3128`, but `8080` is also common.

`cache_peer` *`HOST_NAME TYPE PROXY_PORT ICP_PORT`*

This option allows creating a network of caches that work together. The cache peer is a computer that also hosts a network cache and stands in a relationship to your own. The type of relationship is specified as the *`TYPE`*. The type can either be `parent` or `sibling`. As the *`HOST_NAME`*, specify the name or IP address of the proxy server to use. For *`PROXY_PORT`*, specify the port number for use in a browser (usually `8080`). Set *`ICP_PORT`* to `7` or, if the ICP port of the parent is not known and its use is irrelevant to the provider, to `0`.

To make Squid behave like a Web browser instead of like a proxy server, prohibit the use of the ICP protocol. You can do so by appending the options `default` and `no-query`.

`cache_mem` *`SIZE`*

This option defines the amount of memory Squid can use for very popular replies. The default is `8 MB`. This does not specify the memory usage of Squid and may be exceeded.

`cache_dir` *`STORAGE_TYPE`* *`CACHE_DIRECTORY`* *`CACHE_SIZE`* *`LEVEL_1_DIRECTORIES`* *`LEVEL_2_DIRECTORIES`*

The option `cache_dir` defines the directory for the disk cache. In the default configuration on openSUSE Leap, Squid does not create a disk cache.

The placeholder *`STORAGE_TYPE`* can be one of the following:

- Directory-based storage types: `ufs`, `aufs` (the default), `diskd`. All three are variations of the storage format `ufs`. However, while `ufs` runs as part of the core Squid thread, `aufs` runs in a separate thread, and `diskd` uses a separate process. This means that the latter two types avoid blocking Squid because of disk I/O.
- Database-based storage systems: `rock`. This storage format relies on a single database file, in which each object takes up one or more memory units of a fixed size ("slots").

In the following, only the parameters for storage types based on `ufs` will be discussed. `rock` has somewhat different parameters.

The *`CACHE_DIRECTORY`* is the directory for the disk cache. By default, that is `/var/cache/squid`. *`CACHE_SIZE`* is the maximum size of that directory in megabytes; by default, this is set to 100 MB. Set it to between 50% and a maximum of 80% of available disk space.

The final two values, *`LEVEL_1_DIRECTORIES`* and *`LEVEL_2_DIRECTORIES`* specify how many subdirectories are created in the *`CACHE_DIRECTORY`*. By default, 16 subdirectories are created at the first level below *`CACHE_DIRECTORY`* and 256 within each of these. These values should only be increased with caution, because creating too many directories can lead to performance problems.

If you have several disks that share a cache, specify several `cache_dir` lines.

`cache_access_log` *`LOG_FILE`*,
`cache_log` *`LOG_FILE`*,
`cache_store_log` *`LOG_FILE`*

These three options specify the paths where Squid logs all its actions. Normally, nothing needs to be changed here. If Squid is burdened by heavy usage, it might make sense to distribute the cache and the log files over several disks.

`client_netmask` *`NETMASK`*

This option allows masking IP addresses of clients in the log files by applying a subnet mask. For example, to set the last digit of the IP address to `0`, specify `255.255.255.0`.

`ftp_user` *`E-MAIL`*

This option allows setting the password that Squid should use for anonymous FTP login. Specify a valid e-mail address here, because some FTP servers check these for validity.

`cache_mgr` *`E-MAIL`*

If it unexpectedly crashes, Squid sends a message to this e-mail address. The default is *webmaster*.

`logfile_rotate` *`VALUE`*

If you run **`squid`** `-k rotate`, **`squid`** can rotate log files. The files are numbered in this process and, after reaching the specified value, the oldest file is overwritten. The default value is `10` which rotates log files with the numbers 0 to 9.

However, on openSUSE Leap, rotating log files is performed automatically using `logrotate` and the configuration file `/etc/logrotate.d/squid`.

`append_domain` *`DOMAIN`*

Use *append_domain* to specify which domain to append automatically when none is given. Usually, your own domain is specified here, so specifying *www* in the browser accesses your own Web server.

`forwarded_for` *`STATE`*

If this option is set to `on`, it adds a line to the header similar to this:

```
X-Forwarded-For: 192.168.0.1
```

If you set this option to `off`, Squid removes the IP address and the system name of the client from HTTP requests.

`negative_ttl` *`TIME`*,
`negative_dns_ttl` *`TIME`*

If these options are set, Squid will cache some types of failures, such as `404` responses. It will then refuse to issue new requests, even if the resource would be available then.

By default, `negative_ttl` is set to `0`, `negative_dns_ttl` is set to `1 minutes`. This means that negative responses to Web requests are not cached by default, while negative responses to DNS requests are cached for 1 minute.

`never_direct allow` *`ACL_NAME`*

To prevent Squid from taking requests directly from the Internet, use the option `never_direct` to force connection to another proxy server. This must have previously been specified in `cache_peer`. If `all` is specified as the *`ACL_NAME`*, all requests are forwarded directly to the `parent`. This can be necessary, for example, if you are using a provider that dictates the use of its proxies or denies its firewall direct Internet access.

26.5.2 Options for Access Controls

Squid provides a detailed system for controlling the access to the proxy server. These Access Control Lists (ACL) are lists with rules that are processed sequentially. ACLs must be defined before they can be used. Some default ACLs, such as `all` and `localhost`, already exist. However, the mere definition of an ACL does not mean that it is actually applied. This only happens when there is a corresponding `http_access` rule.

The syntax for the option `acl` is as follows:

```
acl ACL_NAME TYPE DATA
```

The placeholders within this syntax stand for the following:

- The name *`ACL_NAME`* can be chosen arbitrarily.
- For *`TYPE`*, select from a variety of different options which can be found in the `ACCESS CONTROLS` section in the `/etc/squid/squid.conf` file.
- The specification for *`DATA`* depends on the individual ACL type, for example host names, IP addresses, or URLs, and can also be read from a file.

To add rules in the YaST squid module, open the module and click the *Access Control* tab. Click *Add* under the ACL Groups list and enter the name of your rule, the type, and its parameters.

For more information on types of ACL rules, see the Squid documentation at http://www.squid-cache.org/Versions/v3/3.5/cfgman/acl.html.

EXAMPLE 26.2: DEFINING ACL RULES

```
acl mysurfers srcdomain .example.com ❶
acl teachers src 192.168.1.0/255.255.255.0 ❷
acl students src 192.168.7.0-192.168.9.0/255.255.255.0 ❸
acl lunch time MTWHF 12:00-15:00 ❹
```

❶ This ACL defines `mysurfers` as all users coming from within `.example.com` (as determined by a reverse lookup for the IP).

❷ This ACL defines `teachers` as the users of computers with IP addresses starting with `192.168.1.`.

❸ This ACL defines `students` as the users of the computer with IP addresses starting with `192.168.7.`, `192.168.8.`, or `192.168.9.`.

4 This ACL defines `lunch` as a time on the days Monday through Friday between noon and 3 p.m.

http_access allow *ACL_NAME*

`http_access` defines who is allowed to use the proxy server and who can access what on the Internet. For this, ACLs must be defined. `localhost` and `all` have already been defined above for which you can deny or allow access via `deny` or `allow`. A list containing any number of `http_access` entries can be created, processed from top to bottom. Depending on which occurs first, access is allowed or denied to the respective URL. The last entry should always be `http_access deny all`. In the following example, `localhost` has free access to everything while all other hosts are denied access completely:

```
http_access allow localhost
http_access deny all
```

In another example using these rules, the group `teachers` always has access to the Internet. The group `students` only has access between Monday and Friday during lunch time:

```
http_access deny localhost
http_access allow teachers
http_access allow students lunch time
http_access deny all
```

For readability, within the configuration file `/etc/squid/squid.conf`, specify all `http_access` options as a block.

`url_rewrite_program` *PATH*

With this option, specify a URL rewriter.

`auth_param basic program` *PATH*

If users must be authenticated on the proxy server, set a corresponding program, such as `/usr/sbin/pam_auth`. When accessing **`pam_auth`** for the first time, the user sees a login window in which they need to specify a user name and a password. In addition, you need an ACL, so only clients with a valid login can use the Internet:

```
acl password proxy_auth REQUIRED

http_access allow password
http_access deny all
```

In the `acl proxy_auth` option, using `REQUIRED` means that all valid user names are accepted. `REQUIRED` can also be replaced with a list of permitted user names.

`ident_lookup_access allow ACL_NAME`

With this option, have an ident request run to find each user's identity for all clients defined by an ACL of the type `src`. Alternatively, use this for all clients, apply the predefined ACL `all` as the *`ACL_NAME`*.

All clients covered by `ident_lookup_access` must run an ident daemon. On Linux, you can use `pidentd` (package `pidentd`) as the ident daemon. For other operating systems, free software is usually available. To ensure that only clients with a successful ident lookup are permitted, define a corresponding ACL:

```
acl identhosts ident REQUIRED

http_access allow identhosts
http_access deny all
```

In the `acl identhosts ident` option, using `REQUIRED` means that all valid user names are accepted. `REQUIRED` can also be replaced with a list of permitted user names.

Using `ident` can slow down access time, because ident lookups are repeated for each request.

26.6 Configuring a Transparent Proxy

A transparent proxy intercepts and answers the requests of the Web browser, so the Web browser receives the requested pages without knowing where they are coming from. As the name indicates, the entire process is transparent to the user.

The usual way of working with proxy servers is as follows: the Web browser sends requests to a certain port of the proxy server and the proxy always provides these required objects, regardless of whether they are in its cache. However, in some cases the transparent proxy mode of Squid makes sense:

- If, for security reasons, it is recommended that all clients use a proxy server to surf the Internet.
- If all clients must use a proxy server, regardless of whether they are aware of it.
- If the proxy server in a network is moved, but the existing clients need to retain their old configuration.

PROCEDURE 26.1: SQUID AS A TRANSPARENT PROXY SERVER (COMMAND LINE)

1. In `/etc/squid/squid.conf`, on the line of the option `http_port` add the parameter `transparent`. You should then have 2 lines:

```
http_port 3128
http_port 3128 transparent
```

2. Restart Squid:

```
tux > sudo systemctl restart squid
```

3. Set up the firewall to redirect HTTP traffic to the port given in `http_proxy`. In the example above it is port 3128. Then reload the firewall configuration. This assumes that the zone `internal` is assigned to your LAN interface.

```
tux > sudo firewall-cmd --permanent --zone=internal \
    --add-forward-port=port=80:proto=tcp:toport=3128:toaddr=LAN_IP
tux > sudo firewall-cmd --permanent --zone=internal --add-port=3128/tcp
tux > sudo firewall-cmd --reload
```

   Replace *`LAN_IP`* with the IP address of your LAN interface or the interface Squid is listening on.

4. To verify that everything is working properly, check the Squid log files in `/var/log/squid/access.log`.

26.7 Using the Squid Cache Manager CGI Interface (`cachemgr.cgi`)

The Squid cache manager CGI interface (`cachemgr.cgi`) is a CGI utility for displaying statistics about the memory usage of a running Squid process. It is also a convenient way to manage the cache and view statistics without logging the server.

PROCEDURE 26.2: SETTING UP `cachemgr.cgi`

1. Make sure the Apache Web server is running on your system. Configure Apache as described in *Chapter 24, The Apache HTTP Server*. In particular, see *Section 24.5, "Enabling CGI Scripts"*. To check whether Apache is already running, use:

```
tux > sudo systemctl status apache2
```

If `inactive` is shown, you can start Apache with the openSUSE Leap default settings:

```
tux > sudo systemctl start apache2
```

2. Now enable `cachemgr.cgi` in Apache. To do so, create a configuration file for a `ScriptAlias`.
   Create the file in the directory `/etc/apache2/conf.d` and name it `cachemgr.conf`. In it, add the following:

   ```
   ScriptAlias /squid/cgi-bin/ /usr/lib64/squid/

   <Directory "/usr/lib64/squid/">
   Options +ExecCGI
   AddHandler cgi-script .cgi
   Require host HOST_NAME
   </Directory>
   ```

   Replace *`HOST_NAME`* with the host name of the computer you want to access `cachemgr.cgi` from. This allows only your computer to access `cachemgr.cgi`. To allow access from anywhere, use `Require all granted` instead.

3. - If Squid and your Apache Web server run on the same computer, there should be no changes that need to be made to `/etc/squid/squid.conf`. However, verify that `/etc/squid/squid.conf` contains the following lines:

     ```
     http_access allow manager localhost
     http_access deny manager
     ```

     These lines allow you to access the manager interface from your own computer (`localhost`) but not from elsewhere.

   - If Squid and your Apache Web server run on different computers, you need to add extra rules to allow access from the CGI script to Squid. Define an ACL for your server (replace *`WEB_SERVER_IP`* with the IP address of your Web server):

     ```
     acl webserver src WEB_SERVER_IP/255.255.255.255
     ```

     Make sure the following rules are in the configuration file. Compared to the default configuration, only the rule in the middle is new. However, the sequence is important.

     ```
     http_access allow manager localhost
     ```

```
http_access allow manager webserver
http_access deny manager
```

4. *(Optional)* Optionally, you can configure one or more passwords for `cachemgr.cgi`. This also allows access to more actions such as closing the cache remotely or viewing more information about the cache. For this, configure the options `cache_mgr` and `cachemgr_passwd` with one or more password for the manager and a list of allowed actions.

   For example, to explicitly enable viewing the index page, the menu, 60-minute average of counters without authentication, to enable toggling offline mode using the password `secretpassword`, and to completely disable everything else, use the following configuration:

   ```
   cache_mgr user
   cachemgr_passwd none index menu 60min
   cachemgr_passwd secretpassword offline_toggle
   cachemgr_passwd disable all
   ```

   `cache_mgr` defines a user name. `cache_mgr` defines which actions are allowed using which password.

   The keywords `none` and `disable` are special: `none` removes the need for a password, `disable` disables functionality outright.

   The full list of actions can be best seen after logging in to `cachemgr.cgi`. To find out how the operation needs to be referenced in the configuration file, see the string after `&operation=` in the URL of the action page. `all` is a special keyword meaning all actions.

5. Reload Squid and Apache after the configuration file changes:

   ```
   tux > sudo systemctl reload squid
   ```

6. To view the statistics, go to the `cachemgr.cgi` page that you set up before. For example, it could be `http://webserver.example.org/squid/cgi-bin/cachemgr.cgi`.

   Choose the right server, and, if set, specify user name and password. Then click *Continue* and browse through the different statistics.

26.8 Cache Report Generation with Calamaris

Calamaris is a Perl script used to generate reports of cache activity in ASCII or HTML format. It works with native Squid access log files. The Calamaris home page is located at http://cord.de/calamaris-english. This tool does not belong to the openSUSE Leap default installation scope —to use it, install the `calamaris` package.

Log in as `root`, then enter:

```
root # cat access1.log [access2.log access3.log] | calamaris OPTIONS > reportfile
```

When using more than one log file, make sure they are chronologically ordered, with older files listed first. This can be achieved by either listing the files one after the other as in the example above, or by using `access{1..3}.log`.

`calamaris` takes the following options:

`-a`
: output all available reports

`-w`
: output as HTML report

`-l`
: include a message or logo in report header

More information about the various options can be found in the program's manual page with **`man`** `calamaris`.

A typical example is:

```
root # cat access.log.{10..1} access.log | calamaris -a -w \
> /usr/local/httpd/htdocs/Squid/squidreport.html
```

This puts the report in the directory of the Web server. Apache is required to view the reports.

26.9 For More Information

Visit the home page of Squid at http://www.squid-cache.org/. Here, find the “Squid User Guide” and a very extensive collection of FAQs on Squid.

In addition, mailing lists are available for Squid at http://www.squid-cache.org/Support/mailing-lists.html.

IV Mobile Computers

27 Mobile Computing with Linux

Mobile computing is mostly associated with laptops, PDAs and cellular phones (and the data exchange between them). Mobile hardware components, such as external hard disks, flash disks, or digital cameras, can be connected to laptops or desktop systems. A number of software components are involved in mobile computing scenarios and some applications are tailor-made for mobile use.

27.1 Laptops

The hardware of laptops differs from that of a normal desktop system. This is because criteria like exchangeability, space requirements and power consumption must be taken into account. The manufacturers of mobile hardware have developed standard interfaces like Mini PCI and Mini PCIe that can be used to extend the hardware of laptops. The standards cover memory cards, network interface cards, and external hard disks.

27.1.1 Power Conservation

The inclusion of energy-optimized system components during laptop manufacturing contributes to their suitability for use without access to the electrical power grid. Their contribution to conservation of power is at least as important as that of the operating system. openSUSE® Leap supports various methods that control the power consumption of a laptop and have varying effects on the operating time under battery power. The following list is in descending order of contribution to power conservation:

- Throttling the CPU speed.
- Switching off the display illumination during pauses.
- Manually adjusting the display illumination.
- Disconnecting unused, hotplug-enabled accessories (USB CD-ROM, external mouse, Wi-Fi, etc.).
- Spinning down the hard disk when idling.

Detailed background information about power management in openSUSE Leap is provided in *Chapter 29, Power Management*.

27.1.2 Integration in Changing Operating Environments

Your system needs to adapt to changing operating environments when used for mobile computing. Many services depend on the environment and the underlying clients must be reconfigured. openSUSE Leap handles this task for you.

FIGURE 27.1: INTEGRATING A MOBILE COMPUTER IN AN EXISTING ENVIRONMENT

The services affected in the case of a laptop commuting back and forth between a small home network and an office network are:

Network

This includes IP address assignment, name resolution, Internet connectivity and connectivity to other networks.

Printing

A current database of available printers and an available print server must be present, depending on the network.

E-Mail and Proxies

As with printing, the list of the corresponding servers must be current.

X (Graphical Environment)

If your laptop is temporarily connected to a projector or an external monitor, different display configurations must be available.

openSUSE Leap offers several ways of integrating laptops into existing operating environments:

NetworkManager

NetworkManager is especially tailored for mobile networking on laptops. It provides a means to easily and automatically switch between network environments or different types of networks such as mobile broadband (such as GPRS, EDGE, or 3G), wireless LAN, and Ethernet. NetworkManager supports WEP and WPA-PSK encryption in wireless LANs. It also supports dial-up connections. The GNOME desktop includes a front-end for NetworkManager. For more information, see *Section 28.3, "Configuring Network Connections"*.

TABLE 27.1: USE CASES FOR NETWORKMANAGER

My computer…	Use NetworkManager
is a laptop	Yes
is sometimes attached to different networks	Yes
provides network services (such as DNS or DHCP)	No
only uses a static IP address	No

Use the YaST tools to configure networking whenever NetworkManager should not handle network configuration.

Tip: DNS Configuration and Various Types of Network Connections

If you travel frequently with your laptop and change different types of network connections, NetworkManager works fine when all DNS addresses are assigned correctly assigned with DHCP. If some connections use static DNS address(es), add it to the `NETCONFIG_DNS_STATIC_SERVERS` option in `/etc/sysconfig/network/config`.

SLP

The service location protocol (SLP) simplifies the connection of a laptop to an existing network. Without SLP, the administrator of a laptop usually requires detailed knowledge of the services available in a network. SLP broadcasts the availability of a certain type of service to all clients in a local network. Applications that support SLP can process the information dispatched by SLP and be configured automatically. SLP can also be used to install a system, minimizing the effort of searching for a suitable installation source. Find detailed information about SLP in *Chapter 17, SLP*.

27.1.3 Software Options

There are various task areas in mobile use that are covered by dedicated software: system monitoring (especially the battery charge), data synchronization, and wireless communication with peripherals and the Internet. The following sections cover the most important applications that openSUSE Leap provides for each task.

27.1.3.1 System Monitoring

Two system monitoring tools are provided by openSUSE Leap:

Power Management

Power Management is an application that lets you adjust the energy saving related behavior of the GNOME desktop. You can typically access it via *Computer* › *Control Center* › *System* › *Power Management*.

System Monitor

The *System Monitor* gathers measurable system parameters into one monitoring environment. It presents the output information in three tabs by default. *Processes* gives detailed information about currently running processes, such as CPU load, memory usage, or process ID number and priority. The presentation and filtering of the collected data can be customized—to add a new type of process information, left-click the process table header and choose which column to hide or add to the view. It is also possible to monitor different system parameters in various data pages or collect the data of various machines in parallel over the network. The *Resources* tab shows graphs of CPU, memory and network history and the *File System* tab lists all partitions and their usage.

27.1.3.2 Synchronizing Data

When switching between working on a mobile machine disconnected from the network and working at a networked workstation in an office, it is necessary to keep processed data synchronized across all instances. This could include e-mail folders, directories and individual files that need to be present for work on the road and at the office. The solution in both cases is as follows:

Synchronizing E-Mail

Use an IMAP account for storing your e-mails in the office network. Then access the e-mails from the workstation using any disconnected IMAP-enabled e-mail client, like Mozilla Thunderbird or Evolution as described in *Book "*GNOME User Guide*"*. The e-mail client must be configured so that the same folder is always accessed for `Sent messages`. This ensures that all messages are available along with their status information after the synchronization process has completed. Use an SMTP server implemented in the mail client for sending messages instead of the system-wide MTA postfix or sendmail to receive reliable feedback about unsent mail.

Synchronizing Files and Directories

There are several utilities suitable for synchronizing data between a laptop and a workstation. One of the most widely used is a command-line tool called **`rsync`**. For more information, see its manual page (**`man 1 rsync`**).

27.1.3.3 Wireless Communication: Wi-Fi

With the largest range of these wireless technologies, Wi-Fi is the only one suitable for the operation of large and sometimes even spatially separate networks. Single machines can connect with each other to form an independent wireless network or access the Internet. Devices called *access points* act as base stations for Wi-Fi-enabled devices and act as intermediaries for access to the Internet. A mobile user can switch among access points depending on location and which access point is offering the best connection. Like in cellular telephony, a large network is available to Wi-Fi users without binding them to a specific location for accessing it.

Wi-Fi cards communicate using the 802.11 standard, prepared by the IEEE organization. Originally, this standard provided for a maximum transmission rate of 2 Mbit/s. Meanwhile, several supplements have been added to increase the data rate. These supplements define details such as the modulation, transmission output, and transmission rates (see *Table 27.2, "Overview of Various Wi-Fi Standards"*). Additionally, many companies implement hardware with proprietary or draft features.

TABLE 27.2: OVERVIEW OF VARIOUS WI-FI STANDARDS

Name (802.11)	Frequency (GHz)	Maximum Transmission Rate (Mbit/s)	Note
a	5	54	Less interference-prone
b	2.4	11	Less common
g	2.4	54	Widespread, backward-compatible with 11b
n	2.4 and/or 5	300	Common
ac	5	up to ~865	Expected to be common in 2015

Name (802.11)	Frequency (GHz)	Maximum Transmission Rate (Mbit/s)	Note
ad	60	up to appr. 7000	Released 2012, currently less common; not supported in openSUSE Leap

802.11 Legacy cards are not supported by openSUSE® Leap. Most cards using 802.11 a/b/g/n are supported. New cards usually comply with the 802.11n standard, but cards using 802.11g are still available.

27.1.3.3.1 Operating Modes

In wireless networking, various techniques and configurations are used to ensure fast, high-quality, and secure connections. Usually your Wi-Fi card operates in *managed mode*. However, different operating types need different setups. Wireless networks can be classified into four network modes:

Managed Mode (Infrastructure Mode), via Access Point (default mode)

Managed networks have a managing element: the access point. In this mode (also called infrastructure or default mode), all connections of the Wi-Fi stations in the network run through the access point, which may also serve as a connection to an Ethernet. To make sure only authorized stations can connect, various authentication mechanisms (WPA, etc.) are used. This is also the main mode that consumes the least amount of energy.

Ad-hoc Mode (Peer-to-Peer Network)

Ad-hoc networks do not have an access point. The stations communicate directly with each other, therefore an ad-hoc network is usually slower than a managed network. However, the transmission range and number of participating stations are greatly limited in ad-hoc networks. They also do not support WPA authentication. Additionally, not all cards support ad-hoc mode reliably.

Master Mode

In master mode, your Wi-Fi card is used as the access point, assuming your card supports this mode. Find out the details of your Wi-Fi card at http://linux-wless.passys.nl.

Mesh Mode

Wireless mesh networks are organized in a *mesh topology*. A wireless mesh network's connection is spread among all wireless mesh *nodes*. Each node belonging to this network is connected to other nodes to share the connection, possibly over a large area.

27.1.3.3.2 Authentication

Because a wireless network is much easier to intercept and compromise than a wired network, the various standards include authentication and encryption methods.

Old Wi-Fi cards support only WEP (Wired Equivalent Privacy). However, because WEP has proven to be insecure, the Wi-Fi industry has defined an extension called WPA, which is supposed to eliminate the weaknesses of WEP. WPA, sometimes synonymous with WPA2, should be the default authentication method.

Usually the user cannot choose the authentication method. For example, when a card operates in managed mode the authentication is set by the access point. NetworkManager shows the authentication method.

27.1.3.3.3 Encryption

There are various encryption methods to ensure that no unauthorized person can read the data packets that are exchanged in a wireless network or gain access to the network:

WEP (defined in IEEE 802.11)

This standard uses the RC4 encryption algorithm, originally with a key length of 40 bits, later also with 104 bits. Often, the length is declared as 64 bits or 128 bits, depending on whether the 24 bits of the initialization vector are included. However, this standard has some weaknesses. Attacks against the keys generated by this system may be successful. Nevertheless, it is better to use WEP than not to encrypt the network.

Some vendors have implemented the non-standard “Dynamic WEP”. It works exactly as WEP and shares the same weaknesses, except that the key is periodically changed by a key management service.

TKIP (defined in WPA/IEEE 802.11i)

This key management protocol defined in the WPA standard uses the same encryption algorithm as WEP, but eliminates its weakness. Because a new key is generated for every data packet, attacks against these keys are fruitless. TKIP is used together with WPA-PSK.

CCMP (defined in IEEE 802.11i)

CCMP describes the key management. Usually, it is used in connection with WPA-EAP, but it can also be used with WPA-PSK. The encryption takes place according to AES and is stronger than the RC4 encryption of the WEP standard.

27.1.3.4 Wireless Communication: Bluetooth

Bluetooth has the broadest application spectrum of all wireless technologies. It can be used for communication between computers (laptops) and PDAs or cellular phones, as can IrDA. It can also be used to connect various computers within range. Bluetooth is also used to connect wireless system components, like a keyboard or a mouse. The range of this technology is, however, not sufficient to connect remote systems to a network. Wi-Fi is the technology of choice for communicating through physical obstacles like walls.

27.1.3.5 Wireless Communication: IrDA

IrDA is the wireless technology with the shortest range. Both communication parties must be within viewing distance of each other. Obstacles like walls cannot be overcome. One possible application of IrDA is the transmission of a file from a laptop to a cellular phone. The short path from the laptop to the cellular phone is then covered using IrDA. Long-range transmission of the file to the recipient is handled by the mobile network. Another application of IrDA is the wireless transmission of printing jobs in the office.

27.1.4 Data Security

Ideally, you protect data on your laptop against unauthorized access in multiple ways. Possible security measures can be taken in the following areas:

Protection against Theft

Always physically secure your system against theft whenever possible. Various securing tools (like chains) are available in retail stores.

Strong Authentication

Use biometric authentication in addition to standard authentication via login and password. openSUSE Leap supports fingerprint authentication.

Securing Data on the System

Important data should not only be encrypted during transmission, but also on the hard disk. This ensures its safety in case of theft. The creation of an encrypted partition with openSUSE Leap is described in *Book "Security 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 when they are connected with the system over the corresponding interface. The file manager of GNOME offers flexible handling of mobile hardware items. To unmount any of these media safely, use the *Unmount Volume* (GNOME) feature of the file manager. For more details refer to *Book "GNOME User Guide"*.

External Hard Disks (USB and FireWire)

When an external hard disk is correctly recognized by the system, its icon appears in the file manager. Clicking the icon displays the contents of the drive. It is possible to create directories and files here and edit or delete them. To rename a hard disk, select the corresponding menu item from the right-click contextual menu. This name change is limited to display in the file manager. The descriptor by which the device is mounted in `/media` remains unaffected.

USB Flash 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 17 "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 on the move. NetworkManager can automatically connect to known wireless networks or manage several network connections in parallel—the fastest connection is then used as default. Furthermore, you can manually switch between available networks and manage your network connection using an applet in the system tray.

Instead of only one connection being active, multiple connections may be active at once. This enables you to unplug your laptop from an Ethernet and remain connected via a wireless connection.

28.1 Use Cases for NetworkManager

NetworkManager provides a sophisticated and intuitive user interface, which enables users to easily switch their network environment. However, NetworkManager is not a suitable solution in the following cases:

- Your computer provides network services for other computers in your network, for example, it is a DHCP or DNS server.
- Your computer is a Xen server or your system is a virtual system inside Xen.

28.2 Enabling or Disabling NetworkManager

On laptop computers, NetworkManager is enabled by default. However, it can be at any time enabled or disabled in the YaST Network Settings module.

1. Run YaST and go to *System* › *Network Settings*.
2. The *Network Settings* dialog opens. Go to the *Global Options* tab.
3. To configure and manage your network connections with NetworkManager:
    a. In the *Network Setup Method* field, select *User Controlled with NetworkManager*.

b. Click *OK* and close YaST.

c. Configure your network connections with NetworkManager as described in *Section 28.3, "Configuring Network Connections"*.

4. To deactivate NetworkManager and control the network with your own configuration:

a. In the *Network Setup Method* field, choose *Controlled by wicked*.

b. Click *OK*.

c. Set up your network card with YaST using automatic configuration via DHCP or a static IP address.
Find a detailed description of the network configuration with YaST in *Section 13.4, "Configuring a Network Connection with YaST"*.

28.3 Configuring Network Connections

After 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.

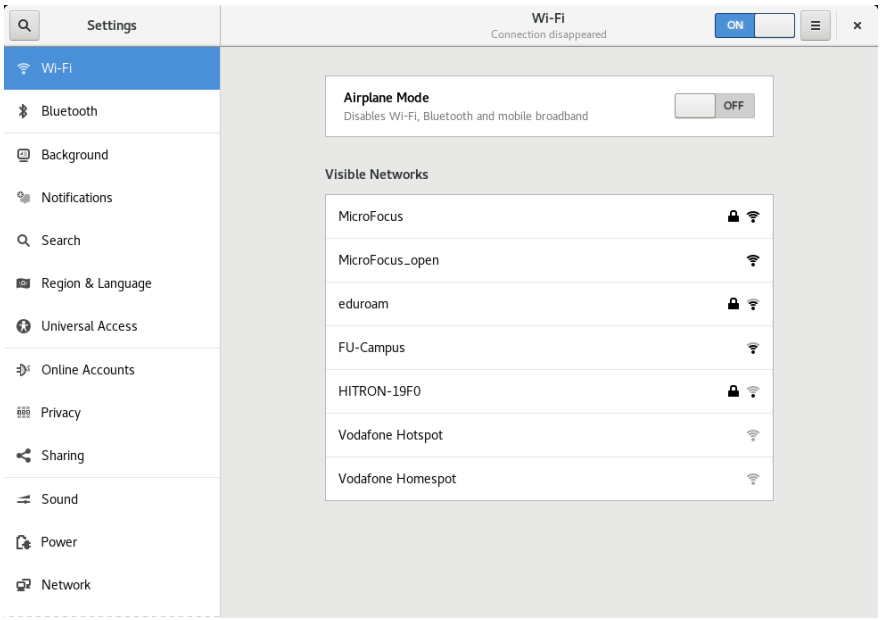


FIGURE 28.1: GNOME NETWORK CONNECTIONS DIALOG

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 a system connection go to the *Identity* tab and set the check box *Make available to other users*. For more information about user and system connections, see *Section 28.4.1, “User and System Connections”*.

28.3.1 Managing Wired Network Connections

If your computer is connected to a wired network, use the NetworkManager applet to manage the connection.

1. Open the Status Menu and click *Wired* to change the connection details or to switch it off.
2. To change the settings click *Wired Settings* and then click the gear icon.
3. To switch off all network connections, activate the *Airplane Mode* setting.

28.3.2 Managing Wireless Network Connections

Visible wireless networks are listed in the GNOME NetworkManager applet menu under *Wireless Networks*. The signal strength of each network is also shown in the menu. Encrypted wireless networks are marked with a shield icon.

PROCEDURE 28.2: CONNECTING TO A VISIBLE WIRELESS NETWORK

1. To connect to a visible wireless network, open the Status Menu and click *Wi-Fi*.
2. Click *Turn On* to enable it.
3. Click *Select Network*, select your Wi-Fi Network and click *Connect*.
4. If the network is encrypted, a configuration dialog opens. It shows the type of encryption the network uses and text boxes for entering the login credentials.

PROCEDURE 28.3: CONNECTING TO AN INVISIBLE WIRELESS NETWORK

1. To connect to a network that does not broadcast its service set identifier (SSID or ESSID) and therefore cannot be detected automatically, open the Status Menu and click *Wi-Fi*.
2. Click *Wi-Fi Settings* to open the detailed settings menu.
3. Make sure your Wi-Fi is enabled and click *Connect to Hidden Network*.
4. In the dialog that opens, enter the SSID or ESSID in *Network Name* and set encryption parameters if necessary.

A wireless network that has been chosen explicitly will remain connected as long as possible. If a network cable is plugged in during that time, any connections that have been set to *Stay connected when possible* will be connected, while the wireless connection remains up.

28.3.3 Configuring Your Wi-Fi/Bluetooth Card as an Access Point

If your Wi-Fi/Bluetooth card supports access point mode, you can use NetworkManager for the configuration.

1. Open the Status Menu and click *Wi-Fi*.
2. Click *Wi-Fi Settings* to open the detailed settings menu.
3. Click *Use as Hotspot* and follow the instructions.
4. Use the credentials shown in the resulting dialog to connect to the hotspot from a remote machine.

28.3.4 NetworkManager and VPN

NetworkManager supports several Virtual Private Network (VPN) technologies. For each technology, openSUSE Leap comes with a base package providing the generic support for NetworkManager. In addition to that, you also need to install the respective desktop-specific package for your applet.

OpenVPN

To use this VPN technology, install:

- `NetworkManager-openvpn`
- `NetworkManager-openvpn-gnome`

OpenConnect

To use this VPN technology, install:

- `NetworkManager-openconnect`
- `NetworkManager-openconnect-gnome`

PPTP (Point-to-Point Tunneling Protocol)

To use this VPN technology, install:

- `NetworkManager-pptp`
- `NetworkManager-pptp-gnome`

The following procedure describes how to set up your computer as an OpenVPN client using NetworkManager. Setting up other types of VPNs works analogously.

Before you begin, make sure that the package `NetworkManager-openvpn-gnome` is installed and all dependencies have been resolved.

PROCEDURE 28.4: SETTING UP OPENVPN WITH NETWORKMANAGER

1. Open the application *Settings* by clicking the status icons at the right end of the panel and clicking the *wrench and screwdriver* icon. In the window *All Settings*, choose *Network*.
2. Click the + icon.
3. Select *VPN* and then *OpenVPN*.
4. Choose the *Authentication* type. Depending on the setup of your OpenVPN server, choose *Certificates (TLS)* or *Password with Certificates (TLS)*.
5. Insert the necessary values into the respective text boxes. For our example configuration, these are:

Gateway	The remote endpoint of the VPN server
User name	The user (only available when you have selected *Password with Certificates (TLS)*)
Password	The password for the user (only available when you have selected *Password with Certificates (TLS)*)
User Certificate	`/etc/openvpn/client1.crt`
CA Certificate	`/etc/openvpn/ca.crt`
Private Key	`/etc/openvpn/client1.key`

6. Finish the configuration with *Add*.
7. To enable the connection, in the *Network* panel of the *Settings* application click the switch button. Alternatively, click the status icons at the right end of the panel, click the name of your VPN and then *Connect*.

28.4 NetworkManager and Security

NetworkManager distinguishes two types of wireless connections: trusted and untrusted. A trusted connection is any network that you explicitly selected in the past. All others are untrusted. Trusted connections are identified by the name and MAC address of the access point. Using the MAC address ensures that you cannot use a different access point with the name of your trusted connection.

NetworkManager periodically scans for available wireless networks. If multiple trusted networks are found, the most recently used is automatically selected. NetworkManager waits for your selection in case if all networks are untrusted.

If the encryption setting changes but the name and MAC address remain the same, NetworkManager attempts to connect, but first you are asked to confirm the new encryption settings and provide any updates, such as a new key.

If you switch from using a wireless connection to offline mode, NetworkManager blanks the SSID or ESSID. This ensures that the card is disconnected.

28.4.1 User and System Connections

NetworkManager knows two types of connections: `user` and `system` connections. User connections 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 connections 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 on how to configure user or system connections with NetworkManager, refer to *Section 28.3, "Configuring Network Connections"*.

28.4.2 Storing Passwords and Credentials

If you do not want to re-enter your credentials each time you want to connect to an encrypted network, you can use the GNOME Keyring Manager to store your credentials encrypted on the disk, secured by a master password.

NetworkManager can also retrieve its certificates for secure connections (for example, encrypted wired, wireless or VPN connections) from the certificate store. For more information, refer to *Book "Security 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 this in GNOME, first look up the MAC address of your device (use the *Connection Information* available from the applet, or use the output of command line tools like **`nm-tool`** or **`wicked show all`**). Then start the dialog for configuring network connections and choose the connection you want to modify. On the *Wired* or *Wireless* tab, enter the *MAC Address* of the device and confirm your changes.

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 with other computers?*

The primary device (the device which is connected to the Internet) does not need any special configuration. However, you need to configure the device that is connected to the local hub or machine as follows:

1. Start the dialog for configuring network connections as described in *Section 28.3, "Configuring Network Connections"*. Choose the connection you want to modify and click *Edit*. Switch to the *IPv4 Settings* tab and from the *Method* drop-down box, activate *Shared to other computers*. That will enable IP traffic forwarding and run a DHCP server on the device. Confirm your changes in NetworkManager.

2. As the DCHP 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 applets starts automatically if the network is set up for NetworkManager control. If the applet does not start, check if NetworkManager is enabled in YaST as described in *Section 28.2, "Enabling or Disabling NetworkManager"*. Then make sure that the NetworkManager-gnome package is also installed.

If the desktop applet is installed but is not running for some reason, start it manually with the command `nm-applet`.

NetworkManager Applet Does Not Include the VPN Option

Support for NetworkManager, applets, and VPN for NetworkManager is distributed in separate packages. If your NetworkManager applet does not include the VPN option, check if the packages with NetworkManager support for your VPN technology are installed. For more information, see *Section 28.3.4, "NetworkManager and VPN"*.

No Network Connection Available

If you have configured your network connection correctly and all other components for the network connection (router, etc.) are also up and running, it sometimes helps to restart the network interfaces on your computer. To do so, log in to a command line as `root` and run `systemctl restart wickeds`.

28.7 For More Information

More information about NetworkManager can be found on the following Web sites and directories:

NetworkManager Project Page

http://projects.gnome.org/NetworkManager/

Package Documentation

Also check out the information in the following directories for the latest information about NetworkManager and the GNOME applet:

- `/usr/share/doc/packages/NetworkManager/`,
- `/usr/share/doc/packages/NetworkManager-gnome/`.

29 Power Management

Power management is especially important on laptop computers, but is also useful on other systems. ACPI (Advanced Configuration and Power Interface) is available on all modern computers (laptops, desktops, and servers). Power management technologies require suitable hardware and BIOS routines. Most laptops and many modern desktops and servers meet these requirements. It is also possible to control CPU frequency scaling to save power or decrease noise.

29.1 Power Saving Functions

Power saving functions are not only significant for the mobile use of laptops, but also for desktop systems. The main functions and their use in ACPI are:

Standby

Not supported.

Suspend (to memory)

This mode writes the entire system state to the RAM. Subsequently, the entire system except the RAM is put to sleep. In this state, the computer consumes very little power. The advantage of this state is the possibility of resuming work at the same point within a few seconds without having to boot and restart applications. This function corresponds to the ACPI state `S3`.

Hibernation (suspend to disk)

In this operating mode, the entire system state is written to the hard disk and the system is powered off. There must be a swap partition at least as big as the RAM to write all the active data. Reactivation from this state takes about 30 to 90 seconds. The state prior to the suspend is restored. Some manufacturers offer useful hybrid variants of this mode, such as RediSafe in IBM Thinkpads. The corresponding ACPI state is `S4`. In Linux, suspend to disk is performed by kernel routines that are independent from ACPI.

Note: Changed UUID for Swap Partitions When Formatting via `mkswap`

Do not reformat existing swap partitions with `mkswap` if possible. Reformatting with `mkswap` will change the UUID value of the swap partition. Either reformat via YaST (which will update `/etc/fstab`) or adjust `/etc/fstab` manually.

Battery Monitor

ACPI checks the battery charge status and provides information about it. Additionally, it coordinates actions to perform when a critical charge status is reached.

Automatic Power-Off

Following a shutdown, the computer is powered off. This is especially important when an automatic shutdown is performed shortly before the battery is empty.

Processor Speed Control

In connection with the CPU, energy can be saved in three different ways: frequency and voltage scaling (also known as PowerNow! or Speedstep), throttling and putting the processor to sleep (C-states). Depending on the operating mode of the computer, these methods can also be combined.

29.2 Advanced Configuration and Power Interface (ACPI)

ACPI was designed to enable the operating system to set up and control the individual hardware components. ACPI supersedes both Power Management Plug and Play (PnP) and Advanced Power Management (APM). It delivers information about the battery, AC adapter, temperature, fan and system events, like “close lid” or “battery low.”

The BIOS provides tables containing information about the individual components and hardware access methods. The operating system uses this information for tasks like assigning interrupts or activating and deactivating components. Because the operating system executes commands stored into the BIOS, the functionality depends on the BIOS implementation. The tables ACPI can detect and load are reported in journald. See *Chapter 11, `journalctl`: Query the `systemd` Journal* for more information on viewing the journal log messages. See *Section 29.2.2, “Troubleshooting”* for more information about troubleshooting ACPI problems.

29.2.1 Controlling the CPU Performance

The CPU can save energy in three ways:

- Frequency and Voltage Scaling
- Throttling the Clock Frequency (T-states)
- Putting the Processor to Sleep (C-states)

Depending on the operating mode of the computer, these methods can be combined. Saving energy also means that the system heats up less and the fans are activated less frequently.

Frequency scaling and throttling are only relevant if the processor is busy, because the most economic C-state is applied anyway when the processor is idle. If the CPU is busy, frequency scaling is the recommended power saving method. Often the processor only works with a partial load. In this case, it can be run with a lower frequency. Usually, dynamic frequency scaling controlled by the kernel on-demand governor is the best approach.

Throttling should be used as the last resort, for example, to extend the battery operation time despite a high system load. However, some systems do not run smoothly when they are throttled too much. Moreover, CPU throttling does not make sense if the CPU has little to do.

For in-depth information, refer to *Book "System Analysis and Tuning Guide", Chapter 11 "Power Management"*.

29.2.2 Troubleshooting

There are two different types of problems. On one hand, the ACPI code of the kernel may contain bugs that were not detected in time. In this case, a solution will be made available for download. More often, the problems are caused by the BIOS. Sometimes, deviations from the ACPI specification are purposely integrated in the BIOS to circumvent errors in the ACPI implementation of other widespread operating systems. Hardware components that have serious errors in the ACPI implementation are recorded in a blacklist that prevents the Linux kernel from using ACPI for these components.

The first thing to do when problems are encountered is to update the BIOS. If the computer does not boot, one of the following boot parameters may be helpful:

pci=noacpi

Do not use ACPI for configuring the PCI devices.

acpi=ht

Only perform a simple resource configuration. Do not use ACPI for other purposes.

acpi=off

Disable ACPI.

Warning: Problems Booting without ACPI

Some newer machines (especially SMP systems and AMD64 systems) need ACPI for configuring the hardware correctly. On these machines, disabling ACPI can cause problems.

Sometimes, the machine is confused by hardware that is attached over USB or FireWire. If a machine refuses to boot, unplug all unneeded hardware and try again.

Monitor the boot messages of the system with the command `dmesg -T | grep -2i acpi` (or all messages, because the problem may not be caused by ACPI) after booting. If an error occurs while parsing an ACPI table, the most important table—the DSDT (*Differentiated System Description Table*)—can be replaced with an improved version. In this case, the faulty DSDT of the BIOS is ignored. The procedure is described in *Section 29.4, "Troubleshooting"*.

In the kernel configuration, there is a switch for activating ACPI debug messages. If a kernel with ACPI debugging is compiled and installed, detailed information is issued.

If you experience BIOS or hardware problems, it is always advisable to contact the manufacturers. Especially if they do not always provide assistance for Linux, they should be confronted with the problems. Manufacturers will only take the issue seriously if they realize that an adequate number of their customers use Linux.

29.2.2.1 For More Information

- http://tldp.org/HOWTO/ACPI-HOWTO/ (detailed ACPI HOWTO, contains DSDT patches)
- http://www.acpi.info (Advanced Configuration & Power Interface Specification)
- http://acpi.sourceforge.net/dsdt/index.php (DSDT patches by Bruno Ducrot)

29.3 Rest for the Hard Disk

In Linux, the hard disk can be put to sleep entirely if it is not needed or it can be run in a more economic or quieter mode. On modern laptops, you do not need to switch off the hard disks manually, because they automatically enter an economic operating mode whenever they are not needed. However, if you want to maximize power savings, test some of the following methods, using the **hdparm** command.

It can be used to modify various hard disk settings. The option `-y` instantly switches the hard disk to the standby mode. `-Y` puts it to sleep. **hdparm** `-S` *X* causes the hard disk to be spun down after a certain period of inactivity. Replace *X* as follows: `0` disables this mechanism, causing the hard disk to run continuously. Values from `1` to `240` are multiplied by 5 seconds. Values from `241` to `251` correspond to 1 to 11 times 30 minutes.

Internal power saving options of the hard disk can be controlled with the option `-B`. Select a value from `0` to `255` for maximum saving to maximum throughput. The result depends on the hard disk used and is difficult to assess. To make a hard disk quieter, use the option `-M`. Select a value from `128` to `254` for quiet to fast.

Often, it is not so easy to put the hard disk to sleep. In Linux, numerous processes write to the hard disk, waking it up repeatedly. Therefore, it is important to understand how Linux handles data that needs to be written to the hard disk. First, all data is buffered in the RAM. This buffer is monitored by the `pdflush` daemon. When the data reaches a certain age limit or when the buffer is filled to a certain degree, the buffer content is flushed to the hard disk. The buffer size is dynamic and depends on the size of the memory and the system load. By default, pdflush is set to short intervals to achieve maximum data integrity. It checks the buffer every 5 seconds and writes the data to the hard disk. The following variables are interesting:

`/proc/sys/vm/dirty_writeback_centisecs`

Contains the delay until a pdflush thread wakes up (in hundredths of a second).

`/proc/sys/vm/dirty_expire_centisecs`

Defines after which timeframe a dirty page should be written at latest. Default is `3000`, which means 30 seconds.

`/proc/sys/vm/dirty_background_ratio`

Maximum percentage of dirty pages until pdflush begins to write them. Default is `5` %.

`/proc/sys/vm/dirty_ratio`

When the dirty pages exceed this percentage of the total memory, processes are forced to write dirty buffers during their time slice instead of continuing to write.

Warning: Impairment of the Data Integrity

Changes to the `pdflush` daemon settings endanger the data integrity.

Apart from these processes, journaling file systems, like `Btrfs`, `Ext3`, `Ext4` and others write their metadata independently from `pdflush`, which also prevents the hard disk from spinning down. To avoid this, a special kernel extension has been developed for mobile devices. To use the extension, install the `laptop-mode-tools` package and see `/usr/src/linux/Documentation/laptops/laptop-mode.txt` for details.

Another important factor is the way active programs behave. For example, good editors regularly write hidden backups of the currently modified file to the hard disk, causing the disk to wake up. Features like this can be disabled at the expense of data integrity.

In this connection, the mail daemon postfix uses the variable `POSTFIX_LAPTOP`. If this variable is set to `yes`, postfix accesses the hard disk far less frequently.

In openSUSE Leap these technologies are controlled by `laptop-mode-tools`.

29.4 Troubleshooting

All error messages and alerts are logged in the system journal, which can be queried with the command **`journalctl`** (see *Chapter 11, `journalctl`: Query the `systemd` Journal* for more information). The following sections cover the most common problems.

29.4.1 CPU Frequency Does Not Work

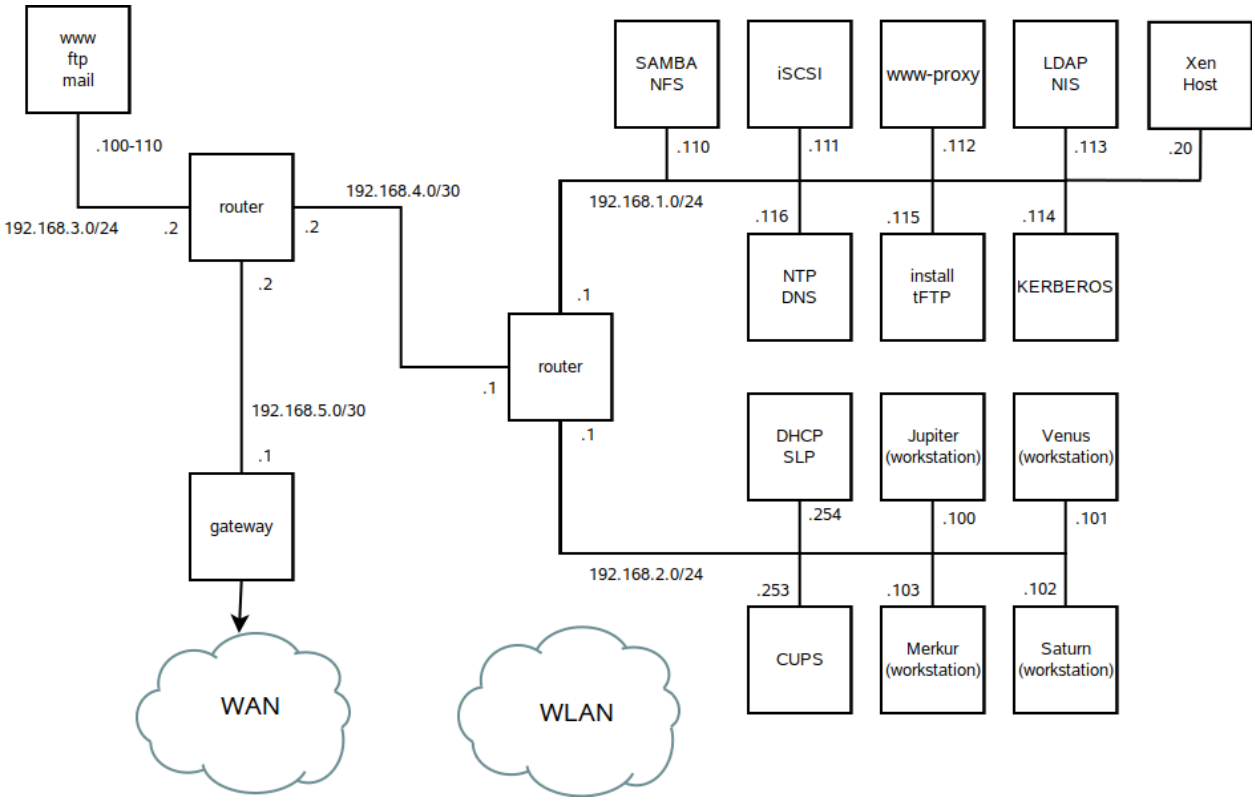
Refer to the kernel sources to see if your processor is supported. You may need a special kernel module or module option to activate CPU frequency control. If the `kernel-source` package is installed, this information is available in `/usr/src/linux/Documentation/cpu-freq/*`.

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

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or 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.