

The Media Streaming Journal

July 2020



Covering Audio and Video Internet Broadcasting

Brought To You By

RADIOSOLUTION

www.radiosolution.info



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

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

Welcome to The Media Streaming Journal

Welcome to the latest installment of The Media Streaming Journal.

In this edition, we cover the latest version of Opensuse, Leap 15.2. Opensuse is a Linux distribution that is developed as a community edition (Free) and an enterprise version (Paid). The community edition is updated regularly and has a vast assortment of current and stable applications that are suitable for production work. Opensuse is also capable of being deployed on older desktops, laptops, and servers.

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

The Grand Master of Digital Disaster
(Editor In Chief)



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

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

30 Creative Commons Computer, Technical and Internet Broadcasting Guides

Newspaper Interviews

New York Times

Lagniappe - "Something Extra for Mobile"

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

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.ScenicTelevision.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.organicdtm.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 15.2

Security Guide
openSUSE Leap 15.2

System Analysis and Tuning Guide
openSUSE Leap 15.2

Virtualization Guide
openSUSE Leap 15.2

AutoYaST Guide
openSUSE Leap 15.2



Join our technical discussion on Facebook

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

Magazine cover:

https://commons.wikimedia.org/wiki/File:Automate_industriel_WAGO_pour_un_syst%C3%A8me_de_monitoring_en_industrie_pharmaceutique.jpg

**The Media Streaming Journal is licensed under the
Attribution-ShareAlike 4.0 International
(CC BY-SA 4.0)
Creative Commons License.**

www.creativecommons.org/licenses/by-sa/4.0/



RADIO SOLUTION

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 15.2

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

- Installation
- Administration
- Managing and Updating Software
- The Bash Shell
- Help and Troubleshooting

Security Guide

openSUSE Leap 15.2

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.

System Analysis and Tuning Guide

openSUSE Leap 15.2

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.

Virtualization Guide

openSUSE Leap 15.2

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

AutoYaST Guide

openSUSE Leap 15.2

AutoYaST is a system for unattended mass deployment of openSUSE Leap systems. AutoYaST installations are performed using an AutoYaST control file (also called a “profile”) with your customized installation and configuration data.

Hey You! Yes, You! Why Should Anyone Listen to You?!

Do you need compelling, clever copy or catchphrases for your Internet station? If you do, please visit and lets talk!

<http://www.ielectrify.com/work-with-me/>

I am a professional writer with 15+ years of experience creating high-converting copy, for a variety of radio, broadcasting and marketing applications.



https://www.wpclipart.com/people/professions/professions_3/radio_announcer.png.html



Start-Up

openSUSE Leap 15.2



Start-Up

openSUSE Leap 15.2

Publication Date: July 06, 2020

SUSE LLC

1800 South Novell Place

Provo, UT 84606

USA

<https://documentation.suse.com> ↗

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

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

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

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

Contents

About This Guide **xi**

I INSTALLATION **1**

1 *Installation Quick Start* **2**

1.1 Welcome to openSUSE Leap **2**

Minimum System Requirements **2** • Installing openSUSE Leap **2**

2 *Boot Parameters* **17**

2.1 Using the Default Boot Parameters **17**

2.2 PC (AMD64/Intel 64/Arm AArch64) **17**

The Boot Screen on Machines Equipped with Traditional BIOS **18** • The Boot Screen on Machines Equipped with UEFI **20**

2.3 List of Important Boot Parameters **23**

General Boot Parameters **23** • Configuring the Network Interface **24** • Specifying the Installation Source **26** • Specifying Remote Access **27**

2.4 Advanced Setups **27**

Using IPv6 for the Installation **27** • Using a Proxy for the Installation **28** • Enabling SELinux Support **28** • Enabling the Installer Self-Update **28** • Scale User Interface for High DPI **29** • Using CPU Mitigations **29**

2.5 More Information **29**

3 *Installation Steps* **30**

3.1 Overview **30**

3.2 Installer Self-Update **31**

Self-Update Process **32** • Custom Self-Update Repositories **34**

3.3	Language, Keyboard, and License Agreement	35
3.4	Network Settings	36
3.5	Online Repositories	37
3.6	System Role	39
3.7	Partitioning	42
	Important Information	42
	Suggested Partitioning	44
3.8	Clock and Time Zone	46
3.9	Create New User	48
3.10	Authentication for the System Administrator “root”	51
3.11	Installation Settings	53
	<i>Software</i>	53
	<i>Booting</i>	55
	<i>Security</i>	55
	<i>Network</i>	
	<i>Configuration</i>	56
	<i>Default systemd Target</i>	56
	<i>Import SSH Host Keys and</i>	
	<i>Configuration</i>	57
	<i>System</i>	57
3.12	Performing the Installation	58
4	Troubleshooting	59
4.1	Checking Media	59
4.2	No Bootable Drive Available	59
4.3	Booting from Installation Media Fails	60
4.4	Boot Failure	61
4.5	Fails to Launch Graphical Installer	63
4.6	Only Minimalist Boot Screen Started	64
II	ADMINISTRATION	66
5	Managing Users with YaST	67
5.1	User and Group Administration Dialog	67
5.2	Managing User Accounts	69

- 5.3 Additional Options for User Accounts 70
 - Automatic Login and Passwordless Login 71 • Enforcing Password Policies 71 • Managing Quotas 72
- 5.4 Changing Default Settings for Local Users 74
- 5.5 Assigning Users to Groups 74
- 5.6 Managing Groups 75
- 5.7 Changing the User Authentication Method 76
- 5.8 Default System Users 78
- 6 Changing Language and Country Settings with YaST 80**
- 6.1 Changing the System Language 80
 - Modifying System Languages with YaST 81 • Switching the Default System Language 83 • Switching Languages for Standard X and GNOME Applications 84
- 6.2 Changing the Country and Time Settings 84
- 7 Setting Up Hardware Components with YaST 88**
- 7.1 Setting Up Your System Keyboard Layout 88
- 7.2 Setting Up Sound Cards 88
- 7.3 Setting Up a Printer 92
 - Configuring Printers 92 • Configuring Printing via the Network with YaST 95 • Sharing Printers over the Network 97
- 7.4 Setting Up a Scanner 98
 - Configuring an HP All-In-One Device 98 • Sharing a Scanner over the Network 99 • Scanning over the Network 99
- 8 Printer Operation 100**
- 8.1 The CUPS Workflow 101
- 8.2 Methods and Protocols for Connecting Printers 102

8.3	Installing the Software	102
8.4	Network Printers	103
8.5	Configuring CUPS with Command Line Tools	104
8.6	Printing from the Command Line	105
8.7	Special Features in openSUSE Leap	106
	CUPS and Firewall	106
	Browsing for Network Printers	106
	PPD Files in Various Packages	107
8.8	Troubleshooting	108
	Printers without Standard Printer Language Support	108
	No Suitable PPD File Available for a PostScript Printer	109
	Network Printer Connections	109
	Defective Printouts without Error Message	111
	Disabled Queues	112
	CUPS Browsing: Deleting Print Jobs	112
	Defective Print Jobs and Data Transfer Errors	112
	Debugging CUPS	113
	For More Information	113
9	Accessing File Systems with FUSE	114
9.1	Configuring FUSE	114
9.2	Mounting an NTFS Partition	114
9.3	Mounting Remote File System with SSHFS	115
9.4	Mounting an ISO File System	115
9.5	Available FUSE Plug-ins	116
9.6	For More Information	117
III	MANAGING AND UPDATING SOFTWARE	118
10	Installing or Removing Software	119
10.1	Definition of Terms	119

10.2	Using the YaST Software Manager	121
	Searching Software	121
	Installing and Removing Packages or Patterns	122
	Updating Packages	124
	Package Dependencies	126
	Handling of Package Recommendations	127
10.3	Managing Software Repositories and Services	128
	Adding Software Repositories	128
	Managing Repository Properties	130
	Managing Repository Keys	131
10.4	The GNOME Package Updater	131
10.5	Updating Packages with GNOME Software	134
11	Installing Add-On Products	136
11.1	Add-Ons	136
11.2	Binary Drivers	137
12	YaST Online Update	138
12.1	The Online Update Dialog	138
12.2	Installing Patches	140
12.3	Automatic Online Update	141
13	Upgrading the System and System Changes	144
13.1	Upgrading the System	144
	Preparations	145
	Possible Problems	145
	Upgrading with YaST	146
	Distribution Upgrade with Zypper	153
	Updating Individual Packages	156
13.2	Additional Information	156
IV	THE BASH SHELL	157
14	Shell Basics	158
14.1	Starting a Shell	158

- 14.2 Entering Commands 159
 - Using Commands without Options 160 • Using Commands with Options 160 • Bash Shortcut Keys 162
- 14.3 Getting Help 162
- 14.4 Working with Files and Directories 163
 - Examples for Working with Files and Directories 165
- 14.5 Becoming Root 168
 - Using **su** 168 • Using **sudo** 168
- 14.6 File Access Permissions 169
 - Permissions for User, Group and Others 169 • Files and Folders 171 • Modifying File Permissions 172
- 14.7 Time-Saving Features of Bash 174
 - Examples For Using History, Completion and Wildcards 175
- 14.8 Editing Texts 178
 - Example: Editing with vi 179
- 14.9 Searching for Files or Contents 179
 - Examples for Searching 180
- 14.10 Viewing Text Files 180
- 14.11 Redirection and Pipes 181
 - Examples for Redirection and Pipe 182
- 14.12 Starting Programs and Handling Processes 183
- 14.13 Archives and Data Compression 184
- 14.14 Important Linux Commands 186
 - File Commands 186 • System Commands 192 • For More Information 195
- 15 Bash and Bash Scripts 196**
 - 15.1 What is “The Shell”? 196
 - Bash Configuration Files 196 • The Directory Structure 199

- 15.2 Writing Shell Scripts 203
- 15.3 Redirecting Command Events 204
- 15.4 Using Aliases 205
- 15.5 Using Variables in Bash 206
 - Using Argument Variables 207 • Using Variable Substitution 207
- 15.6 Grouping and Combining Commands 208
- 15.7 Working with Common Flow Constructs 209
 - The if Control Command 209 • Creating Loops with the **for** Command 210
- 15.8 For More Information 210

V HELP AND TROUBLESHOOTING 211

16 Help and Documentation 212

- 16.1 Documentation Directory 212
 - SUSE Manuals 213 • Package Documentation 213
- 16.2 Man Pages 214
- 16.3 Info Pages 215
- 16.4 Online Resources 216

17 Common Problems and Their Solutions 217

- 17.1 Finding and Gathering Information 217
- 17.2 Boot Problems 220
 - The GRUB 2 Boot Loader Fails to Load 220 • No Login or Prompt Appears 221 • No Graphical Login 222 • Root Btrfs Partition Cannot Be Mounted 222 • Force Checking Root Partitions 222 • Disable Swap to Enable Booting 223
- 17.3 Login Problems 223
 - Valid User Name and Password Combinations Fail 223 • Valid User Name and Password Not Accepted 224 • Login to Encrypted Home Partition Fails 227 • GNOME Desktop Has Issues 227

- 17.4 Network Problems 228
 - NetworkManager Problems 232
- 17.5 Data Problems 233
 - Managing Partition Images 233 • Using the Rescue System 234
- A GNU Licenses 241**
 - A.1 GNU Free Documentation License 241

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

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.

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

2 Giving Feedback

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

Bug Reports

Report issues with the documentation at <https://bugzilla.opensuse.org/>. To simplify this process, you can use the *Report Documentation Bug* links next to headlines in the HTML version of this document. These preselect the right product and category in Bugzilla and add a link to the current section. You can start typing your bug report right away. A Bugzilla account is required.

Contributions

To contribute to this documentation, use the *Edit Source* links next to headlines in the HTML version of this document. They take you to the source code on GitHub, where you can open a pull request. A GitHub account is required.

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

Mail

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

Help

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

3 Documentation Conventions

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

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

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

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

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

```
tux > command
```

- Notices



Warning: Warning Notice

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



Important: Important Notice

Important information you should be aware of before proceeding.



Note: Note Notice

Additional information, for example about differences in software versions.



Tip: Tip Notice

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

4 Source Code

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

5 Acknowledgments

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

I Installation

1 *Installation Quick Start* **2**

2 Boot Parameters **17**

3 Installation Steps **30**

4 Troubleshooting **59**

1 *Installation Quick Start*

Use the following procedures to install a new version of openSUSE® Leap 15.2. This document gives a quick overview on how to run through a default installation of openSUSE Leap on the x86_64 architecture.

For installing the AArch64 and POWER variants, see <https://en.opensuse.org/Portal:ARM> and <https://en.opensuse.org/Portal:PowerPC>.

1.1 Welcome to openSUSE Leap

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

1.1.1 Minimum System Requirements

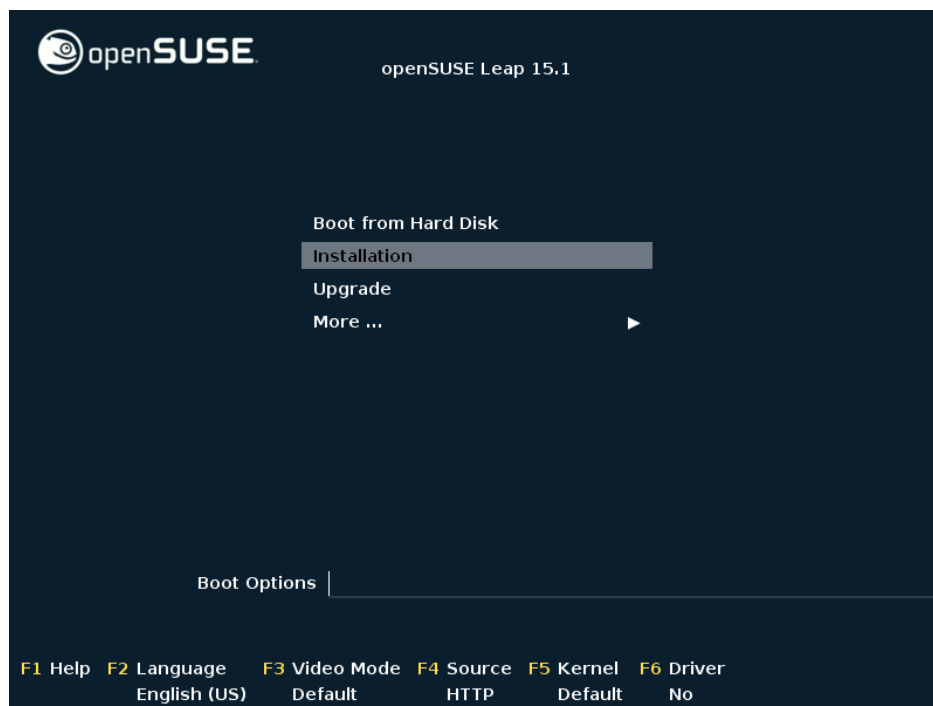
- any AMD64/Intel* EM64T processor (32-bit processors are not supported)
- 1 GB physical RAM (4 GB or more strongly recommended)
- 10 GB available disk space for a minimal installation, 16 GB for a graphical desktop (more is recommended). In case you plan to use Btrfs snapshots a minimum of 40 GB for 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

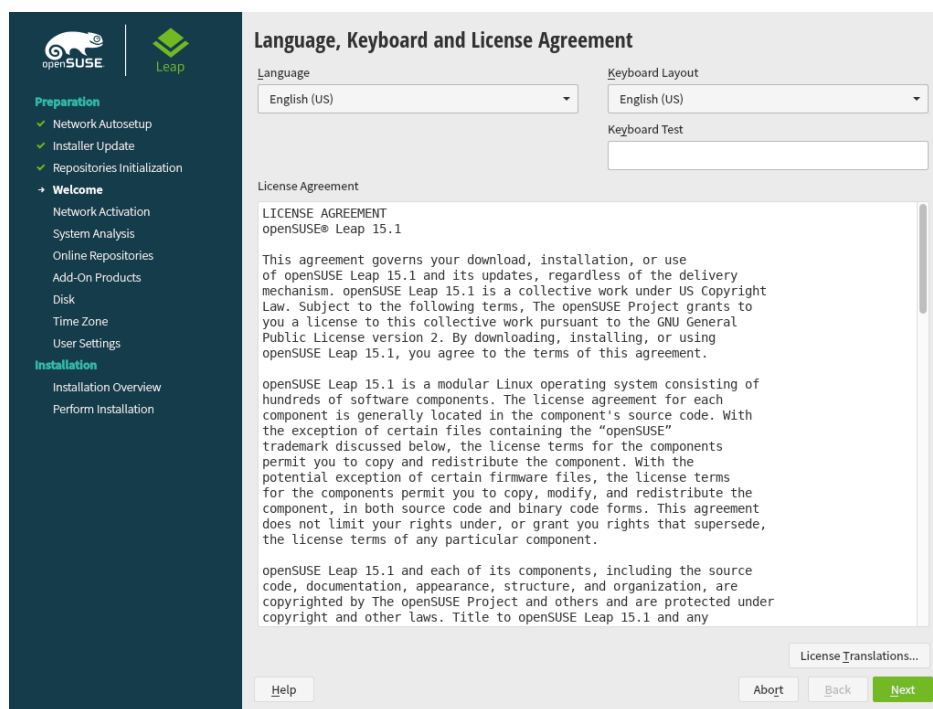
Insert a DVD or a bootable USB stick containing the installation image for openSUSE Leap, then reboot the computer to start the installation program. On machines with a traditional BIOS you will see the graphical boot screen shown below. On machines equipped with UEFI, a slightly different boot screen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

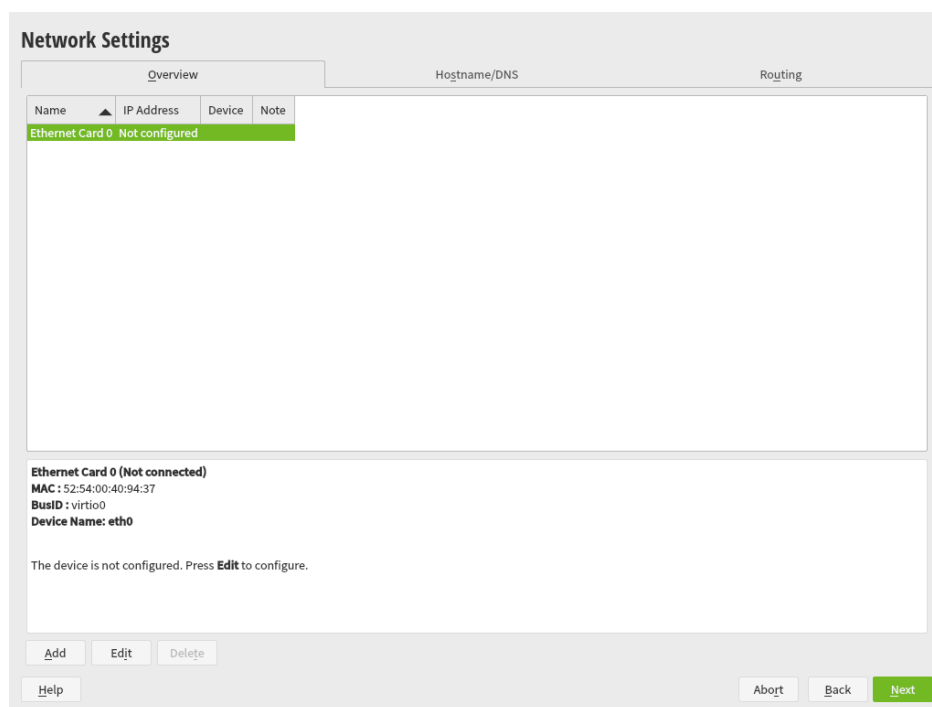
1.1.2.2 Language, Keyboard and License Agreement



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*.

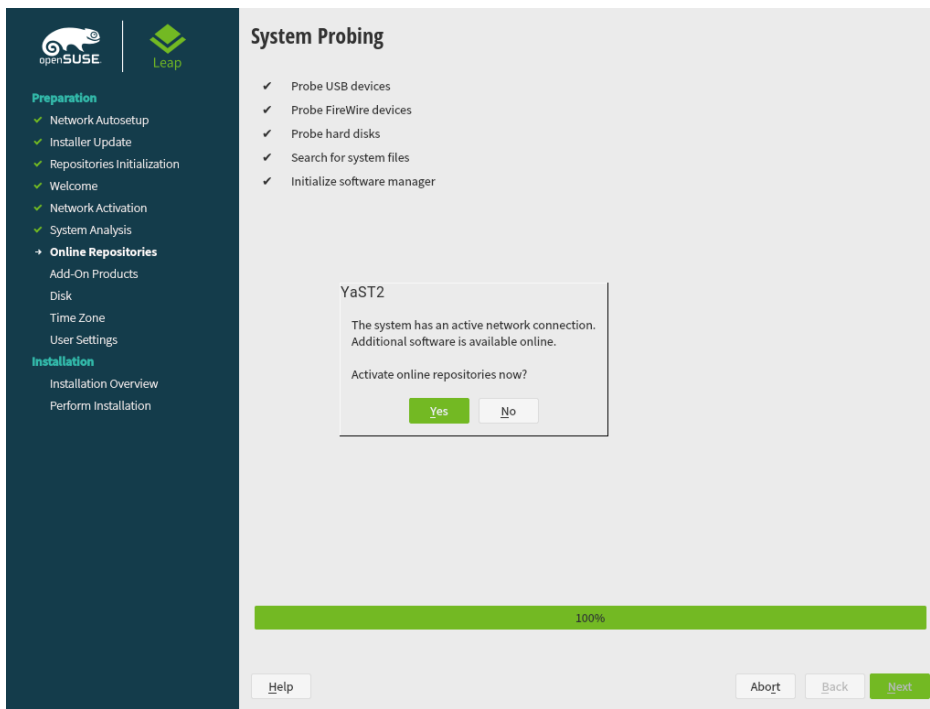
1.1.2.3 Network Settings



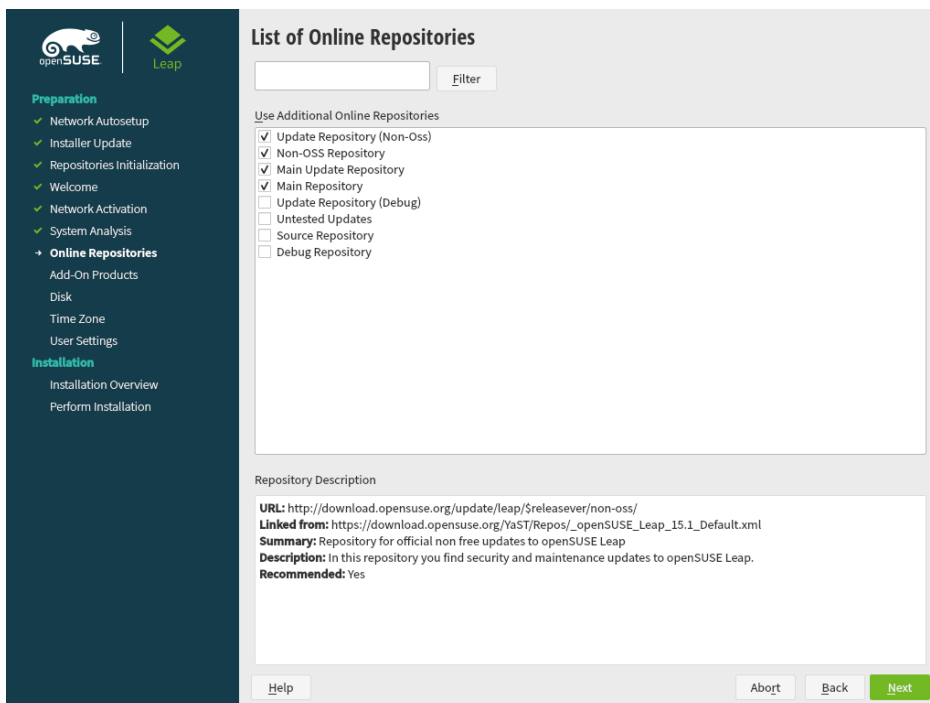
If the network can not be configured automatically, the *Network Settings* dialog opens. Choose a network interface from the list and configure it with *Edit*. Alternatively, *Add* an interface manually. See [Section 3.4, “Network Settings”](#) and *Book “Reference”, Chapter 13 “Basic Networking”, Section 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.

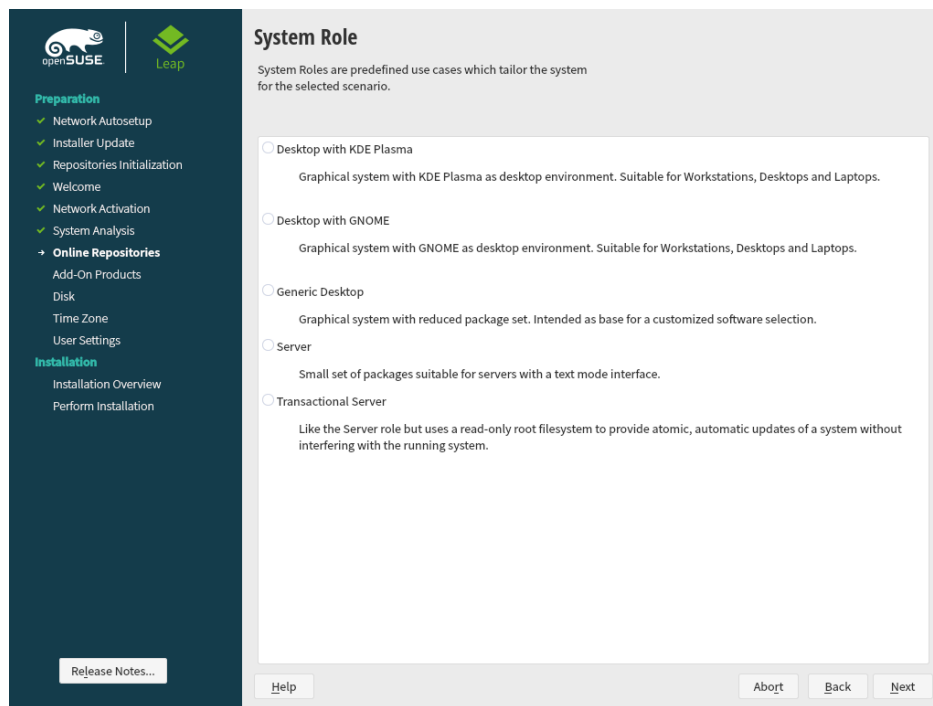


You have the following choices:

- 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 alternative 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.

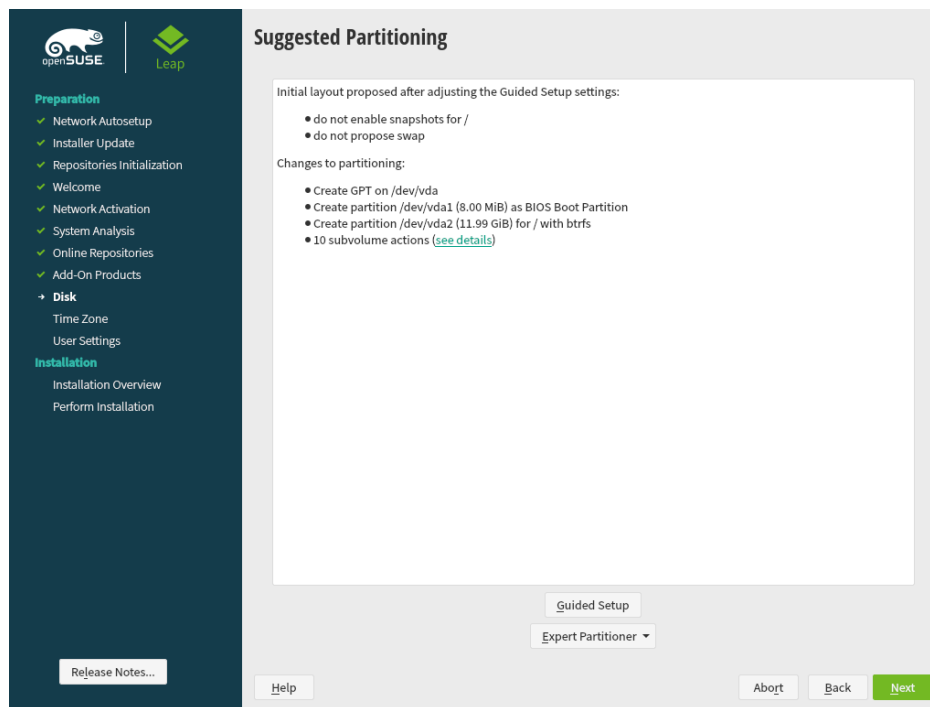
You can also manually choose the software configuration for your system. Select *Custom* and then *Next* to get to the *Software Selection and System Tasks dialog*. Choose one or more patterns for installation. By clicking *Details*, you can select individual packages.



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.



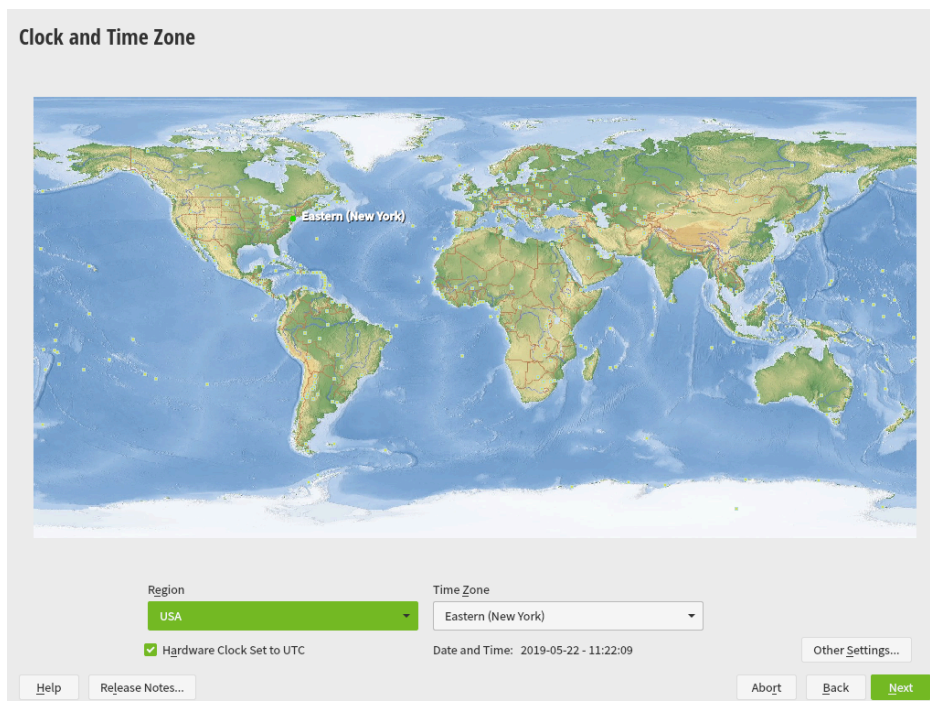
Note: Separate Home Partition

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.

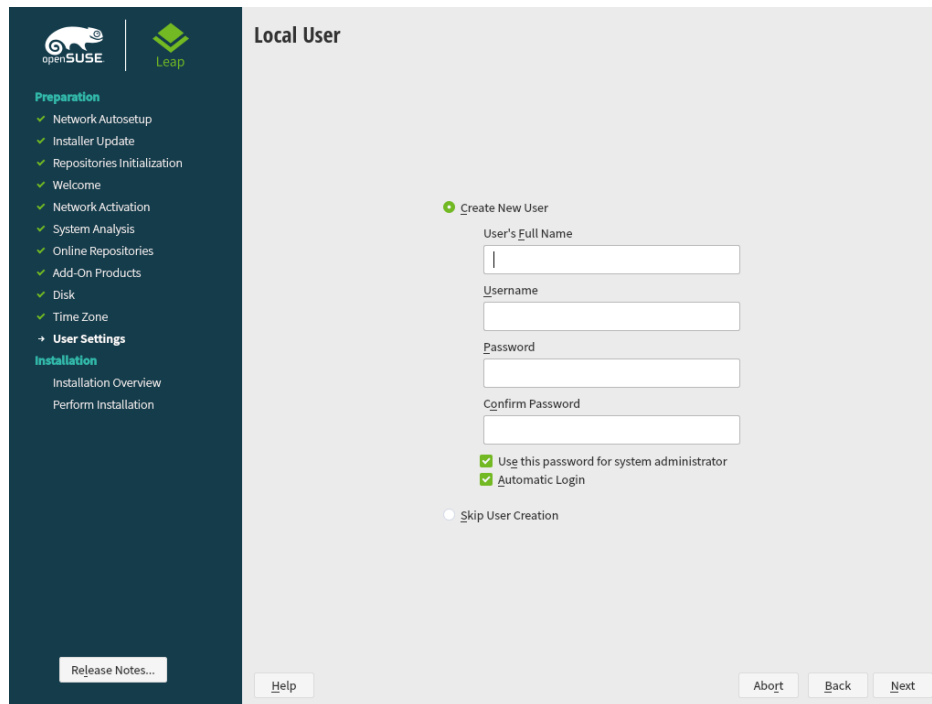
To accept the proposed setup without any changes, choose *Next* to proceed.

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"

Authentication for the System Administrator "root"

Do not forget what you enter here.

Password for root User

Confirm Password

Test Keyboard Layout

Import Public SSH Key

QEMU DVD-ROM (/dev/sr0) Refresh

Browse...

Help Abort Back Next

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*.

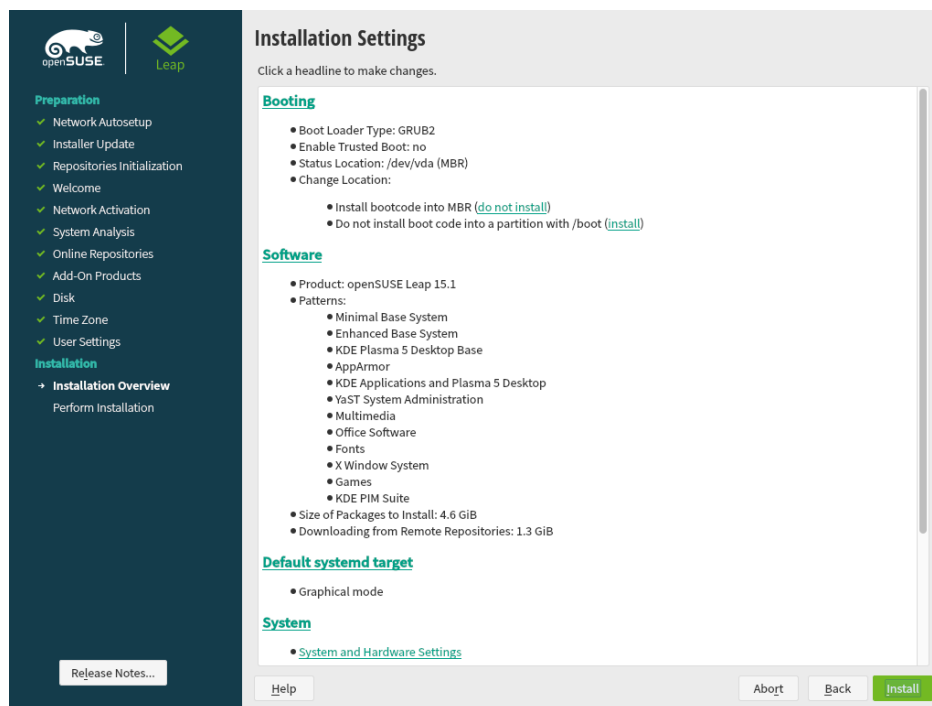


Tip: Passwords and Keyboard Layout

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 key in 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 3.11.7, “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*.

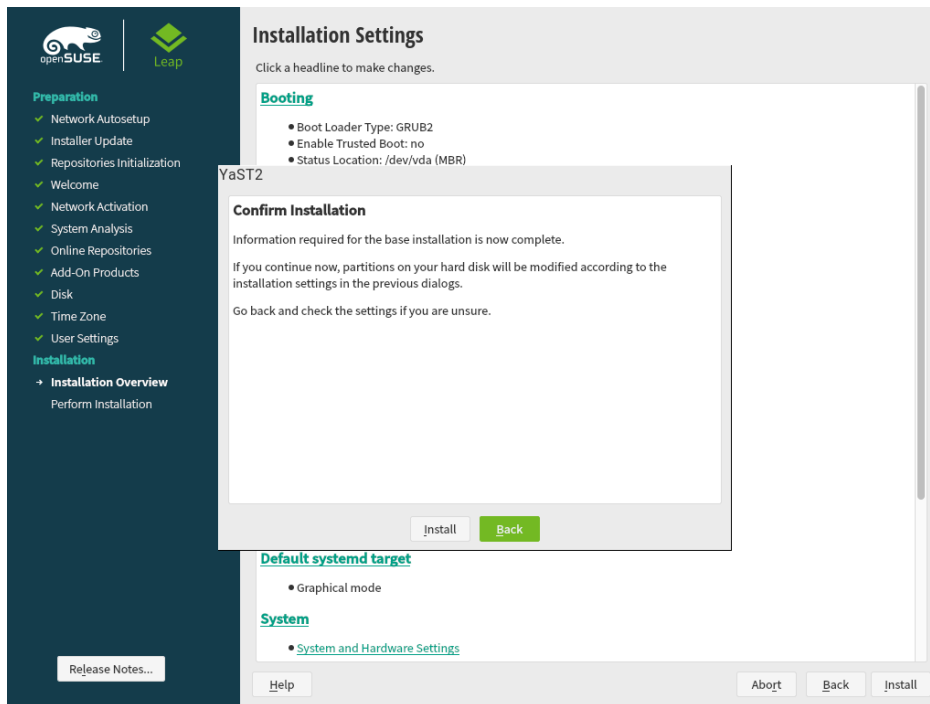
By default, the Firewall is enabled with all network interfaces configured for the public zone. See *Book “Security Guide”, Chapter 18 “Masquerading and Firewalls”, Section 18.4 “firewalld”* for configuration details.

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

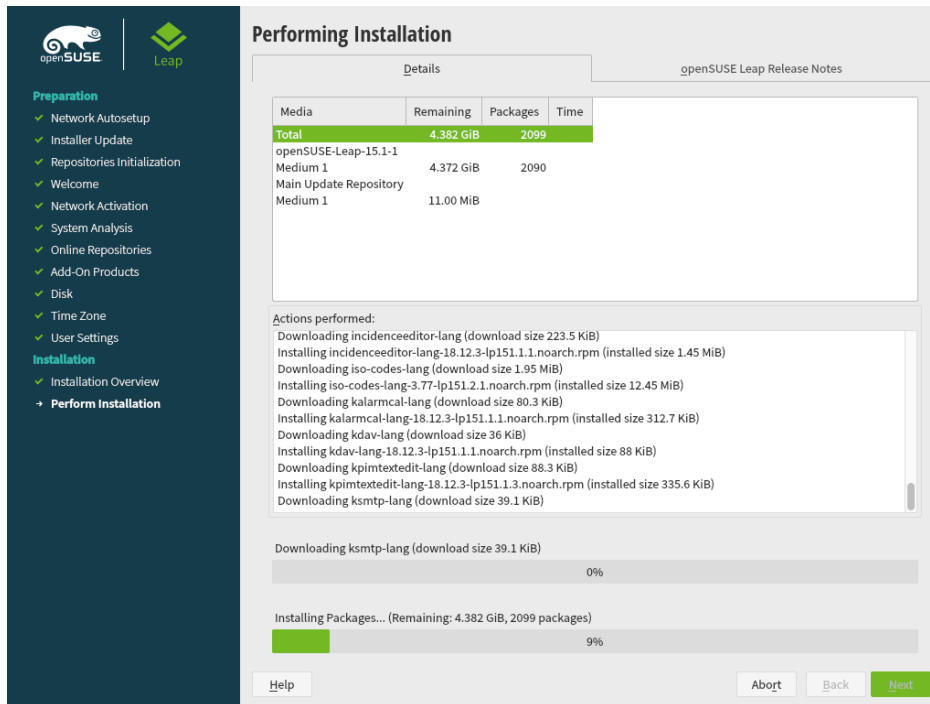
Displays the current network configuration. Click *Network Configuration* to change the settings. For details, see *Book “Reference”, Chapter 13 “Basic Networking”, Section 13.4 “Configuring a Network Connection with YaST”*.

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”](#).

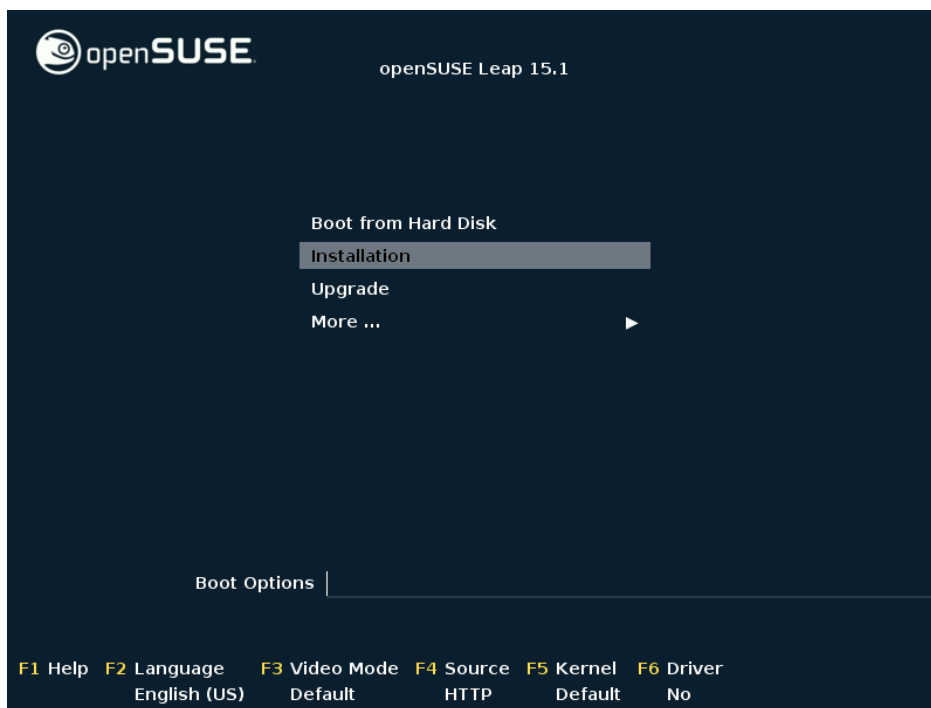


FIGURE 2.1: THE BOOT SCREEN ON MACHINES WITH A TRADITIONAL BIOS

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 or ifcfg parameters.

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]=[.try,]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. If you add the try option, configuration will stop once the installation repository can be reached via one of the configured interfaces.

Alternatively you use static configuration. With static parameters, only the first matching interface will be configured, unless you add the try option. This will configure all interfaces until the repository can be reached.

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"
```



Tip: Other Networking Parameters

The ifcfg boot parameter is very powerful and allows you to set almost all networking parameters. In addition to the parameters mentioned above, you can set values for all configuration options (comma separated) from /etc/sysconfig/network/ifcfg.template and /etc/sysconfig/network/config. The following example sets a custom MTU size on an interface otherwise configured via DHCP:

```
ifcfg=eth0=dhcp,MTU=1500
```

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 DVD or USB flash drive 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: Quarterly Media Update: Self-Update Disabled

The installer self-update is only available if you use the `GM` images of the Unified Installer and Packages ISOs. If you install from the ISOs published as quarterly update (they can be identified by the string `QU` in the name), the installer cannot update itself, because this feature has been disabled in the update media.

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

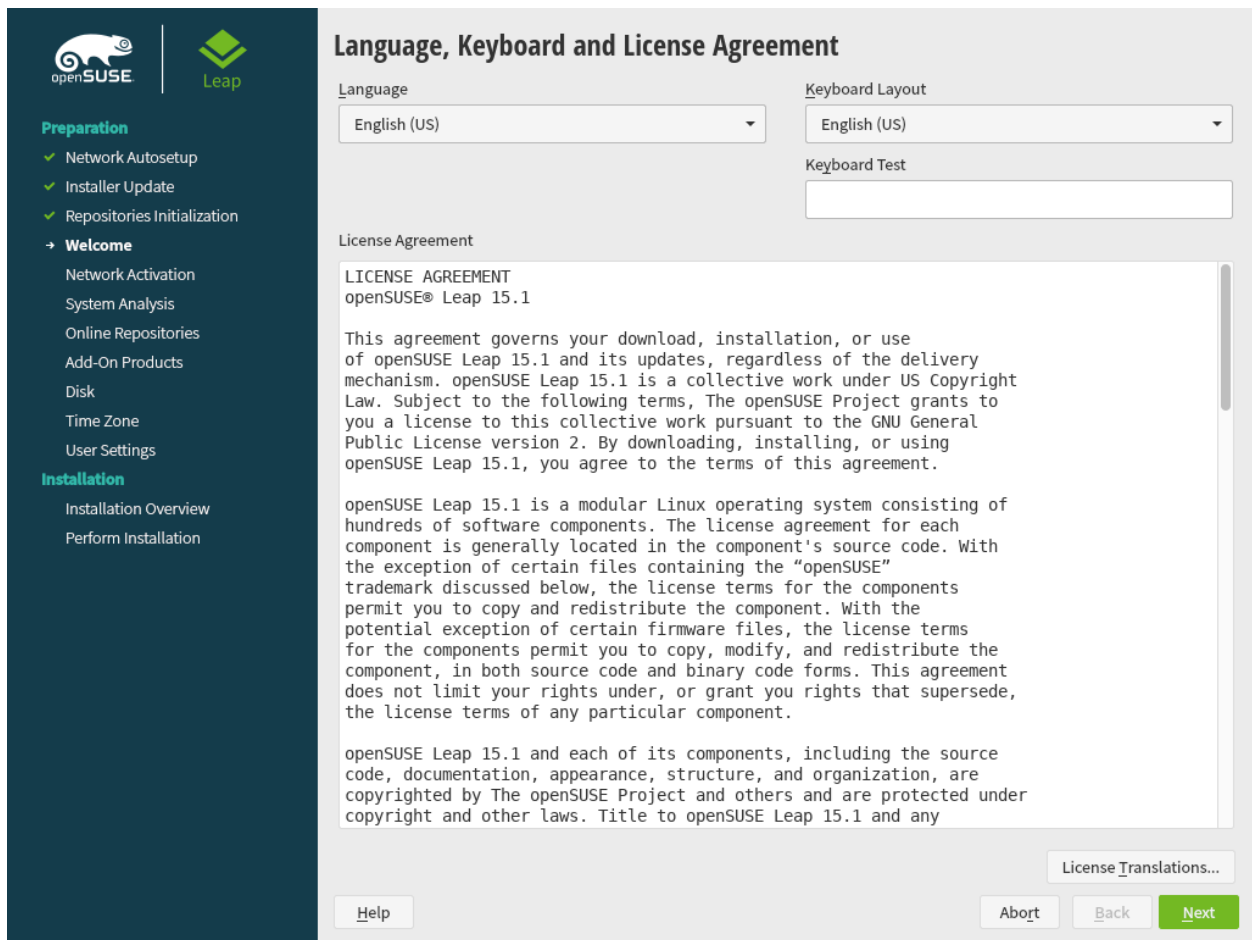


FIGURE 3.1: LANGUAGE, KEYBOARD, AND LICENSE AGREEMENT

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.

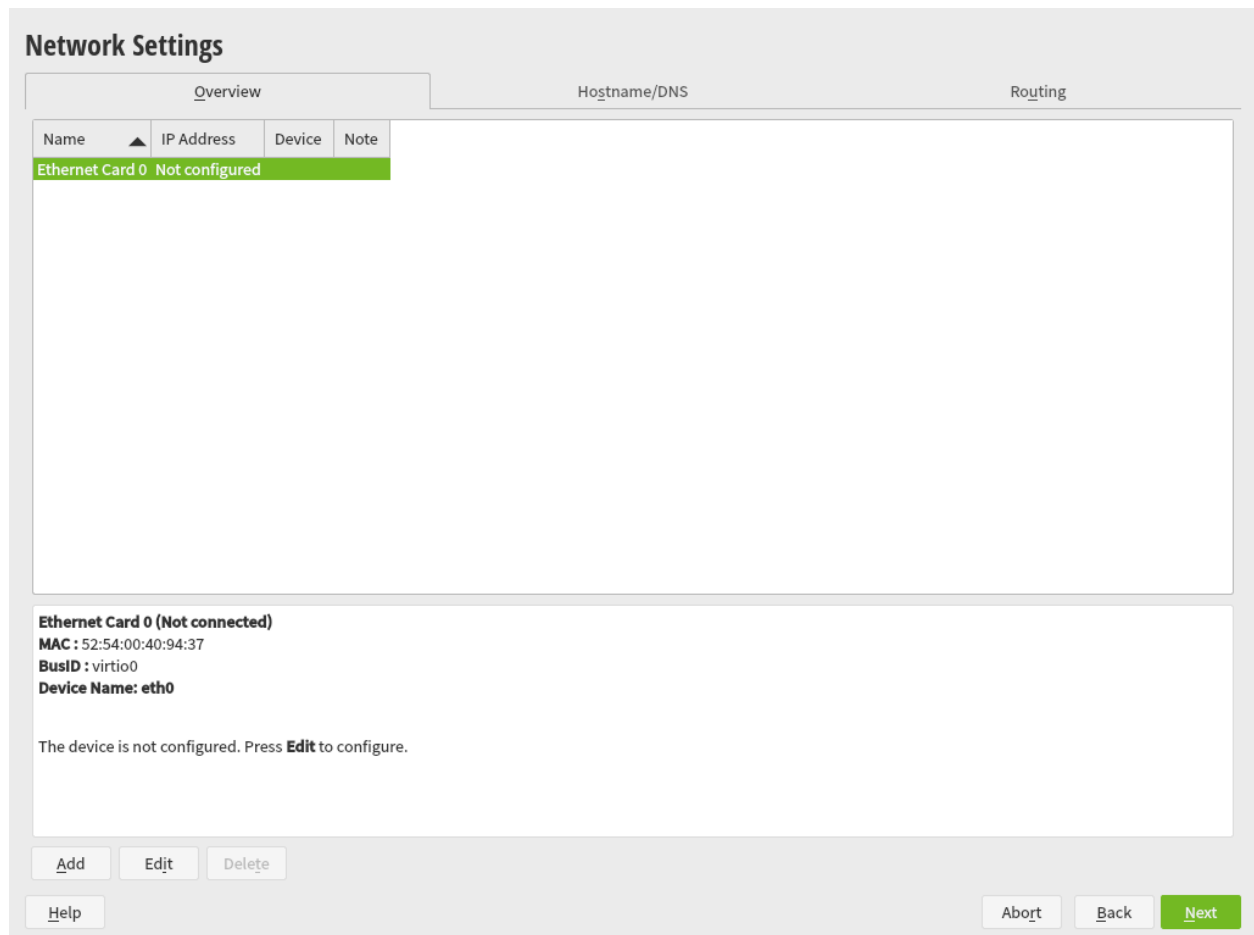


FIGURE 3.2: NETWORK SETTINGS

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

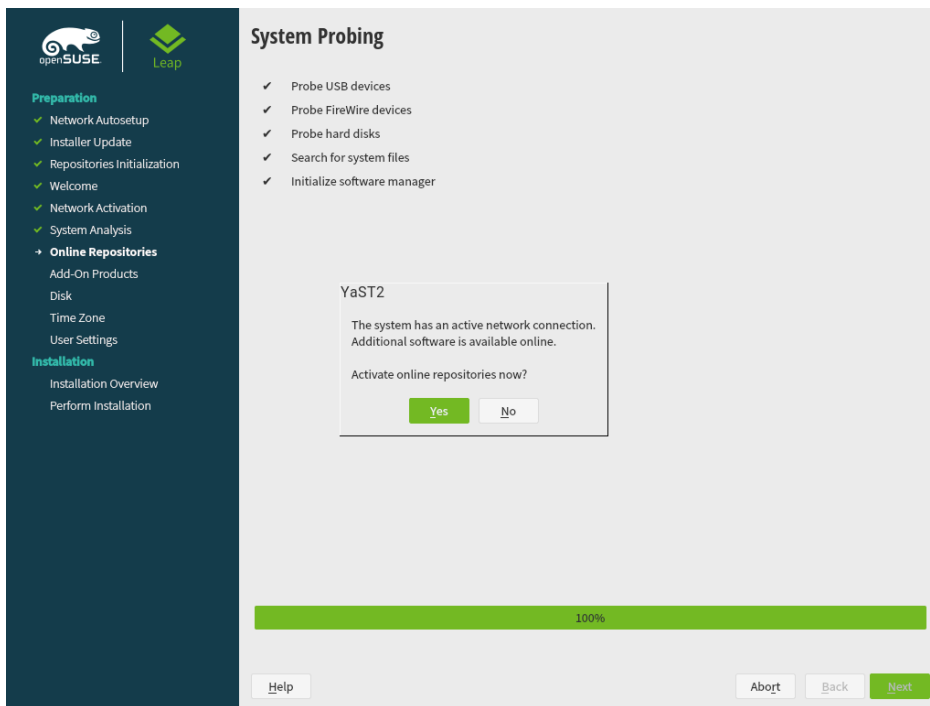
To access a SAN or a local RAID during the installation, you can use the libstorage command line client for this purpose:

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`.

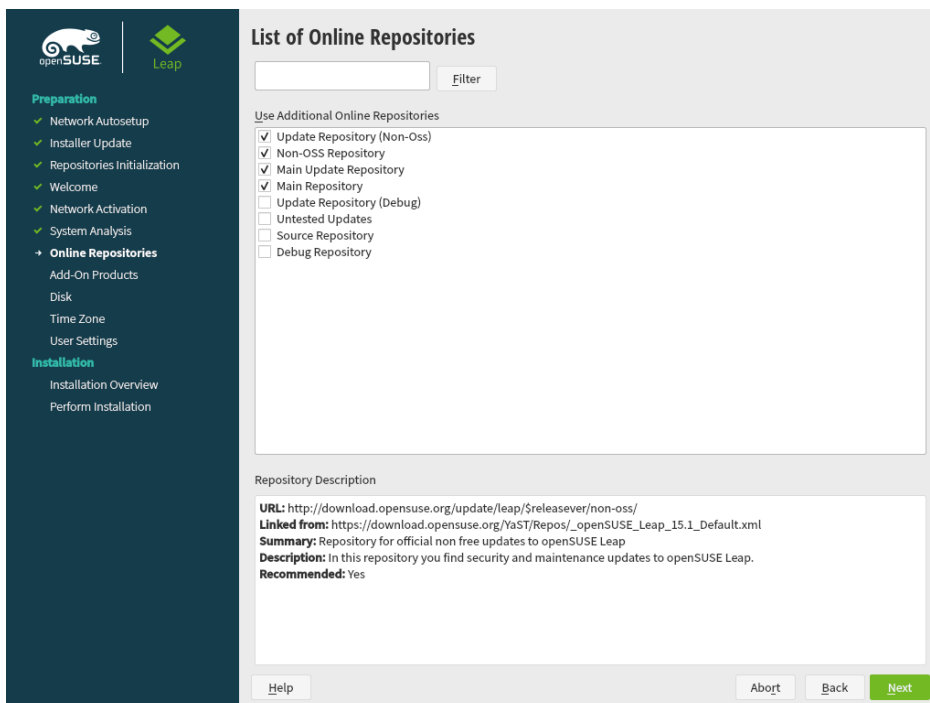
Supported are Netapp Ontap, all SMI-S compatible SAN providers, and LSI MegaRAID.

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.

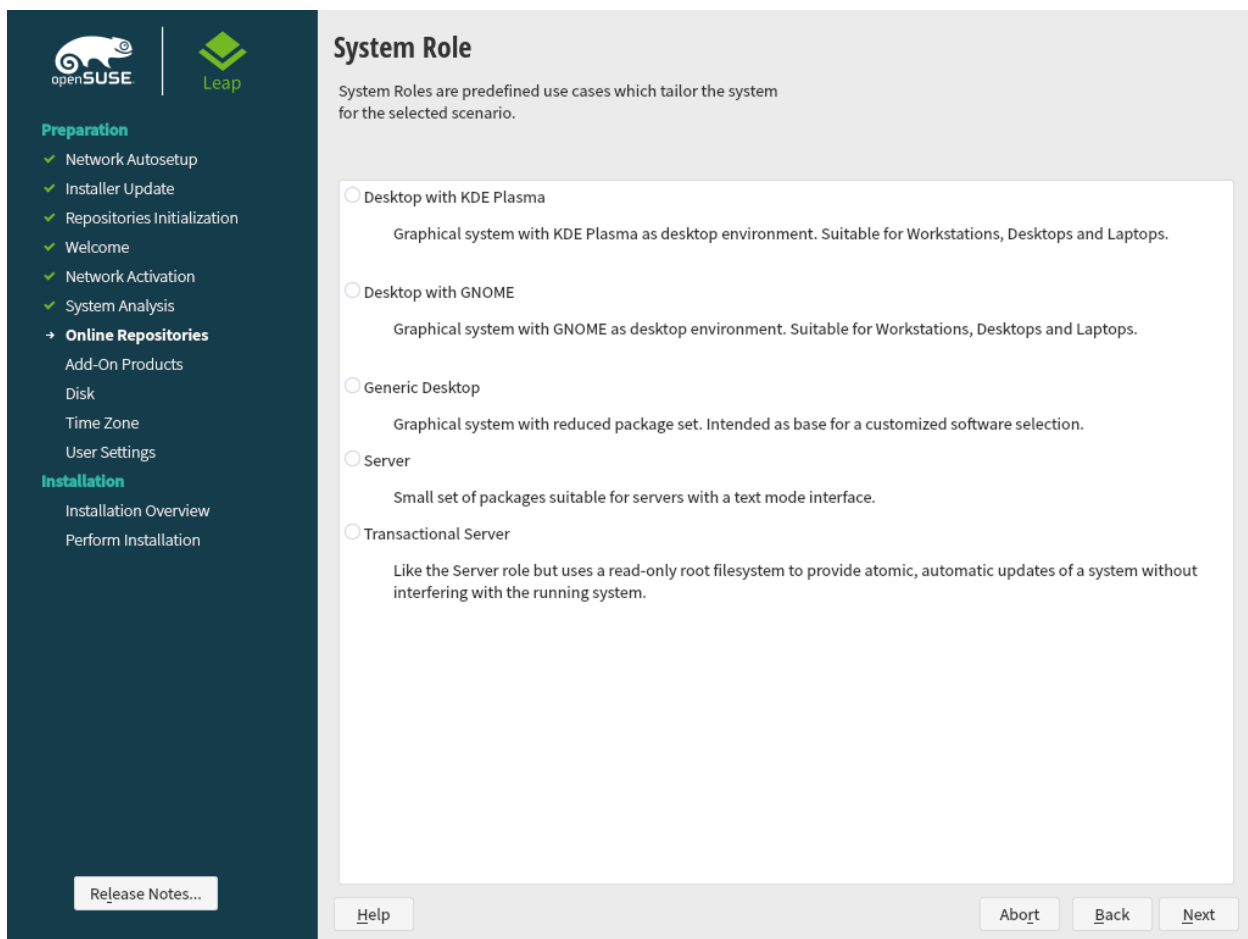


FIGURE 3.3: SYSTEM ROLE

Choose the *System Role* that meets your requirements best. The availability of system roles depends on your selection of modules and extensions. Therefore, the dialog is omitted under the following conditions:

- If from the enabled modules *no role* is suitable for the respective base product. In this case, the installation proceeds with the default settings for this product.
- If from the enabled modules *only one role* is suitable for the respective base product. In this case, the installation proceeds with the settings of this particular role.

With the default selection, the following system roles are available:

Desktop with KDE Plasma

A powerful desktop environment with a complete PIM suite (mail, calendar, tasks, notes, and feeds), widgets running 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 alternative 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`.

If snapshots are enabled, the installer will automatically create single snapshots during and immediately after the installation.

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 disk space 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 15.2. 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
- Intel Virtual RAID on CPU (Intel VROC, see <https://www.intel.com/content/www/us/en/support/articles/000024498/memory-and-storage/ssd-software.html>  for more details)

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.

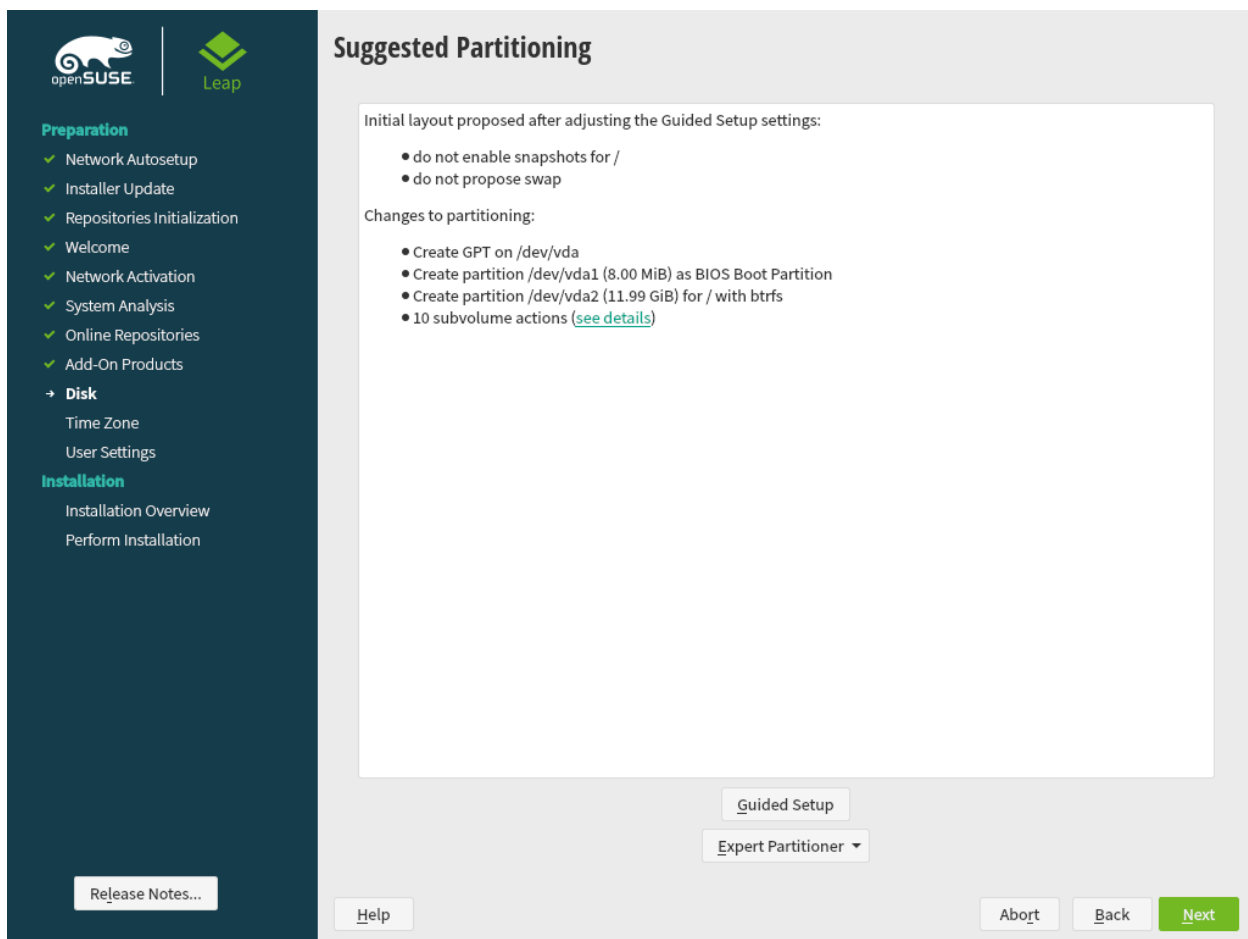


FIGURE 3.4: SUGGESTED PARTITIONING

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. Afterward 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"*.



Warning: Disk Space Units

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

Binary

1 GiB = 1 073 741 824 bytes.

Decimal

1 GB = 1 000 000 000 bytes.

Difference

1 GiB ≈ 1.07 GB.

3.8 Clock and Time Zone

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

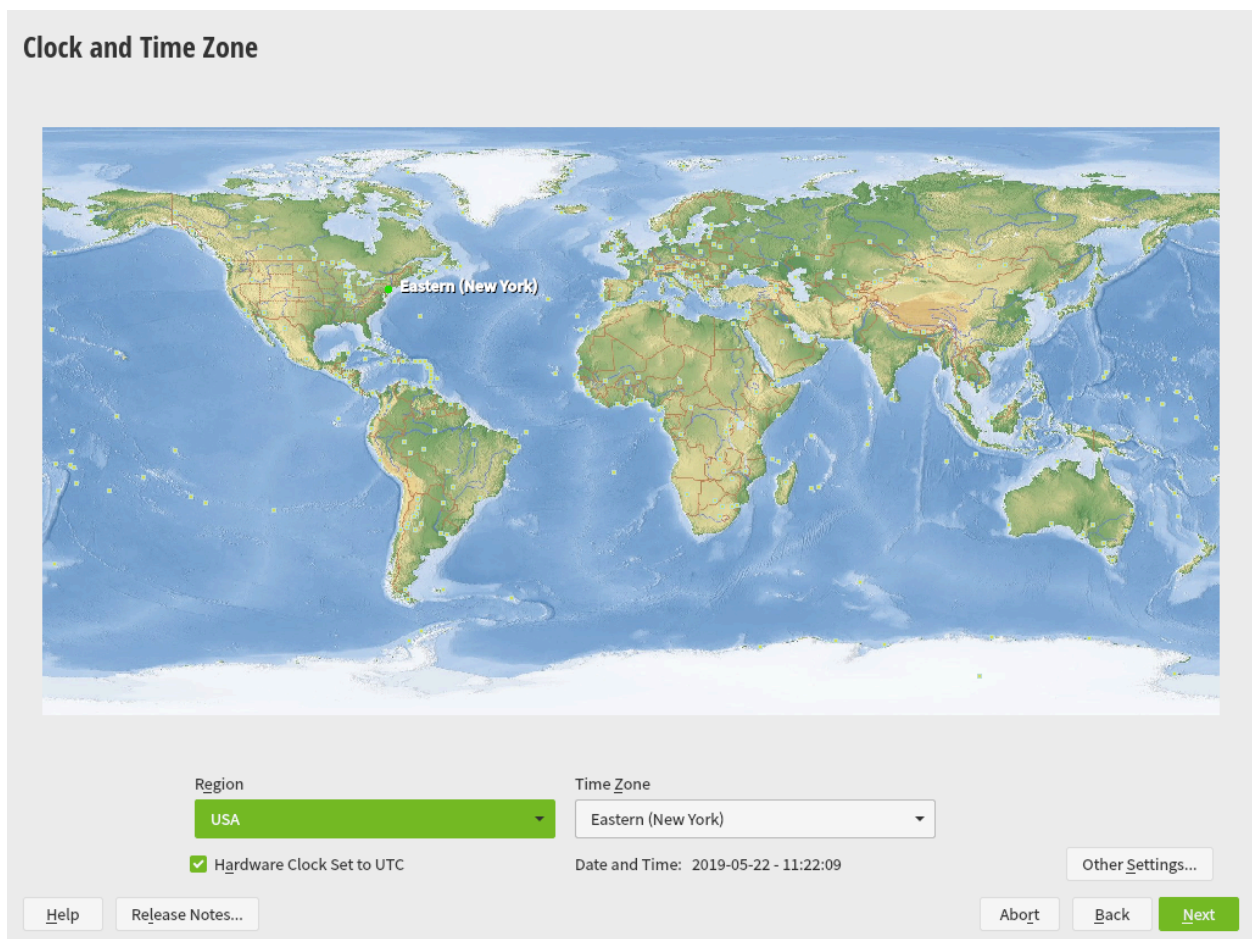


FIGURE 3.5: CLOCK AND TIME ZONE

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.

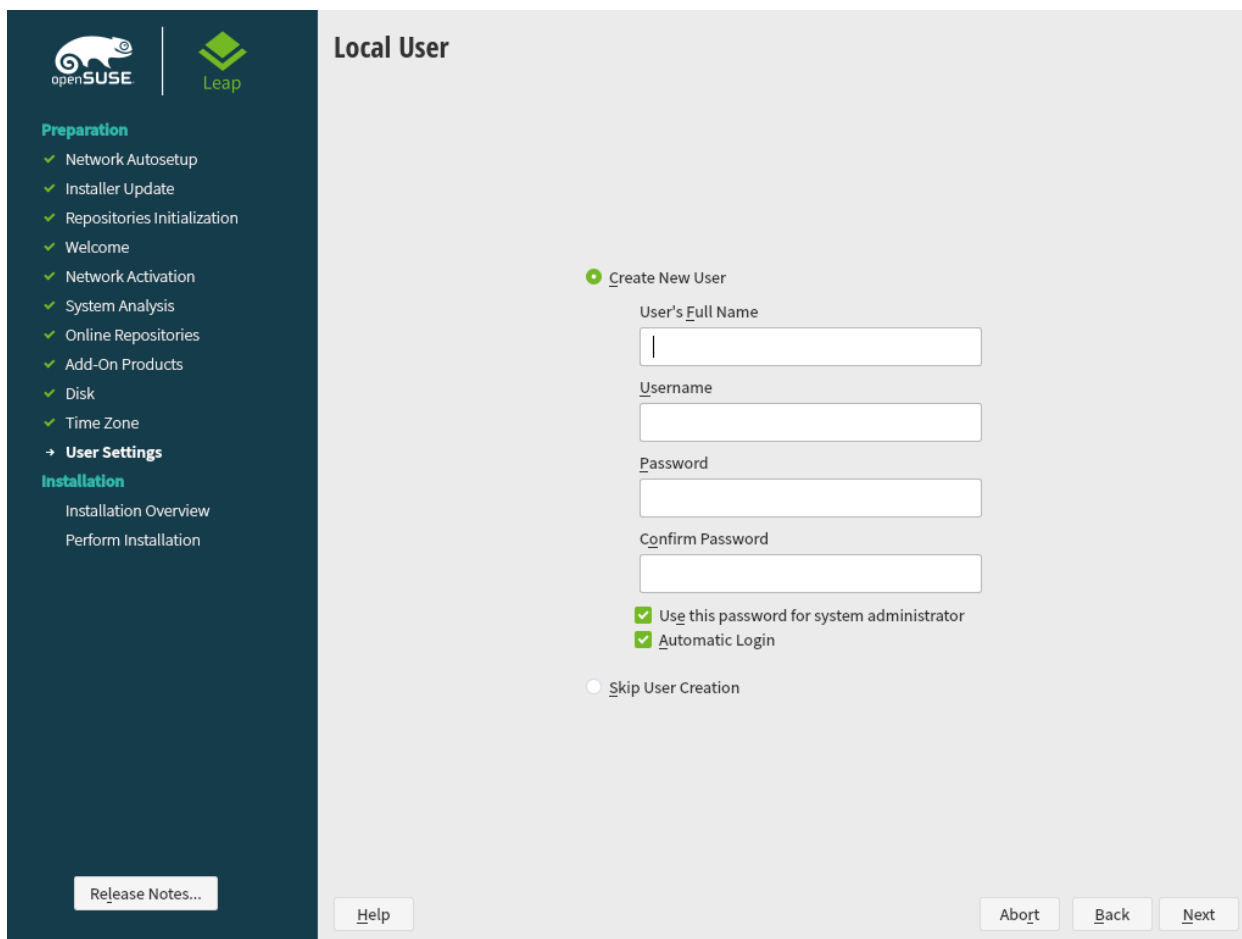


FIGURE 3.6: CREATE NEW USER

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 should 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.

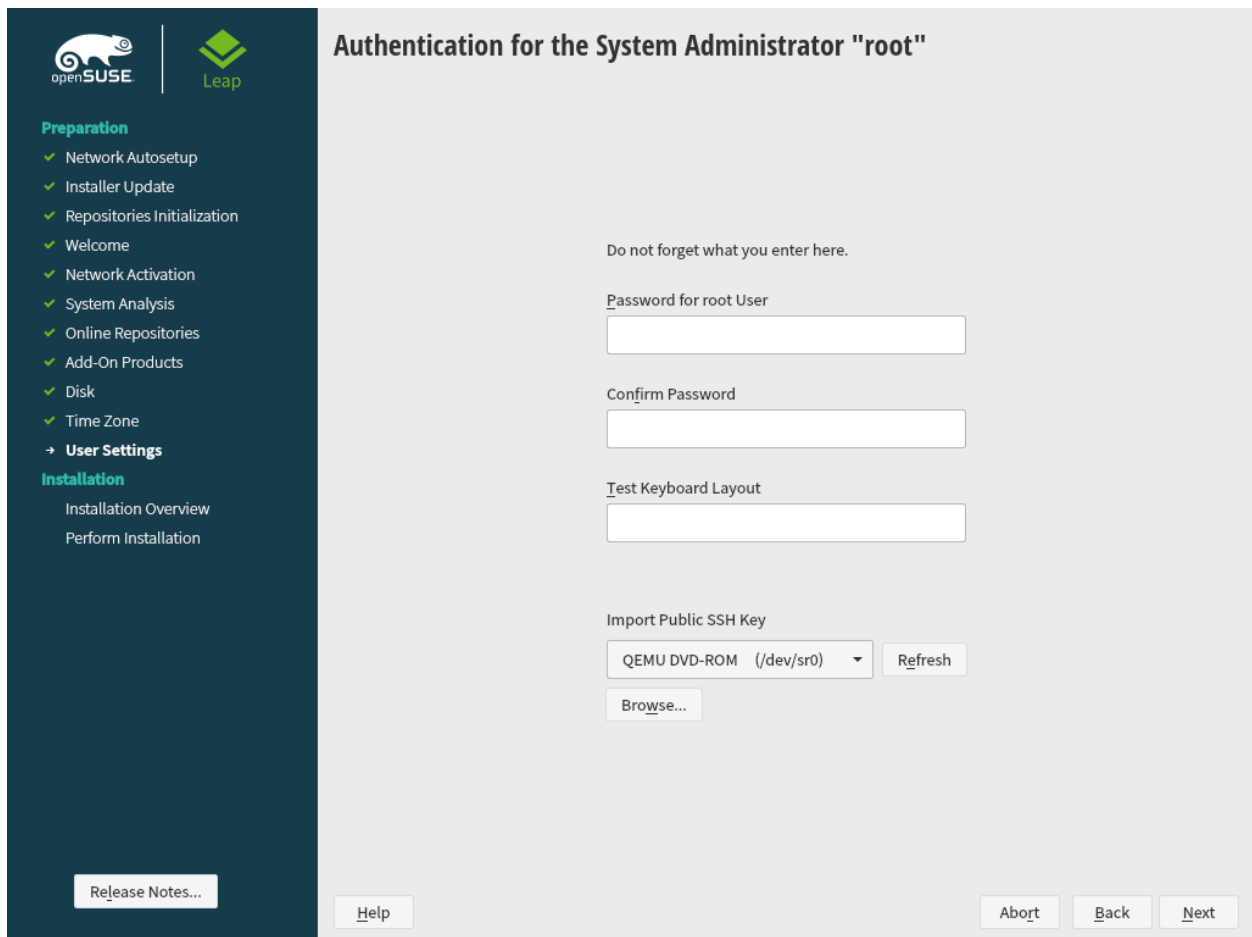


FIGURE 3.7: AUTHENTICATION FOR THE SYSTEM ADMINISTRATOR `root`

`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.

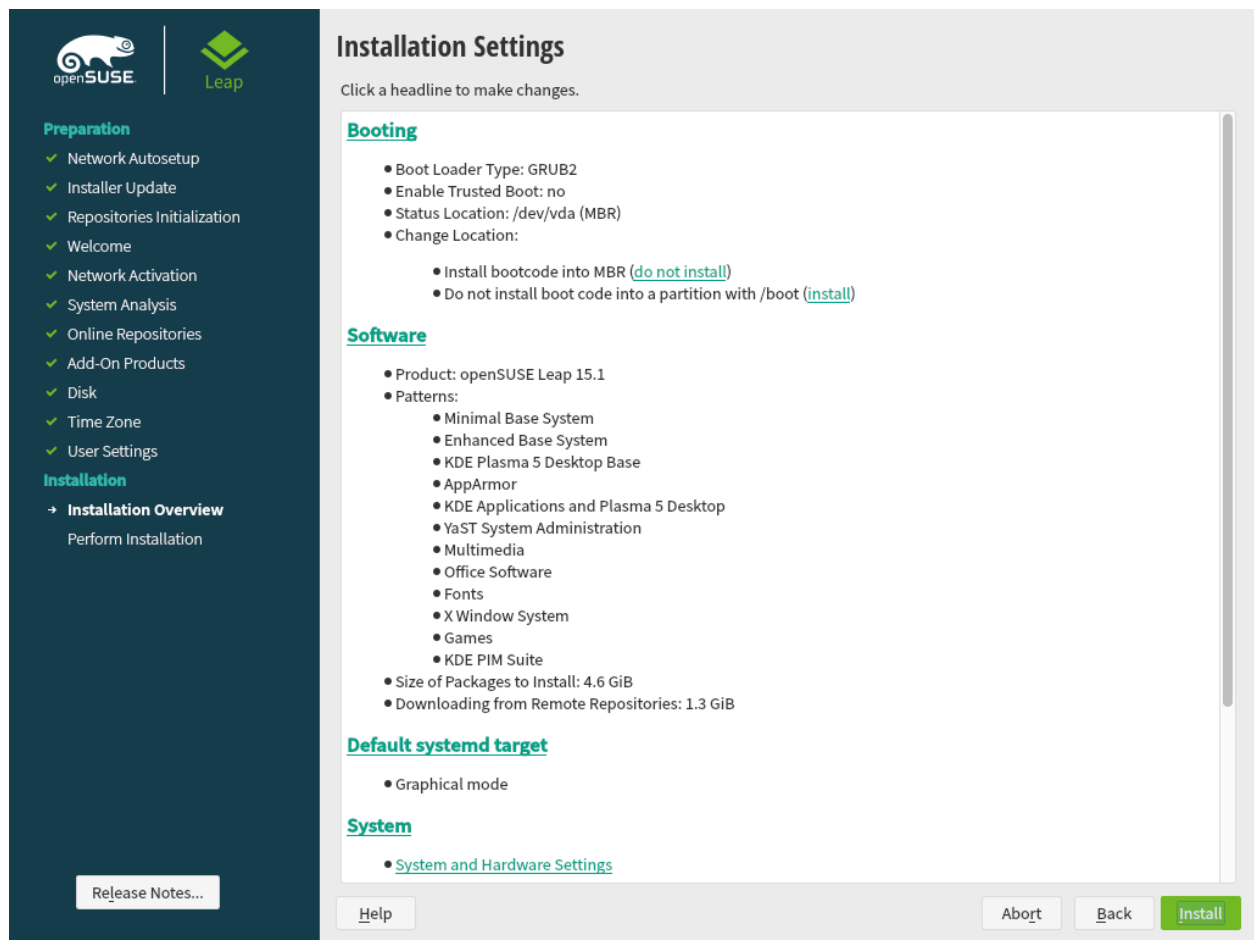


FIGURE 3.8: INSTALLATION SETTINGS

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.

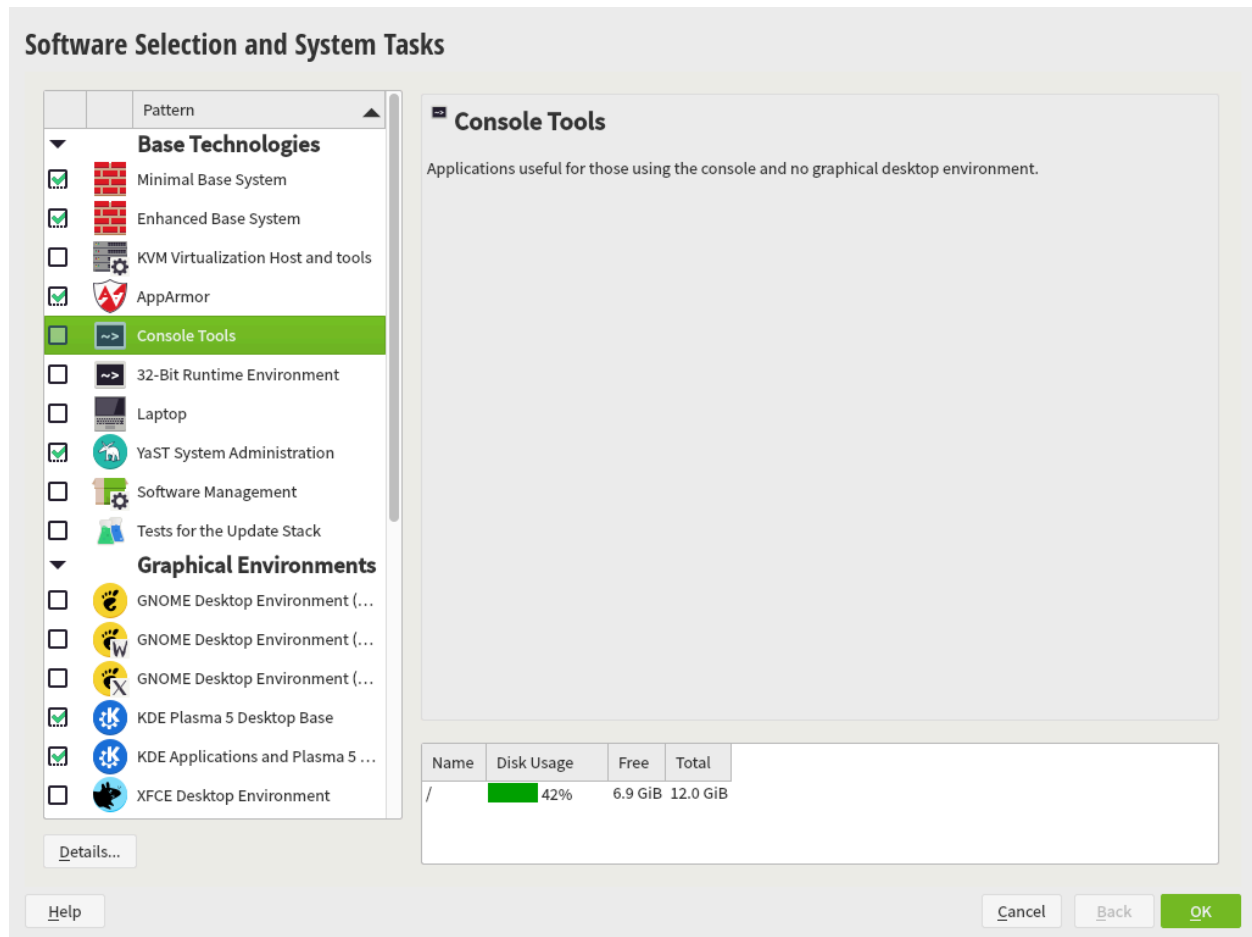


FIGURE 3.9: SOFTWARE SELECTION AND SYSTEM TASKS

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 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 selected entry to choose a different option. For details, see *Book "Reference", Chapter 12 "The Boot Loader GRUB 2" CPU Mitigations*.

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 requir-

ing network access (such as FTP, Samba, Web server, etc.) will only work after having adjusted the firewall settings. Refer to *Book “Security Guide”, Chapter 18 “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 Network Configuration

This category displays the current network settings, as automatically configured after booting into the installation (see [Section 3.4](#)) or as manually configured from the *Registration* or *Add-On Product* dialog during the respective steps of the installation process. If you want to check or adjust the network settings at this stage (before performing the installation), click *Network Configuration*. This takes you to the YaST *Network Settings* module. For details, see *Book “Reference”, Chapter 13 “Basic Networking”, Section 13.4 “Configuring a Network Connection with YaST”*.

3.11.5 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*.

3.11.6 *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.

3.11.7 *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 and 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 Drive Available

If your computer cannot boot from USB or DVD drive, there are several alternatives. This is also an option if your drive is not supported by openSUSE Leap.

Using an External USB Flash Drive or DVD Drive

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

Network Boot via PXE

If a machine lacks both a USB flash drive and 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 USB flash drive or 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 firmware 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 USB flash drive or 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 USB flash drive or 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 USB flash drive or 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 installation media 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](#).

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 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 :

```
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 **67**
- 6 Changing Language and Country Settings with YaST **80**
- 7 Setting Up Hardware Components with YaST **88**
- 8 Printer Operation **100**
- 9 Accessing File Systems with FUSE **114**

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.

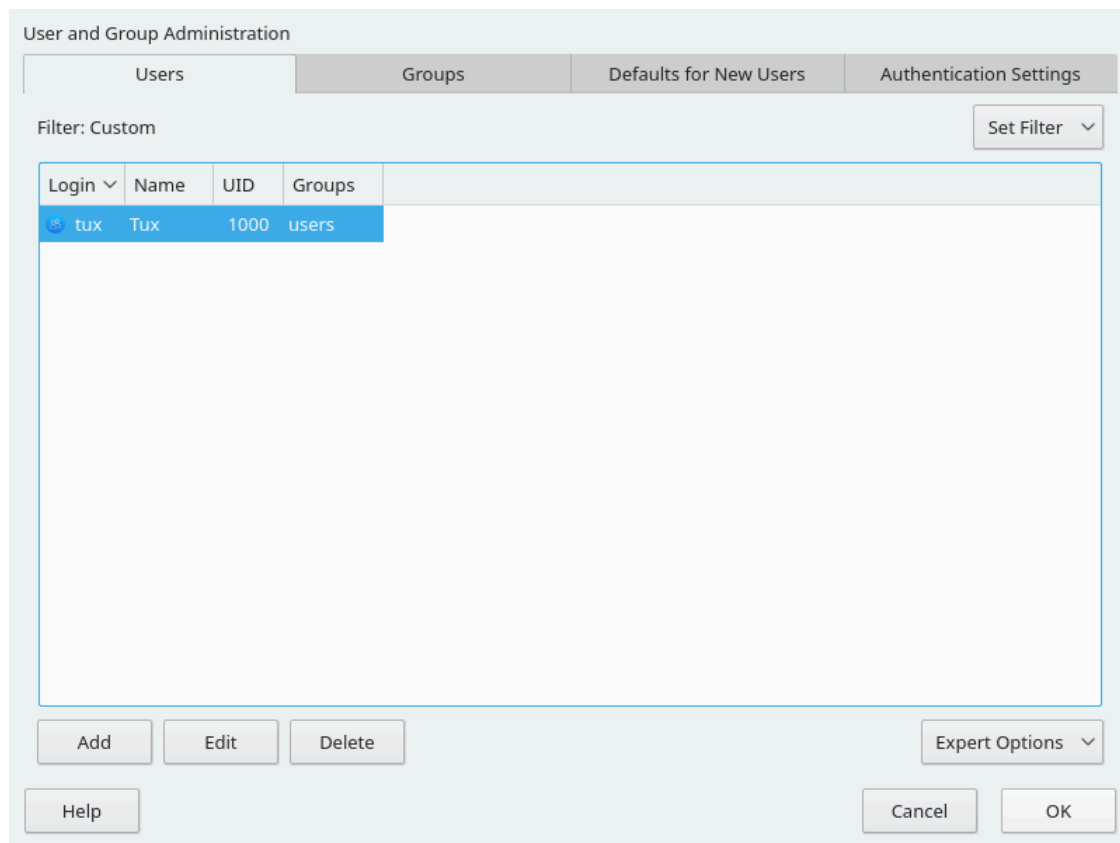


FIGURE 5.1: YAST USER AND GROUP ADMINISTRATION

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 11 "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

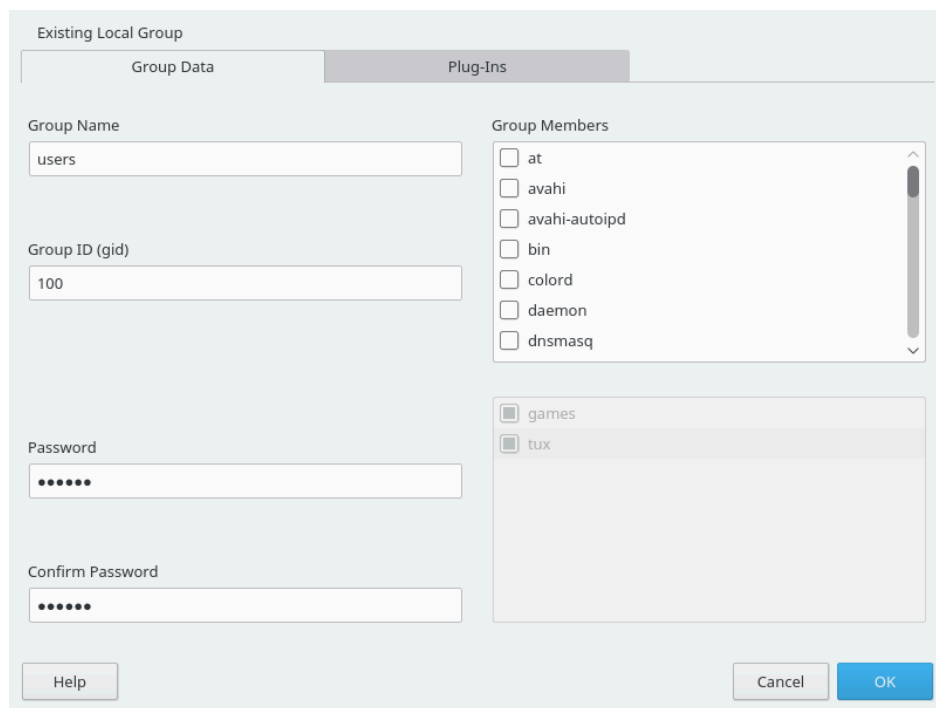
1. Open the YaST *User and Group Administration* dialog and click the *Users* tab. It lists users and the groups the users belong to.
2. Click *Edit* and switch to the *Details* tab.
3. To change the primary group the user belongs to, click *Default Group* and select the group from the list.
4. To assign 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 Clients Using YaST", Section 4.2 "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 Clients Using YaST", Section 4.1 "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 Clients Using YaST", Section 4.2 "SSSD"*

4. After accepting the configuration, return to the *User and Group Administration* overview.
5. Click *OK* to close the administration dialog.

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

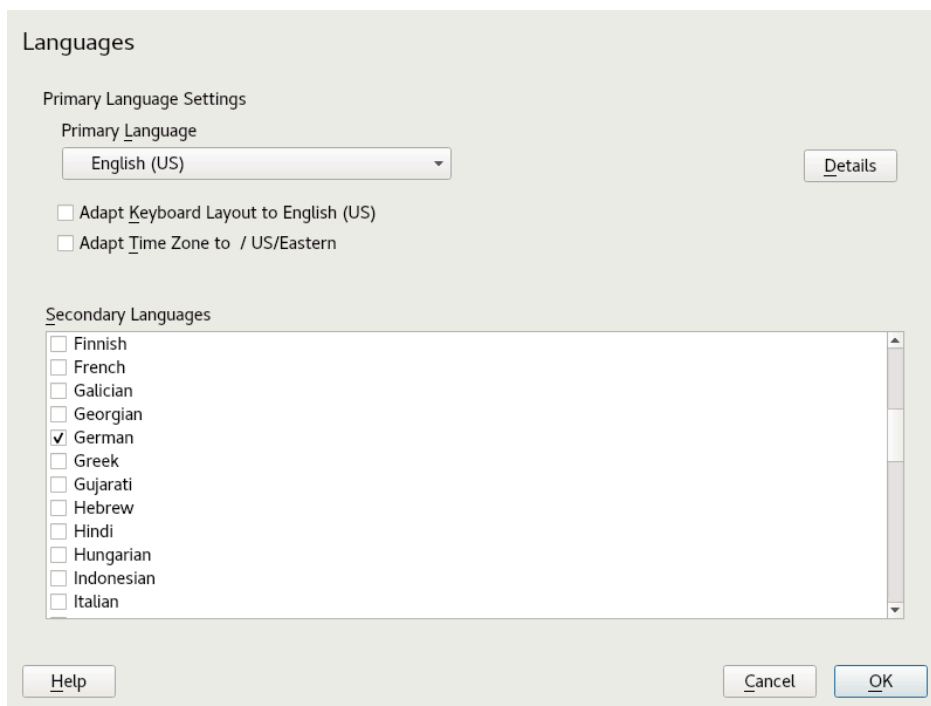
The primary language set in YaST applies to the entire system, including YaST and the desktop environment. This language is used whenever available unless you manually specify another language.

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.

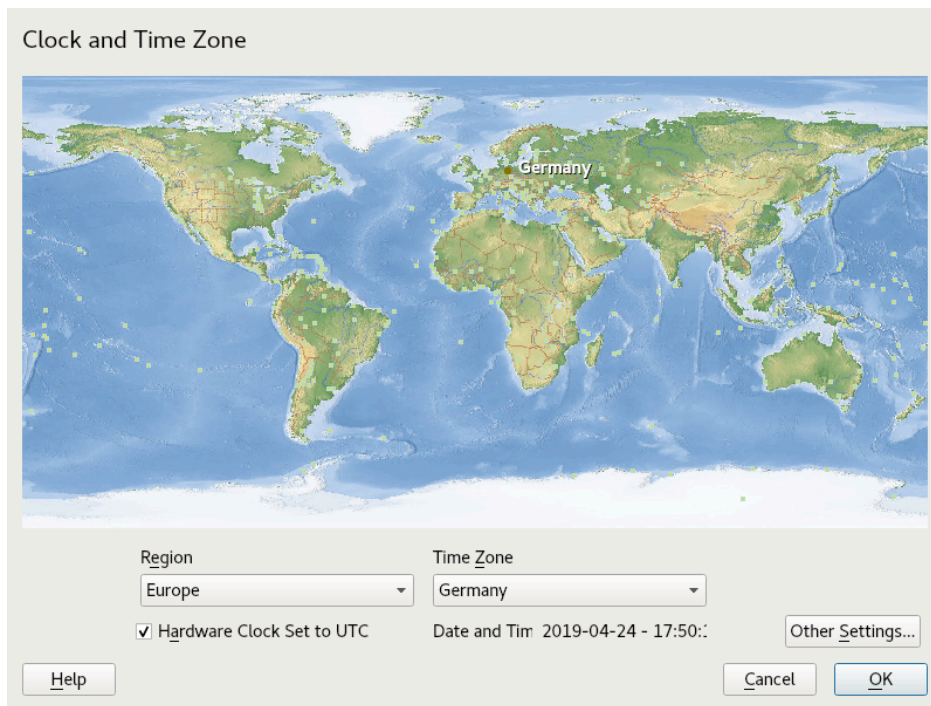
Start the application from the command line by using the following command:

```
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.

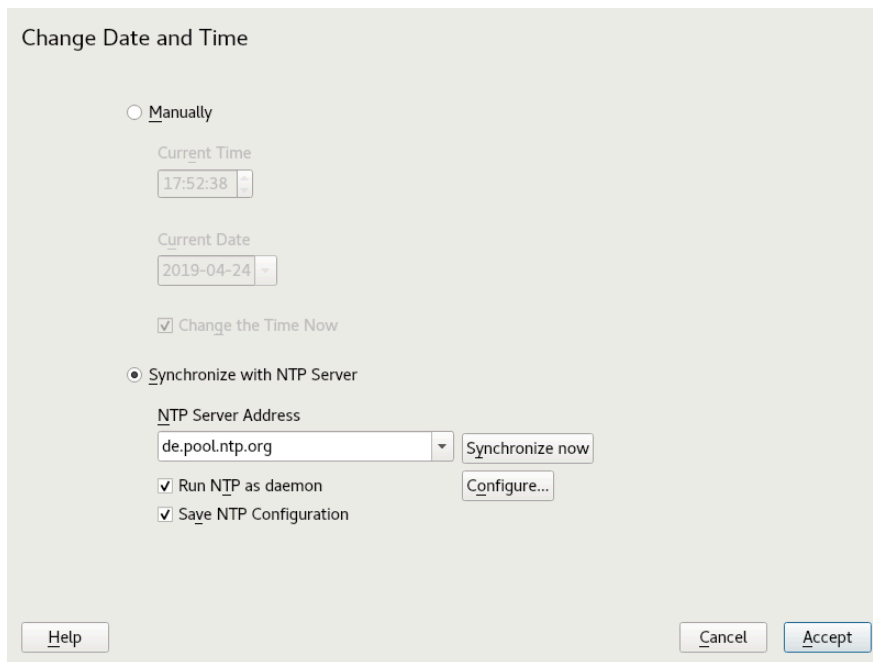
You can change the date and time manually or opt for synchronizing your machine against an NTP server, either permanently or only for adjusting your hardware clock.

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

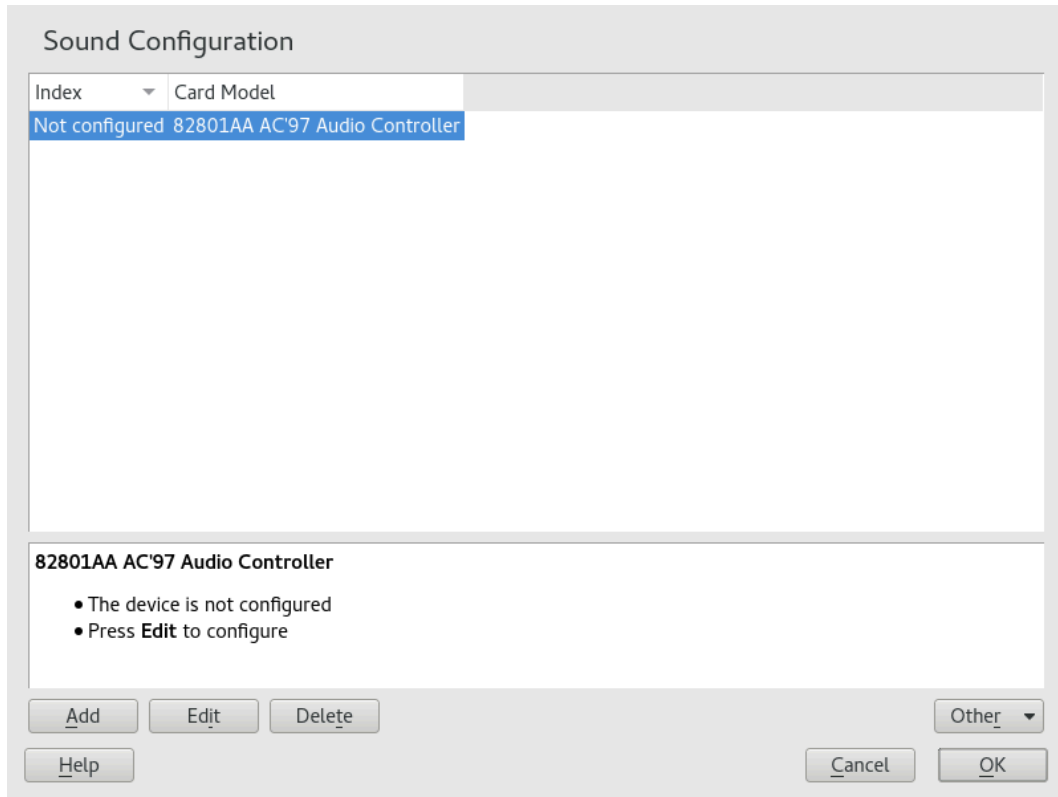
The YaST *System Keyboard Layout* module lets you define the default keyboard layout for the system (also used for the console). Users can modify the keyboard layout in their individual X sessions, using the desktop's tools.

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. If the sound module is not available, install it using the `sudo zypper install yast2-sound` command.



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/SD-B: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.openprinting.org/printers> ↗

The OpenPrinting home page with the printer database. The database shows the latest Linux support status. However, a Linux distribution can only integrate the drivers available at production time. Accordingly, a printer currently rated as “perfectly supported” may not have had this status when the latest 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 18 "Masquerading and Firewalls", Section 18.4 "firewalld"* and at http://en.opensuse.org/SD-B: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 is 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 <https://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 `fuusermount -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

<code>curlftpfs</code>	mount FTP servers
<code>encfs</code>	mount encrypted file systems
<code>fuseiso</code>	mounts CD-ROM images with ISO9660 file systems in them
<code>fusepod</code>	mount iPods
<code>fusesmb</code>	mount browseable Samba clients or Windows shares
<code>gphotofs</code>	mount supported digital cameras through gPhoto
<code>ntfs-3g</code>	mount NTFS volumes (with read and write support)
<code>obexfs</code>	mount Bluetooth devices
<code>sshfs</code>	file system client based on SSH file transfer protocol
<code>wdfs</code>	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 **119**
- 11 Installing Add-On Products **136**
- 12 YaST Online Update **138**
- 13 Upgrading the System and System Changes **144**

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

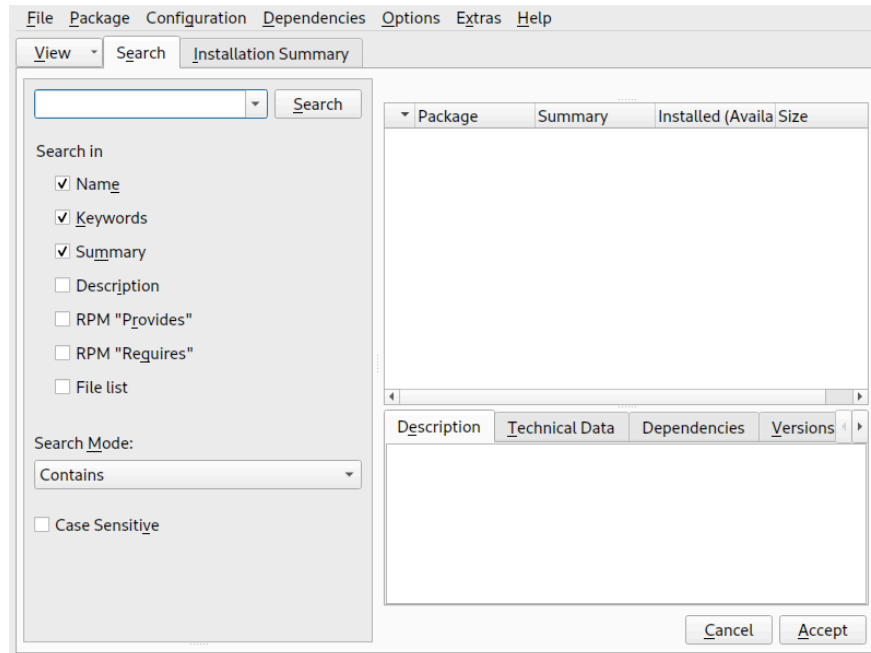
A delta RPM consists only of the binary diff between two defined versions of a package, and therefore has the smallest download size. Before being installed, the full RPM package is rebuilt on the local machine.

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 Searching Software

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

IEWS FOR SEARCHING PACKAGES OR PATTERNS

Patterns

Lists all patterns available for installation on your system.

Package Groups

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

Languages

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

Repositories

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

Services

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

Search

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

Installation Summary

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



Tip: Finding Packages Not Belonging to an Active Repository

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

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, "Searching Software"](#).
2. The packages found are listed in the right pane. To install a package or remove it, right-click it and choose *Install* or *Delete*. If the relevant option is not available, check the package status indicated by the symbol in front of the package name—press `Shift-F1` for help.



Tip: Applying an Action to All Packages Listed

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

3. To install a pattern, right-click the pattern name and choose *Install*.
4. It is not possible to remove a pattern. Instead, select the packages 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 2.1.3.5 "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, "Searching Software"](#).
 - b. On the right hand side of the window, click *Switch system packages to the versions in this repository*. This explicitly allows YaST to change the package vendor when replacing the packages.

When you proceed with *Accept*, all installed packages 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.

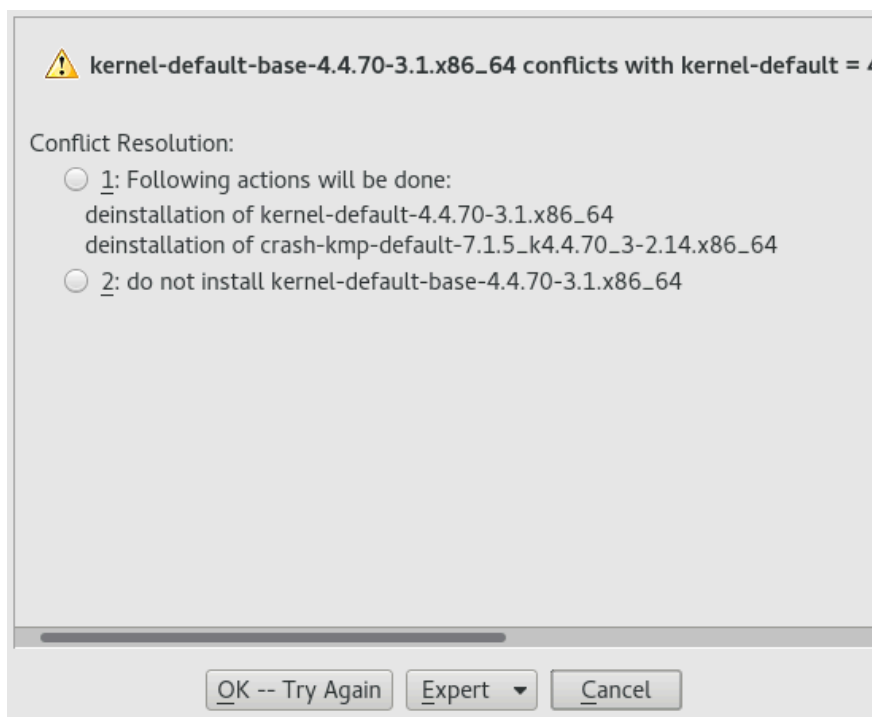


FIGURE 10.1: CONFLICT MANAGEMENT OF THE SOFTWARE MANAGER

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 15.2 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, a USB flash drive, a local directory, an ISO image or a network source.

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

1. Click *Add*.

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

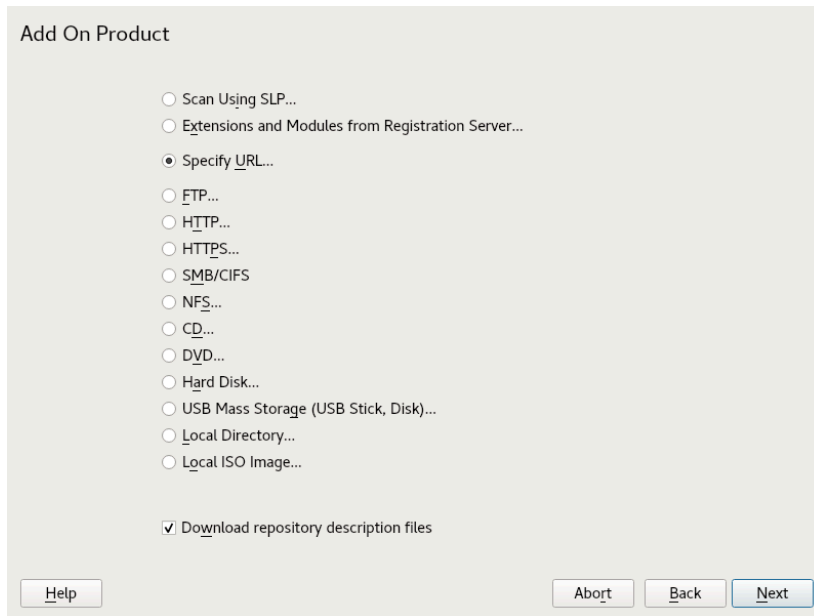


FIGURE 10.2: ADDING A SOFTWARE REPOSITORY

- 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.

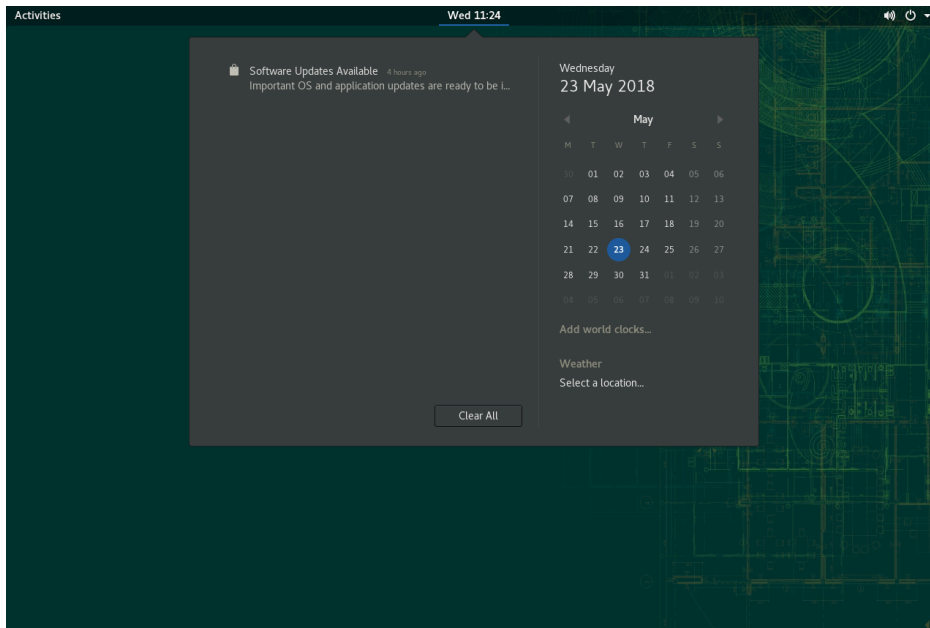
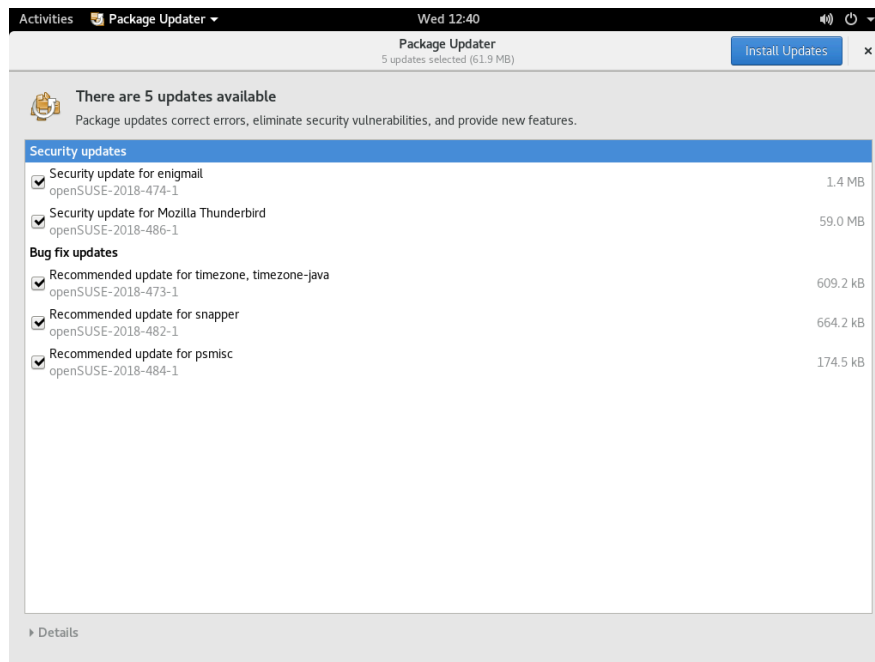


FIGURE 10.3: UPDATE NOTIFICATION ON GNOME DESKTOP

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

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.

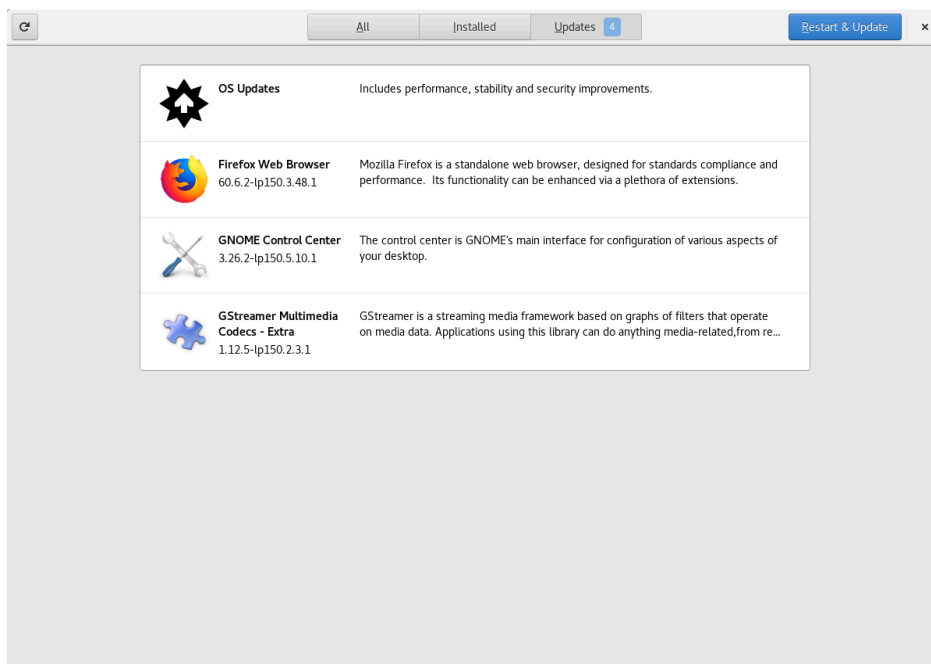


FIGURE 10.4: GNOME SOFTWARE—UPDATES VIEW

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 medium, 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.

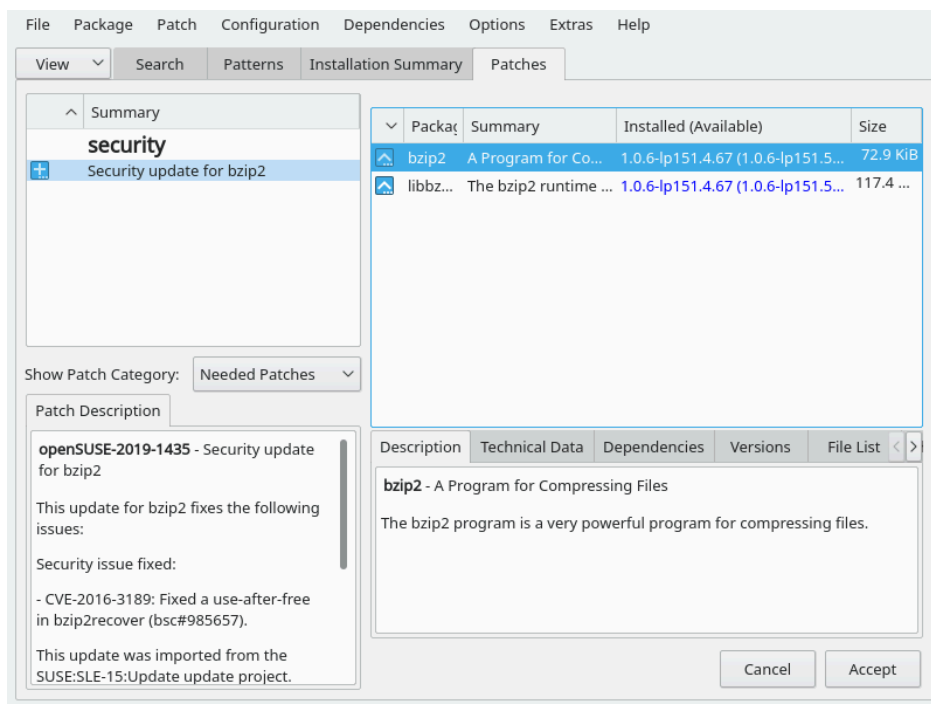


FIGURE 12.1: YAST ONLINE UPDATE

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 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

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

By default, updates are downloaded as delta RPMs. Since rebuilding RPM packages from delta RPMs is a memory- and processor-intensive task, certain setups or hardware configurations 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 skipping patches that require user interaction.

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

PROCEDURE 12.2: CONFIGURING THE AUTOMATIC ONLINE UPDATE

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

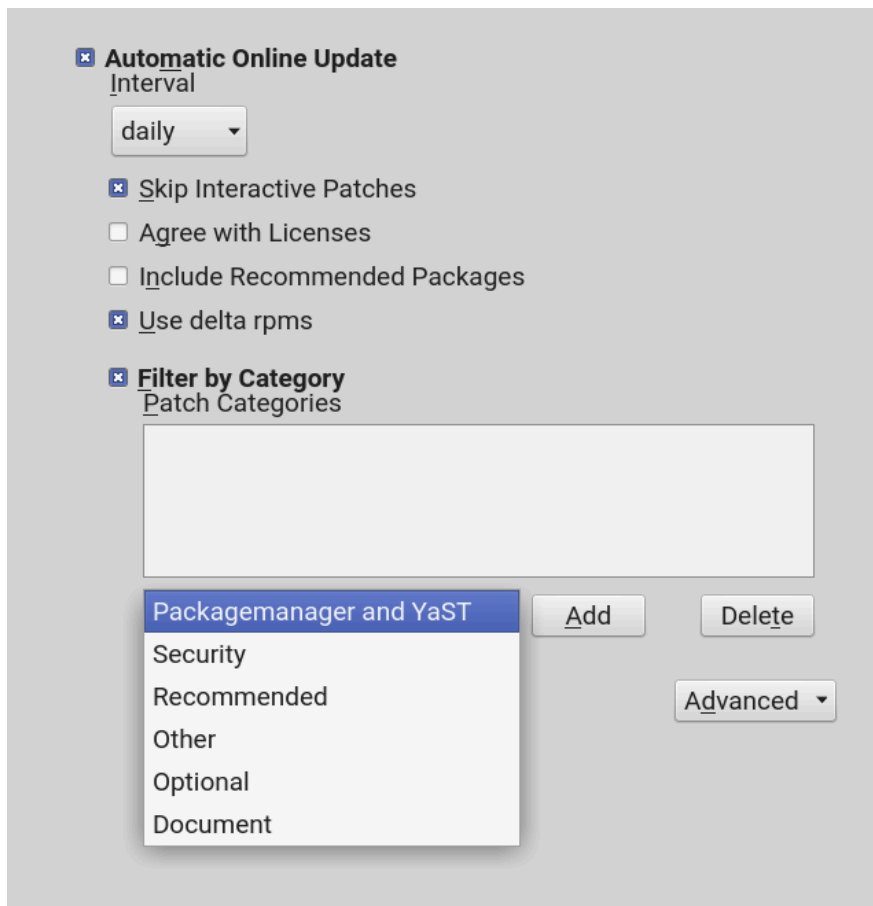


FIGURE 12.2: YAST ONLINE UPDATE CONFIGURATION

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

2. Choose the update interval: *Daily*, *Weekly*, or *Monthly*.
3. Sometimes patches may require the attention of the administrator, for example when restarting critical services. For example, this might be an update for Docker Open Source Engine that requires all containers to be restarted. Before these patches are installed, the user is informed about the consequences and is asked to confirm the installation of the patch. Such patches are called “Interactive Patches”.

When installing patches automatically, it is assumed that you have accepted the installation of interactive patches. If you prefer to review these patches before they get installed, check *Skip Interactive Patches*. In this case, interactive patches will be skipped during automated patching. Make sure to periodically run a manual online update, to check whether interactive patches are waiting to be installed.

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

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

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 15.2 is only available as 64-bit version

openSUSE Leap 15.2 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 \(https://en.opensuse.org/Portal:Tumbleweed\)](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 \(https://doc.opensuse.org/release-notes/\)](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
<code>/dev/sda3</code>	74G	22G	53G	29%	<code>/</code>
<code>udev</code>	252M	124K	252M	1%	<code>/dev</code>
<code>/dev/sda5</code>	116G	5.8G	111G	5%	<code>/home</code>
<code>/dev/sda1</code>	39G	1.6G	37G	4%	<code>/windows/C</code>
<code>/dev/sda2</code>	4.6G	2.6G	2.1G	57%	<code>/windows/D</code>

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

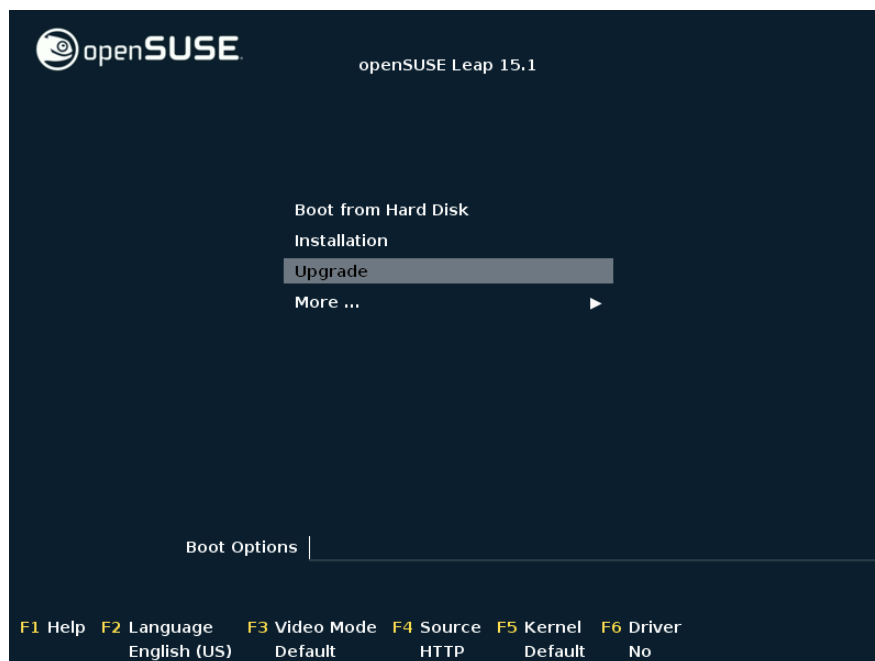
Before updating PostgreSQL (`postgres`), dump the databases. See the manual page of `pg_dump`. This is only necessary if you actually used PostgreSQL prior to your update.

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.

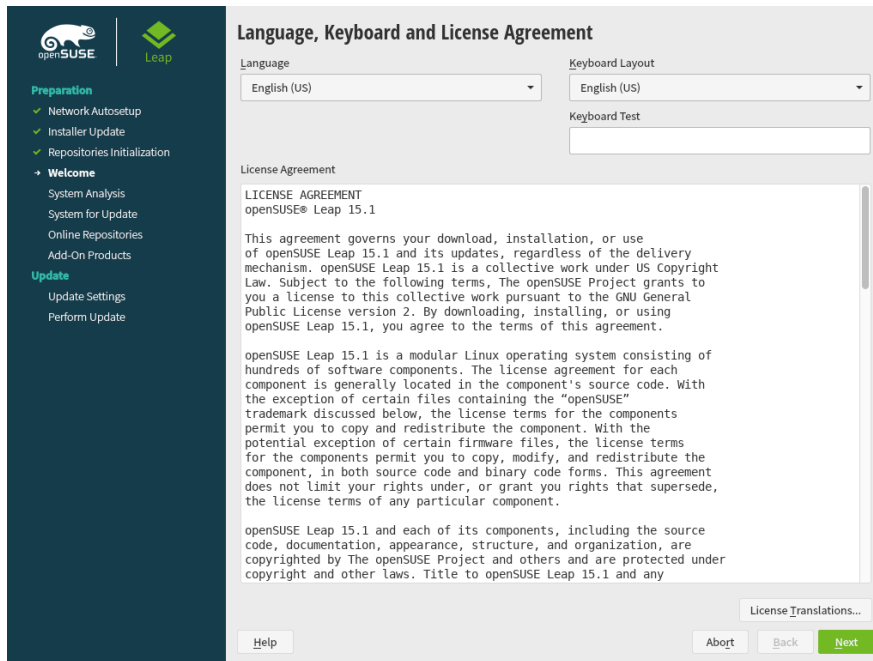
Use `F2` to change the language for the installer. A corresponding keyboard layout is chosen automatically. See [Section 2.2.1, "The Boot Screen on Machines Equipped with Traditional BIOS"](#) or [Section 2.2.2, "The Boot Screen on Machines Equipped with UEFI"](#) for more information about changing boot parameters.



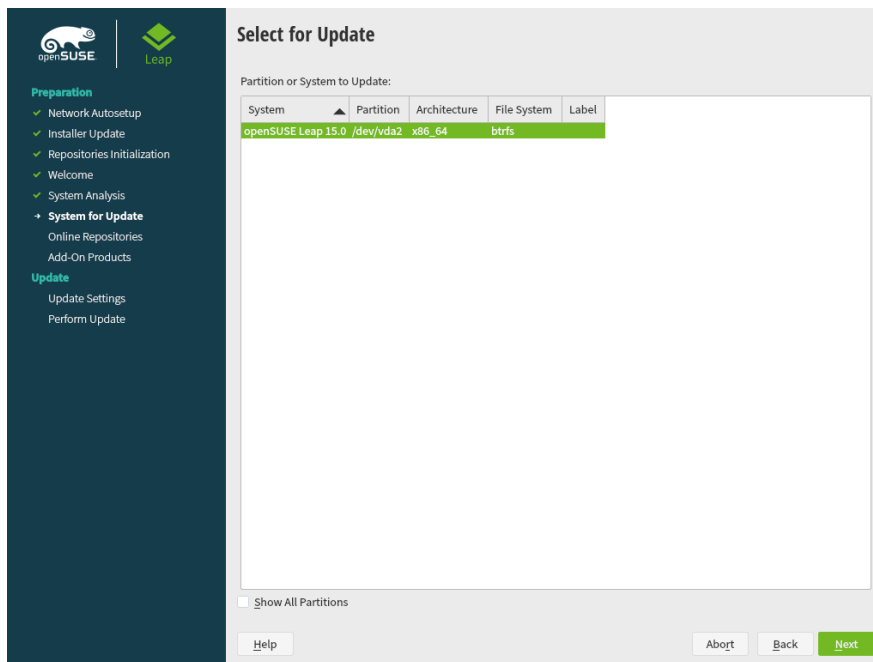
2. Select *Upgrade* on the boot screen, then press `Enter`. This boots the system and loads the openSUSE Leap installer. Do not select *Installation*.

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

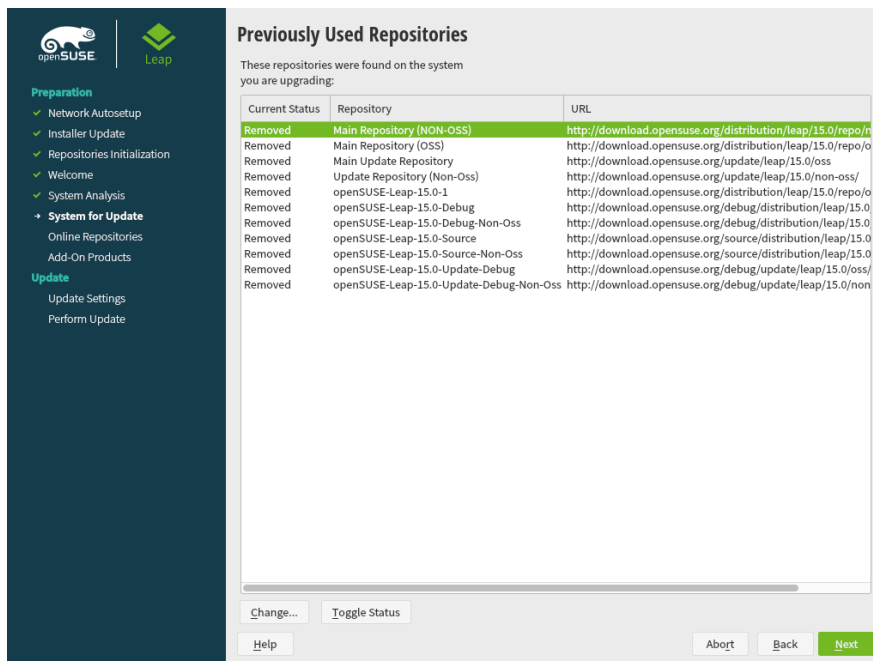
From this point on, the Release Notes can be viewed from any screen during the installation process by selecting *Release Notes*.

- 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.

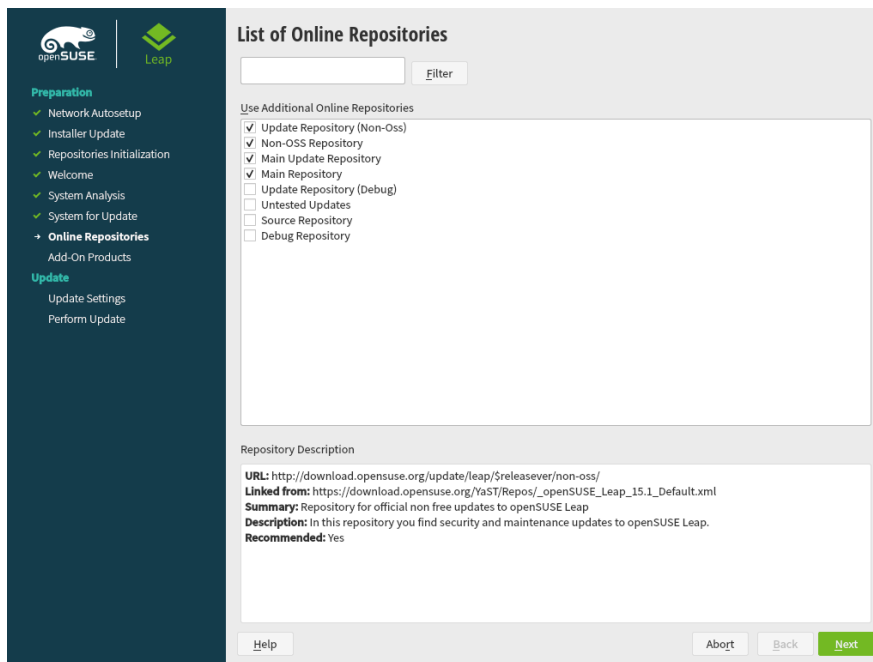
If you have had added custom repositories, for example from the openSUSE Build Service, you have two choices:

- 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 3.11.7, “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

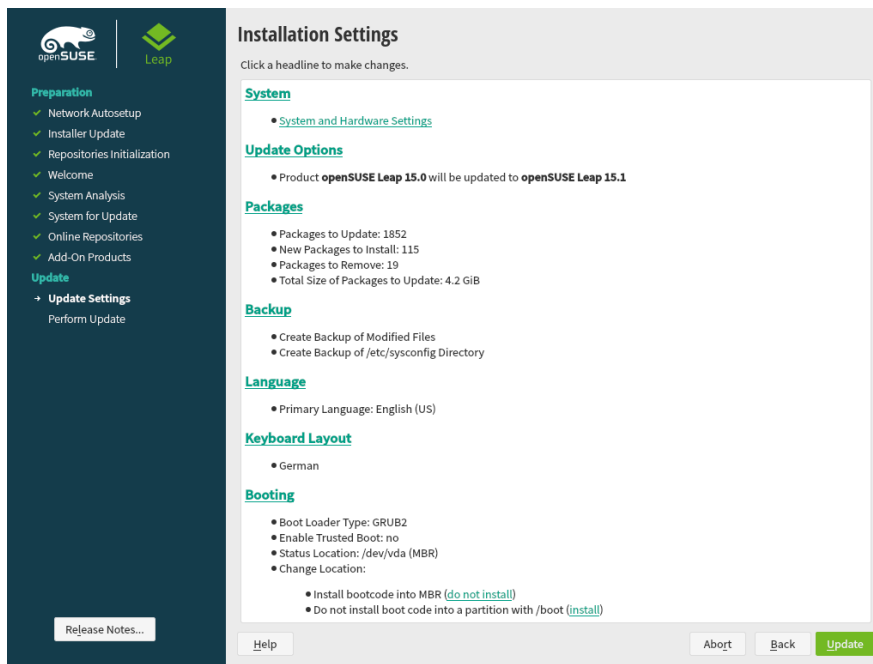
This section allows you to change the *Primary Language* primary language and configure additional *Secondary Languages*. Optionally, you can adjust the keyboard layout and timezone to the selected primary language.

Keyboard Layout

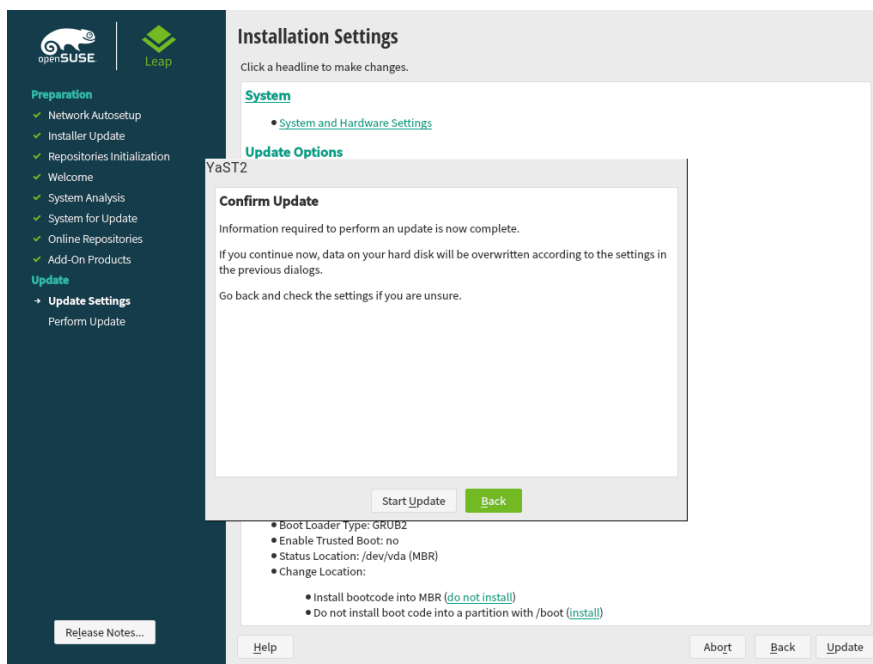
Here you can change the keyboard layout and adjust additional *Expert Keyboard Settings*.

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 openSUSE_Leap_15.1 to openSUSE_Leap_15.2. Add the new repositories with commands such as:

```
server=http://download.example.org  
tux > sudo zypper ar $server/distribution/leap/15.2/repo/oss/ Leap-15.2-0SS  
tux > sudo zypper ar $server/update/leap/15.2/oss/ Leap-15.2-Update
```

And remove the old repositories:

```
zypper rr Leap-15.1-0SS
```

```
zypper rr Leap-15.1-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 --updates-tack-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 **158**
- 15 Bash and Bash Scripts **196**

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:

```
1 2 3
tux@linux:~>
```

- 1 Your login.
- 2 The hostname of your computer.
- 3 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-x ① 1 ② tux ③ users ④ 48 ⑤ 2006-06-23 16:08 ⑥ bin ⑦
```

- ① Type of object and access permissions. For further information, refer to [Section 14.6.1, "Permissions for User, Group and Others"](#).
- ② Number of hard links to this file.
- ③ Owner of the file or directory. For further information, refer to [Section 14.6.1, "Permissions for User, Group and Others"](#).
- ④ Group assigned to the file or directory. For further information, refer to [Section 14.6.1, "Permissions for User, Group and Others"](#).
- ⑤ File size in bytes.
- ⑥ Date and time of the last change.
- ⑦ 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 roots 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 11 "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
<u>-</u>	<u>rw-</u>	<u>r--</u>	<u>r--</u>

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 <code>ls -l</code> , 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 system	Users can change into the directory and execute files there. If they do not have read access to that directory they cannot list the files

Access Permission	File	Folder
	<p>can execute the file directly, users do not need read permission to execute the file. However, if the file must be interpreted like a shell script or a perl program, additional read permission is needed.</p>	<p>but can access them nevertheless if they know of their existence.</p>

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
```

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
<u>?</u>	Matches exactly one arbitrary character
<u>*</u>	Matches any number of characters
<u>[SET]</u>	Matches one of the characters from the group specified inside the square brackets, which is represented here by the string <u>SET</u> .

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 `my-file1.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. Compress 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* Linux shell is Bash (Bourne again Shell). When this chapter speaks about “the shell” it means Bash. There are more shells available (ash, csh, ksh, zsh, ...), each employing different features and characteristics. If you need further information about other shells, search for *shell* in YaST.

15.1.1 Bash Configuration Files

A shell can be invoked as an:

1. **Interactive login shell.** This is used when logging in to a machine, invoking Bash with the `--login` option or when logging in to a remote machine with SSH.
2. **“Ordinary” interactive shell.** This is normally the case when starting xterm, konsole, gnome-terminal, or similar command-line interface (CLI) tools.
3. **Non-interactive shell.** This is invoked 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
<u>/etc/profile</u>	Do not modify this file, otherwise your modifications may be destroyed during your next update!

File	Description
<u>/etc/profile.local</u>	Use this file if you extend <u>/etc/profile</u>
<u>/etc/profile.d/</u>	Contains system-wide configuration files for specific programs
<u>~/.profile</u>	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

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

Additionally, Bash uses some more files:

TABLE 15.3: SPECIAL FILES FOR BASH

File	Description
<u>~/.bash_history</u>	Contains a list of all commands you have typed
<u>~/.bash_logout</u>	Executed when logging out
<u>~/.alias</u>	User defined aliases of frequently used commands. See <u>man 1 alias</u> for more details about defining aliases.

No-Login Shells

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

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

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

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

```

statd          /sbin/nologin
svn            /sbin/nologin
systemd-coredump /sbin/nologin
systemd-network /sbin/nologin
systemd-timesync /sbin/nologin
usbmux         /sbin/nologin
vnc            /sbin/nologin
wwwrun         /sbin/nologin
messagebus     /usr/bin/false
scard          /usr/sbin/nologin

```

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
<u>/</u>	Root directory—the starting point of the directory tree.
<u>/bin</u>	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.
<u>/boot</u>	Static files of the boot loader.
<u>/dev</u>	Files needed to access host-specific devices.
<u>/etc</u>	Host-specific system configuration files.
<u>/home</u>	Holds the home directories of all users who have accounts on the system. However, <u>root</u> 's home directory is not located in <u>/home</u> but in <u>/root</u> .
<u>/lib</u>	Essential shared libraries and kernel modules.
<u>/media</u>	Mount points for removable media.
<u>/mnt</u>	Mount point for temporarily mounting a file system.

Directory	Contents
<u>/opt</u>	Add-on application software packages.
<u>/root</u>	Home directory for the superuser <u>root</u> .
<u>/sbin</u>	Essential system binaries.
<u>/srv</u>	Data for services provided by the system.
<u>/tmp</u>	Temporary files.
<u>/usr</u>	Secondary hierarchy with read-only data.
<u>/var</u>	Variable data such as log files.
<u>/windows</u>	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/PACKAGE-NAME 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 your shell scripts:

TABLE 15.5: USEFUL ENVIRONMENT VARIABLES

<u>HOME</u>	the home directory of the current user
<u>HOST</u>	the current host name
<u>LANG</u>	when a tool is localized, it uses the language from this environment variable. English can also be set to <u>C</u>
<u>PATH</u>	the search path of the shell, a list of directories separated by colon

<u>PS1</u>	specifies the normal prompt printed before each command
<u>PS2</u>	specifies the secondary prompt printed when you execute a multi-line command
<u>PWD</u>	current working directory
<u>USER</u>	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 **212**
- 17 Common Problems and Their Solutions **217**

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 <code>/dev</code>)
5	File formats and conventions (<code>/etc/fstab</code>)
6	Games
7	Miscellaneous (including macro packages and conventions), for example, <code>man(7)</code> , <code>groff(7)</code>
8	System administration commands (usually only for <code>root</code>)
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 <https://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 <https://forums.opensuse.org/> for a list.

GNOME Documentation

Documentation for GNOME users, administrators and developers is available at <https://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 <https://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
<u><code>~/.xsession-errors</code></u>	Messages from the desktop applications currently running.
<u><code>/var/log/apparmor/</code></u>	Log files from AppArmor, see <i>Book "Security Guide"</i> for detailed information.
<u><code>/var/log/audit/audit.log</code></u>	Log file from Audit to track any access to files, directories, or resources of your system, and trace system calls. See <i>Book "Security Guide"</i> for detailed information.
<u><code>/var/log/mail.*</code></u>	Messages from the mail system.
<u><code>/var/log/NetworkManager</code></u>	Log file from NetworkManager to collect problems with network connectivity

Log File	Description
<u>/var/log/samba/</u>	Directory containing Samba server and client log messages.
<u>/var/log/warn</u>	All messages from the kernel and system log daemon with the “warning” level or higher.
<u>/var/log/wtmp</u>	Binary file containing user login records for the current machine session. View it with <u>last</u> .
<u>/var/log/Xorg.*.log</u>	Various start-up and runtime log files from the X Window System. It is useful for debugging failed X start-ups.
<u>/var/log/YaST2/</u>	Directory containing YaST's actions and their results.
<u>/var/log/zypper.log</u>	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
<u>/proc/cpuinfo</u>	Contains processor information, including its type, make, model, and performance.
<u>/proc/dma</u>	Shows which DMA channels are currently being used.
<u>/proc/interrupts</u>	Shows which interrupts are in use, and how many of each have been in use.
<u>/proc/iomem</u>	Displays the status of I/O (input/output) memory.

File	Description
<u>/proc/ioports</u>	Shows which I/O ports are in use at the moment.
<u>/proc/meminfo</u>	Displays memory status.
<u>/proc/modules</u>	Displays the individual modules.
<u>/proc/mounts</u>	Displays devices currently mounted.
<u>/proc/partitions</u>	Shows the partitioning of all hard disks.
<u>/proc/version</u>	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
<u>/sys/block</u>	Contains subdirectories for each block device discovered in the system. Generally, these are mostly disk type devices.
<u>/sys/bus</u>	Contains subdirectories for each physical bus type.
<u>/sys/class</u>	Contains subdirectories grouped together as a functional types of devices (like graphics, net, printer, etc.)
<u>/sys/device</u>	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.

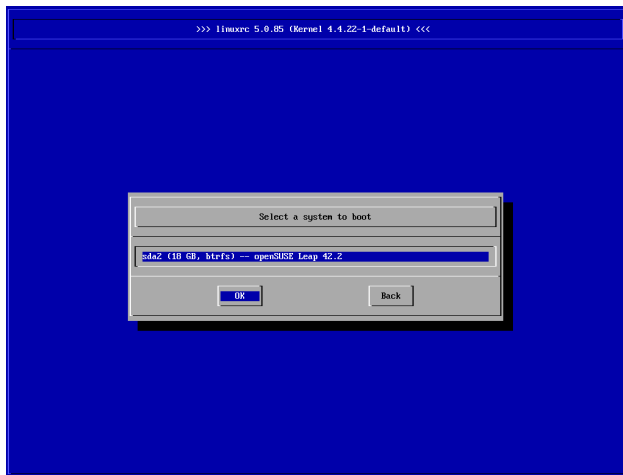


FIGURE 17.1: SELECT DISK

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.

17.2.6 Disable Swap to Enable Booting

When a swap device is not available and the system cannot enable it during boot, booting may fail. Try disabling all swap devices by appending the following options to the kernel command line:

```
systemd.device_wants_unit=off systemd.mask=swap.target
```

You may also try disabling specific swap devices:

```
systemd.mask=dev-sda1.swap
```

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, "GNOME Desktop Has Issues"*.
4. If the user's home directory has been used with another Linux distribution, remove the `Xauthority` file in the user's home. Use a console login via `Ctrl-Alt-F1` and run `rm .Xauthority` as this user. This should eliminate X authentication problems for this user. Try graphical login again.
5. If the desktop could not start because of corrupt configuration files, proceed with *Section 17.3.4, "GNOME Desktop Has Issues"*.

In the following, common reasons a network authentication for a particular user may fail on a specific machine are listed:

- The user may have entered the wrong password.
- The user name exists in the machine's local authentication files and is also provided by a network authentication system, causing conflicts.

- The home directory exists but is corrupt or unavailable. Perhaps it is write protected or is on a server that is inaccessible at the moment.
- The user does not have permission to log in to that particular host in the authentication system.
- The machine has changed host names, for whatever reason, and the user does not have permission to log in to that host.
- The machine cannot reach the authentication server or directory server that contains that user's information.
- There may be problems with the X Window System authenticating this particular user, especially if the user's home has been used with another Linux distribution 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, "GNOME Desktop Has Issues"](#).
6. If the user's home directory has been used with another Linux distribution, remove the `.Xauthority` file in the user's home. Use a console login via `Ctrl-Alt-F1` and run `rm .Xauthority` as this user. This should eliminate X authentication problems for this user. Try graphical login again.

7. If the desktop could not start because of corrupt configuration files, proceed with [Section 17.3.4, “GNOME Desktop Has Issues”](#).

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 GNOME Desktop Has Issues

If you are experiencing issues with the GNOME desktop, there are several ways to troubleshoot the misbehaving graphical desktop environment. The recommended procedure described below offers the safest option to fix a broken GNOME desktop.

PROCEDURE 17.1: TROUBLESHOOTING GNOME

1. Launch YaST and switch to *Security and Users*.
2. Open the *User and Group Management* dialog and click *Add*.
3. Fill out the required fields and click *OK* to create a new user.
4. Log out and log in as the new user. This gives you a fresh GNOME environment.

5. Copy individual subdirectories from the `~/.local/` and `~/.config/` directories of the old user account to the respective directories of the new user account.
Log out and log in again as the new user after every copy operation to check whether GNOME still works correctly.
6. Repeat the previous step until you find the configuration file that breaks GNOME.
7. Log in as the old user, and move the offending configuration file to a different location.
Log out and log in again as the old user.
8. Delete the previously created user.

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 17.2: HOW TO IDENTIFY NETWORK PROBLEMS

When checking the network connection of your machine, proceed as follows:

1. If you use an Ethernet connection, check the hardware first. Make sure that your network cable is properly plugged into your computer and router (or hub, etc.). The control lights next to your Ethernet connector are normally both be active.
If the connection fails, check whether your network cable works with another machine. If it does, your network card causes the failure. If hubs or switches are included in your network setup, they may be faulty, as well.
2. If using a wireless connection, check whether the wireless link can be established by other machines. If not, contact the wireless network's administrator.
3. When you have checked your basic network connectivity, try to find out which service is not responding. Gather the address information of all network servers needed in your setup. Either look them up in the appropriate YaST module or ask your system administrator. The following list gives 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 17.2, "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 17.3: BACKING UP AND RESTORING HARD DISKS

1. Start a Shell as user `root`.
2. Select your source device. Typically this is something like `/dev/sda` (labeled as `SOURCE`).
3. Decide where you want to store your image (labeled as `BACKUP_PATH`). It must be different from your source device. In other words: if you make a backup from `/dev/sda`, your image file must not to be stored under `/dev/sda`.
4. Run the commands to create a compressed image file:

```
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

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary

formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.  
Permission is granted to copy, distribute  
and/or modify this document  
under the terms of the GNU Free  
Documentation License, Version 1.2  
or any later version published by the Free  
Software Foundation;  
with no Invariant Sections, no Front-Cover  
Texts, and no Back-Cover Texts.  
A copy of the license is included in the  
section entitled "GNU  
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being LIST  
THEIR TITLES, with the  
Front-Cover Texts being LIST, and with the  
Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.



Security Guide

openSUSE Leap 15.2



Security Guide

openSUSE Leap 15.2

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: July 06, 2020

SUSE LLC

1800 South Novell Place

Provo, UT 84606

USA

<https://documentation.suse.com> ↗

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

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

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

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

Contents

About This Guide xv

1	Security and Confidentiality	1		
1.1	Overview	1		
1.2	Passwords	2		
1.3	System Integrity	2		
1.4	File Access	3		
1.5	Networking	3		
1.6	Software Vulnerabilities	4		
1.7	Malware	5		
1.8	Important Security Tips	6		
1.9	Reporting Security Issues	6		
I	AUTHENTICATION	7		
2	Authentication with PAM	8		
2.1	What is PAM?	8		
2.2	Structure of a PAM Configuration File	9		
2.3	The PAM Configuration of sshd	11		
2.4	Configuration of PAM Modules	14		
	pam_env.conf	14 • pam_mount.conf.xml	15 • limits.conf	15
2.5	Configuring PAM Using pam-config	15		
2.6	Manually Configuring PAM	16		
2.7	For More Information	17		

3 Using NIS 18

- 3.1 Configuring NIS Servers 18
 - Configuring a NIS Master Server 18 • Configuring a NIS Slave Server 23
- 3.2 Configuring NIS Clients 24

4 Setting Up Authentication Clients Using YaST 26

- 4.1 Configuring an Authentication Client with YaST 26
- 4.2 SSSD 26
 - Checking the Status 27 • Caching 27

5 LDAP—A Directory Service 28

- 5.1 Structure of an LDAP Directory Tree 28
- 5.2 Installing the Software for 389 Directory Server 31
- 5.3 Manually Configuring a 389 Directory Server 31
 - Creating the 389 Directory Server Instance 32 • Using CA Certificates for TSL 33 • Configuring Admin Credentials for Remote/Local Access 34 • Configuring LDAP Users and Groups 35 • Setting Up SSSD 38
- 5.4 Setting Up a 389 Directory Server with YaST 39
 - Creating a 389 Directory Server Instance with YaST 39 • Configuring an LDAP Client with YaST 40
- 5.5 Manually Administering LDAP Data 43
- 5.6 For More Information 43

6 Network Authentication with Kerberos 44

- 6.1 Conceptual Overview 44
- 6.2 Kerberos Terminology 44
- 6.3 How Kerberos Works 46
 - First Contact 46 • Requesting a Service 47 • Mutual Authentication 48 • Ticket Granting—Contacting All Servers 48

- 6.4 User View of Kerberos 49
- 6.5 Installing and Administering Kerberos 50
 - Kerberos Network Topology 51 • Choosing the Kerberos Realms 52 • Setting Up the KDC Hardware 52 • Configuring Time Synchronization 53 • Configuring the KDC 54 • Configuring Kerberos Clients 58 • Configuring Remote Kerberos Administration 60 • Creating Kerberos Service Principals 62 • Enabling PAM Support for Kerberos 64 • Configuring SSH for Kerberos Authentication 64 • Using LDAP and Kerberos 65
- 6.6 Setting up Kerberos using *LDAP and Kerberos Client* 68
- 6.7 Kerberos and NFS 72
 - Group Membership 73 • Performance and Scalability 74 • Master KDC, Multiple Domains, and Trust Relationships 75
- 6.8 For More Information 76
- 7 Active Directory Support 77**
- 7.1 Integrating Linux and Active Directory Environments 77
- 7.2 Background Information for Linux Active Directory Support 78
 - Domain Join 80 • Domain Login and User Homes 81 • Offline Service and Policy Support 82
- 7.3 Configuring a Linux Client for Active Directory 83
 - Choosing Which YaST Module to Use for Connecting to Active Directory 84 • Joining Active Directory Using *User Logon Management* 84 • Joining Active Directory Using *Windows Domain Membership* 89 • Checking Active Directory Connection Status 91
- 7.4 Logging In to an Active Directory Domain 92
 - GDM 92 • Console Login 92
- 7.5 Changing Passwords 93

II	LOCAL SECURITY	95
8	Spectre/Meltdown Checker	96
8.1	Using spectre-meltdown-checker	96
8.2	Additional Information about Spectre/Meltdown	98
9	Configuring Security Settings with YaST	99
9.1	<i>Security Overview</i>	99
9.2	<i>Predefined Security Configurations</i>	100
9.3	<i>Password Settings</i>	101
9.4	<i>Boot Settings</i>	102
9.5	<i>Login Settings</i>	102
9.6	<i>User Addition</i>	102
9.7	<i>Miscellaneous Settings</i>	102
10	Authorization with PolKit	104
10.1	Conceptual Overview	104
	Available Authentication Agents	104
	Structure of PolKit	104
	Available Commands	105
	Available Policies and Supported Applications	105
10.2	Authorization Types	107
	Implicit Privileges	107
	Explicit Privileges	108
	Default Privileges	108
10.3	Querying Privileges	108
10.4	Modifying Configuration Files	109
	Adding Action Rules	109
	Adding Authorization Rules	110
	Modifying Configuration Files for Implicit Privileges	111
10.5	Restoring the Default Privileges	112
11	Access Control Lists in Linux	114
11.1	Traditional File Permissions	114
	The setuid Bit	115
	The setgid Bit	115
	The Sticky Bit	116

- 11.2 Advantages of ACLs 116
- 11.3 Definitions 116
- 11.4 Handling ACLs 117
 - ACL Entries and File Mode Permission Bits 118 • A Directory with an ACL 119 • A Directory with a Default ACL 122 • The ACL Check Algorithm 124
- 11.5 ACL Support in Applications 125
- 11.6 For More Information 125
- 12 Encrypting Partitions and Files 126**
- 12.1 Setting Up an Encrypted File System with YaST 126
 - Creating an Encrypted Partition during Installation 127 • Creating an Encrypted Partition on a Running System 128 • Encrypting the Content of Removable Media 128
- 12.2 Encrypting Files with GPG 129
- 13 Storage Encryption for Hosted Applications with cryptctl 130**
- 13.1 Setting Up a **cryptctl** Server 131
- 13.2 Setting Up a **cryptctl** Client 133
- 13.3 Checking Partition Unlock Status Using Server-side Commands 136
- 13.4 Unlocking Encrypted Partitions Manually 137
- 13.5 Maintenance Downtime Procedure 137
- 13.6 For More Information 137
- 14 Certificate Store 138**
- 14.1 Activating Certificate Store 138
- 14.2 Importing Certificates 138

15 Intrusion Detection with AIDE 140

- 15.1 Why Use AIDE? 140
- 15.2 Setting Up an AIDE Database 140
- 15.3 Local AIDE Checks 143
- 15.4 System Independent Checking 144
- 15.5 For More Information 146

III NETWORK SECURITY 147

16 X Window System and X Authentication 148

17 SSH: Secure Network Operations 149

- 17.1 **ssh**—Secure Shell 149
 - Starting X Applications on a Remote Host 150 • Agent Forwarding 150
- 17.2 **scp**—Secure Copy 150
- 17.3 **sftp**—Secure File Transfer 151
 - Using **sftp** 151 • Setting Permissions for File Uploads 152
- 17.4 The SSH Daemon (**sshd**) 153
 - Maintaining SSH Keys 154 • Rotating Host Keys 154
- 17.5 SSH Authentication Mechanisms 155
 - Generating an SSH Key 156 • Copying an SSH Key 156 • Using the **ssh-agent** 157
- 17.6 Port Forwarding 158
- 17.7 Adding and Removing Public Keys on an Installed System 159
- 17.8 For More Information 159

18 Masquerading and Firewalls 161

- 18.1 Packet Filtering with iptables 161
- 18.2 Masquerading Basics 164

18.3	Firewalling Basics	165
18.4	<code>firewalld</code>	166
	Configuring the Firewall on the Command Line	167
	Accessing Services	
	Listening on Dynamic Ports	172
18.5	Migrating From SuSEfirewall2	175
18.6	For More Information	177
19	Configuring a VPN Server	178
19.1	Conceptual Overview	178
	Terminology	178
	VPN Scenarios	179
19.2	Setting Up a Simple Test Scenario	181
	Configuring the VPN Server	182
	Configuring the VPN	
	Clients	183
	Testing the VPN Example Scenario	184
19.3	Setting Up Your VPN Server Using a Certificate Authority	184
	Creating Certificates	185
	Configuring the VPN Server	186
	Configuring	
	the VPN Clients	188
19.4	Setting Up a VPN Server or Client Using YaST	189
19.5	For More Information	190
IV	CONFINING PRIVILEGES WITH APPARMOR	191
20	Introducing AppArmor	192
20.1	AppArmor Components	192
20.2	Background Information on AppArmor Profiling	192
21	Getting Started	194
21.1	Installing AppArmor	194
21.2	Enabling and Disabling AppArmor	195
21.3	Choosing Applications to Profile	196
21.4	Building and Modifying Profiles	196

21.5	Updating Your Profiles	198								
22	Immunizing Programs	199								
22.1	Introducing the AppArmor Framework	200								
22.2	Determining Programs to Immunize	202								
22.3	Immunizing cron Jobs	203								
22.4	Immunizing Network Applications	203								
	Immunizing Web Applications	205 • Immunizing Network Agents	207							
23	Profile Components and Syntax	208								
23.1	Breaking an AppArmor Profile into Its Parts	209								
23.2	Profile Types	211								
	Standard Profiles	211 • Unattached Profiles	212 • Local Profiles	212 • Hats	213 • Change rules	213				
23.3	Include Statements	214								
	Abstractions	216 • Program Chunks	216 • Tunables	216						
23.4	Capability Entries (POSIX.1e)	216								
23.5	Network Access Control	217								
23.6	Profile Names, Flags, Paths, and Globbing	218								
	Profile Flags	219 • Using Variables in Profiles	220 • Pattern Matching	221 • Namespaces	222 • Profile Naming and Attachment Specification	222 • Alias Rules	223			
23.7	File Permission Access Modes	223								
	Read Mode (r)	224 • Write Mode (w)	224 • Append Mode (a)	224 • File Locking Mode (k)	224 • Link Mode (l)	225 • Link Pair	225 • Optional allow and file Rules	225 • Owner Conditional Rules	226 • Deny Rules	227
23.8	Mount Rules	227								
23.9	Pivot Root Rules	229								
23.10	PTrace Rules	230								

23.11	Signal Rules	230
23.12	Execute Modes	231
	Discrete Profile Execute Mode (Px)	231 • Discrete Local Profile Execute Mode (Cx) 232 • Unconfined Execute Mode (Ux) 232 • Unsafe Exec Modes 232 • Inherit Execute Mode (ix) 233 • Allow Executable Mapping (m) 233 • Named Profile Transitions 233 • Fallback Modes for Profile Transitions 234 • Variable Settings in Execution Modes 235 • safe and unsafe Keywords 236
23.13	Resource Limit Control	236
23.14	Auditing Rules	238
24	AppArmor Profile Repositories	239
25	Building and Managing Profiles with YaST	240
25.1	Manually Adding a Profile	240
25.2	Editing Profiles	241
	Adding an Entry	243 • Editing an Entry 247 • Deleting an Entry 247
25.3	Deleting a Profile	247
25.4	Managing AppArmor	247
	Changing AppArmor Status	248 • Changing the Mode of Individual Profiles 249
26	Building Profiles from the Command Line	250
26.1	Checking the AppArmor Status	250
26.2	Building AppArmor Profiles	251
26.3	Adding or Creating an AppArmor Profile	252
26.4	Editing an AppArmor Profile	252
26.5	Unloading Unknown AppArmor Profiles	252
26.6	Deleting an AppArmor Profile	253

- 26.7 Two Methods of Profiling 253
 - Stand-Alone Profiling 254 • Systemic Profiling 254 • Summary of Profiling Tools 256
- 26.8 Important File Names and Directories 277
- 27 Profiling Your Web Applications Using ChangeHat 278**
- 27.1 Configuring Apache for mod_apparmor 279
 - Virtual Host Directives 280 • Location and Directory Directives 280
- 27.2 Managing ChangeHat-Aware Applications 281
 - With AppArmor's Command Line Tools 281 • Adding Hats and Entries to Hats in YaST 287
- 28 Confining Users with pam_apparmor 289**
- 29 Managing Profiled Applications 290**
- 29.1 Reacting to Security Event Rejections 290
- 29.2 Maintaining Your Security Profiles 290
 - Backing Up Your Security Profiles 290 • Changing Your Security Profiles 291 • Introducing New Software into Your Environment 291
- 30 Support 292**
- 30.1 Updating AppArmor Online 292
- 30.2 Using the Man Pages 292
- 30.3 For More Information 294
- 30.4 Troubleshooting 294
 - How to React to odd Application Behavior? 294 • My Profiles Do not Seem to Work Anymore ... 294 • Resolving Issues with Apache 298 • How to Exclude Certain Profiles from the List of Profiles Used? 298 • Can I Manage Profiles for Applications not Installed on my System? 298 • How to Spot and Fix AppArmor Syntax Errors 298
- 30.5 Reporting Bugs for AppArmor 299

31 AppArmor Glossary 301

V SELINUX 304

32 Configuring SELinux 305

32.1 Why Use SELinux? 305

Support Status 306 • Understanding SELinux Components 307

32.2 Policy 308

32.3 Installing SELinux Packages and Modifying GRUB 2 309

32.4 SELinux Policy 311

32.5 Configuring SELinux 312

32.6 Managing SELinux 314

Viewing the Security Context 314 • Selecting the SELinux Mode 316 • Modifying SELinux Context Types 317 • Applying File Contexts 319 • Configuring SELinux Policies 320 • Working with SELinux Modules 321

32.7 Troubleshooting 322

VI THE LINUX AUDIT FRAMEWORK 326

33 Understanding Linux Audit 327

33.1 Introducing the Components of Linux Audit 330

33.2 Configuring the Audit Daemon 332

33.3 Controlling the Audit System Using **auditctl** 337

33.4 Passing Parameters to the Audit System 339

33.5 Understanding the Audit Logs and Generating Reports 342

Understanding the Audit Logs 343 • Generating Custom Audit Reports 347

33.6 Querying the Audit Daemon Logs with **aureport** 354

33.7 Analyzing Processes with **auditd** 357

33.8 Visualizing Audit Data 358

33.9	Relaying Audit Event Notifications	360
34	Setting Up the Linux Audit Framework	363
34.1	Determining the Components to Audit	364
34.2	Configuring the Audit Daemon	364
34.3	Enabling Audit for System Calls	366
34.4	Setting Up Audit Rules	366
34.5	Configuring Audit Reports	368
34.6	Configuring Log Visualization	372
35	Introducing an Audit Rule Set	375
35.1	Adding Basic Audit Configuration Parameters	376
35.2	Adding Watches on Audit Log Files and Configuration Files	376
35.3	Monitoring File System Objects	377
35.4	Monitoring Security Configuration Files and Databases	379
35.5	Monitoring Miscellaneous System Calls	381
35.6	Filtering System Call Arguments	381
35.7	Managing Audit Event Records Using Keys	384
36	Useful Resources	386
A	GNU Licenses	388
A.1	GNU Free Documentation License	388

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

The following documentation is available for this product:

Book “Start-Up”

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

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.

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

2 Giving Feedback

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

Bug Reports

Report issues with the documentation at <https://bugzilla.opensuse.org/>. To simplify this process, you can use the *Report Documentation Bug* links next to headlines in the HTML version of this document. These preselect the right product and category in Bugzilla and add a link to the current section. You can start typing your bug report right away. A Bugzilla account is required.

Contributions

To contribute to this documentation, use the *Edit Source* links next to headlines in the HTML version of this document. They take you to the source code on GitHub, where you can open a pull request. A GitHub account is required.

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

Mail

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

Help

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

3 Documentation Conventions

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

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

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

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

```
tux > command
```

- Notices



Warning: Warning Notice

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



Important: Important Notice

Important information you should be aware of before proceeding.



Note: Note Notice

Additional information, for example about differences in software versions.



Tip: Tip Notice

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

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 Web sites with further information.

1.1 Overview

One main characteristic of Linux is its ability to handle multiple users at the same time (multi-user) 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.

Because of 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 9.3, "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 when 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 12, Encrypting Partitions and Files*.
- Use **cryptctl** to encrypt hosted storage. For more information, see *Chapter 13, Storage Encryption for Hosted Applications with cryptctl*.
- Use AIDE to detect any changes in your system configuration. For more information, see *Chapter 15, 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 and 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 11.1, "Traditional File Permissions"* and *Section 11.2, "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 12, 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 18, Masquerading and Firewalls*.
- Use OpenVPN to secure communication channels over insecure physical networks. For details, see *Chapter 19, 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 11.1, "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, often 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 **8**
- 3 Using NIS **18**
- 4 Setting Up Authentication Clients Using YaST **26**
- 5 LDAP—A Directory Service **28**
- 6 Network Authentication with Kerberos **44**
- 7 Active Directory Support **77**

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 SSSH (/etc/pam.d/sshd)

```
##PAM-1.0 ①
auth    requisite    pam_nologin.so      ②
auth    include      common-auth         ③
account requisite    pam_nologin.so      ②
account include      common-account      ③
password include      common-password     ③
session required     pam_loginuid.so     ④
```

```

session include      common-session      ③
session optional    pam_lastlog.so      silent noupdate showfailed ⑤

```

- ① 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.
- ② Checks, if `/etc/nologin` exists. If it does, no user other than `root` may log in.
- ③ 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.
- ④ Sets the login UID process attribute for the process that was authenticated.
- ⑤ 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      ①
auth optional pam_gnome_keyring.so ②
auth required pam_unix.so try_first_pass ③

```

- ① `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.
- ② `pam_gnome_keyring.so` checks the user's login and password against the GNOME key ring
- ③ `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"*.



Note: LUKS2 Support

LUKS2 support was added to `cryptsetup` 2.0, and openSUSE Leap has included support for LUKS2 in `pam_mount` since openSUSE Leap 42.3.

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` command 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*.

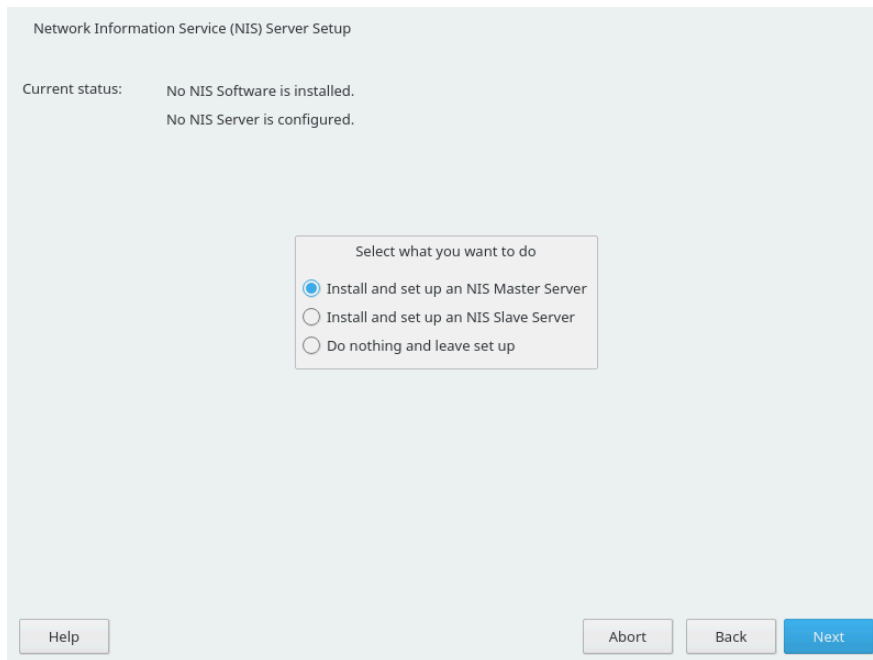


FIGURE 3.1: NIS SERVER SETUP

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.

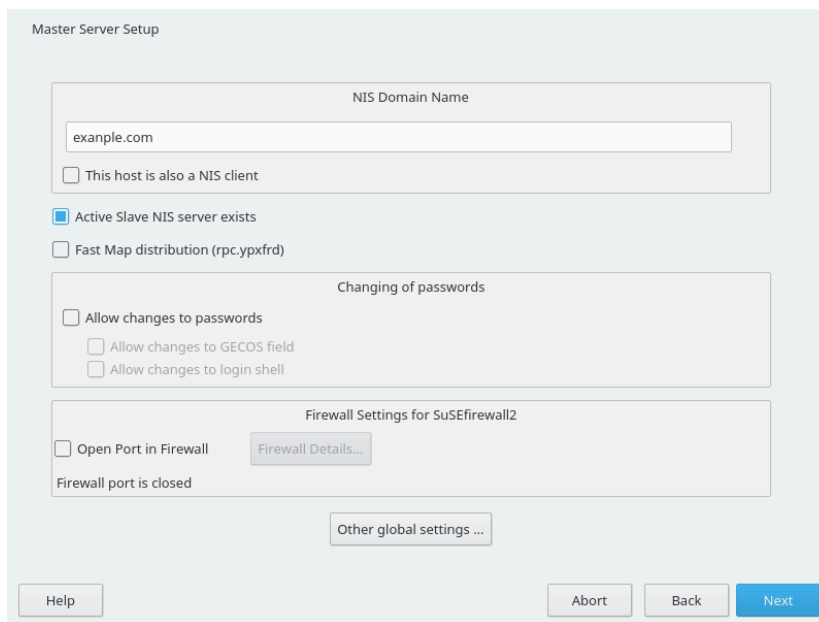


FIGURE 3.2: MASTER SERVER SETUP

- 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.

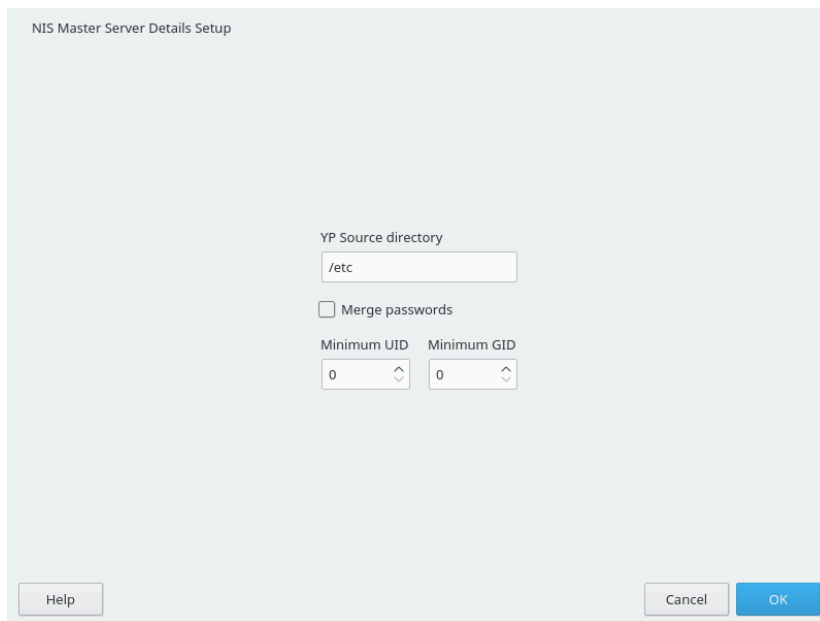


FIGURE 3.3: CHANGING THE DIRECTORY AND SYNCHRONIZING FILES FOR A NIS SERVER

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.

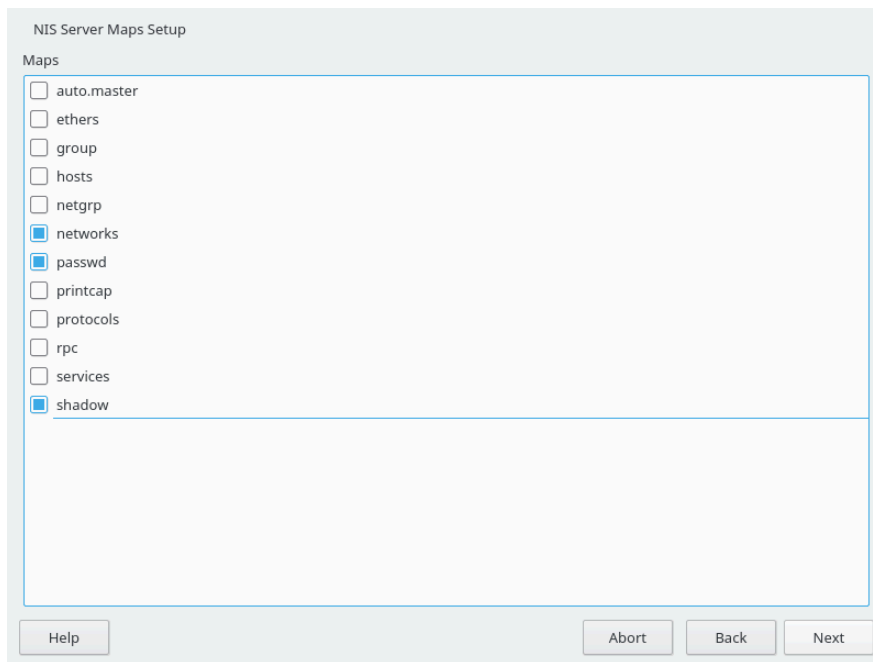


FIGURE 3.4: NIS SERVER MAPS SETUP

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.

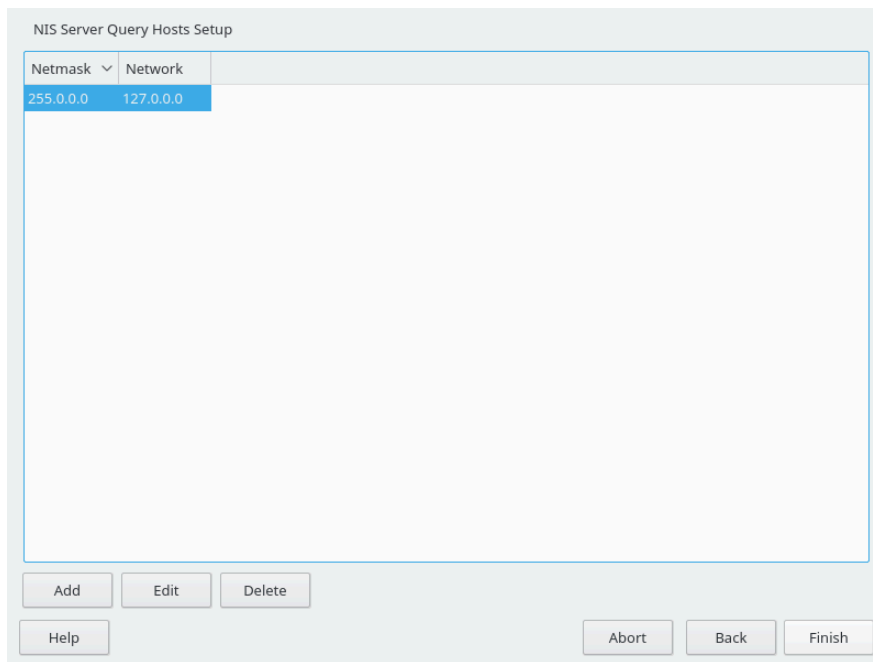


FIGURE 3.5: SETTING REQUEST PERMISSIONS FOR A NIS SERVER

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"*.

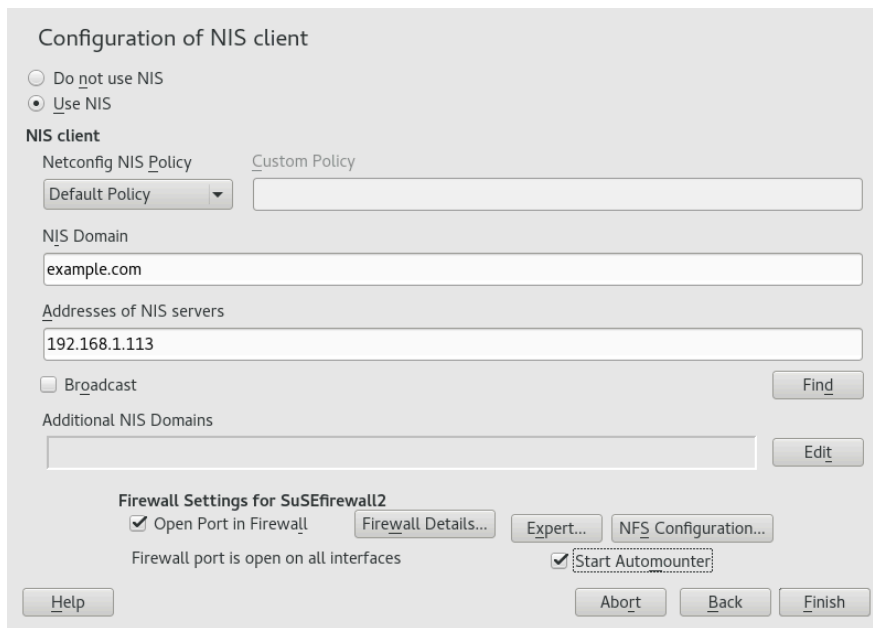


FIGURE 3.6: SETTING DOMAIN AND ADDRESS OF A NIS SERVER

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 Clients Using YaST

Whereas Kerberos is used for authentication, LDAP is used for authorization and identification. Both can work together. 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 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 5.4.2, “Configuring an LDAP Client with YaST”*
- Kerberos: *Section 6.6, “Setting up Kerberos using LDAP and Kerberos Client”*

4.2 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.

4.2.1 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/sss.service; enabled)
  Active: active (running) since Thu 2015-10-23 11:03:43 CEST; 5s ago
  [...]

```

4.2.2 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 **sss-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

```

5 LDAP—A Directory Service

The Lightweight Directory Access Protocol (LDAP) is a protocol 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 LDAP works.

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. The type of data stored in a directory tends to be long lived and changes infrequently. This allows the LDAP service to be optimized for high performance concurrent reads, whereas conventional databases are optimized for accepting many writes to data in a short time.

5.1 Structure of an LDAP Directory Tree

This section introduces the layout of an LDAP directory tree and provides the basic terminology used with regard to LDAP. If you are familiar with LDAP, read on at [Section 5.3, “Manually Configuring a 389 Directory 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. An object in the tree is identified by its *relative distinguished name* (RDN). The distinguished name is built from the RDN’s of all entries on the path to the entry.

The relations within an LDAP directory tree become more evident in the following example, shown in [Figure 5.1, “Structure of an LDAP Directory”](#).

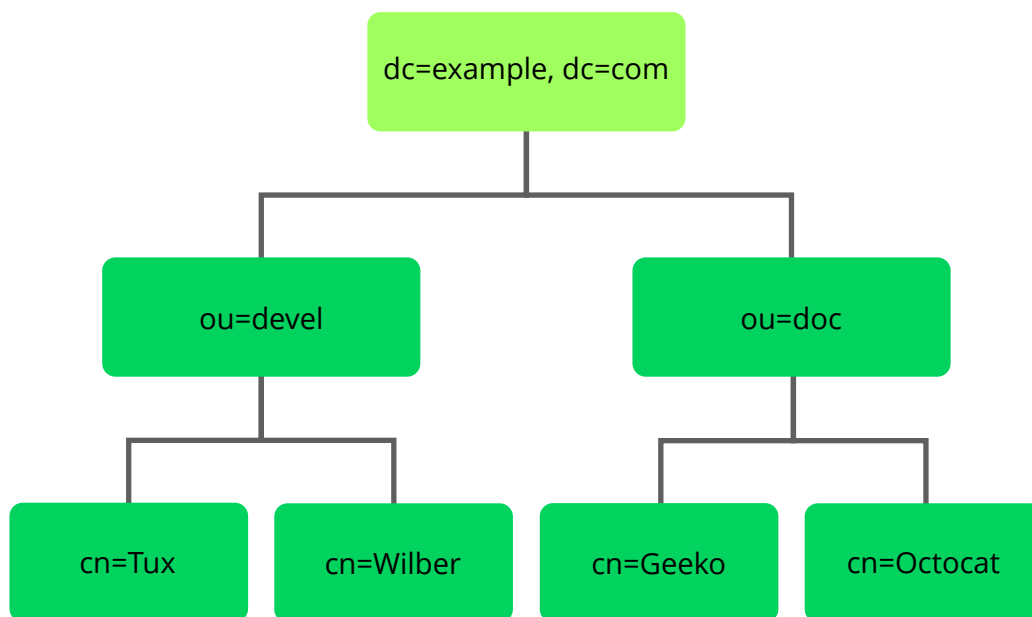


FIGURE 5.1: STRUCTURE OF AN LDAP DIRECTORY

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 may be assigned. The Schema contains all object classes and attributes which can be used by the LDAP server. Attributes are a structured data type. Their syntax, ordering and other behavior is defined by the Schema. LDAP servers supply a core set of Schemas which can work in a broad variety of environments. If a custom Schema is required, you can load it to an LDAP server.

Table 5.1, "Commonly Used Object Classes and Attributes" offers a small overview of the object classes from 00core.ldif and 06inetorgperson.ldif used in the example, including required attributes (Req. Attr.) and valid attribute values. After installing 389-ds, these can be found in usr/share/dirsrv/schema.

TABLE 5.1: COMMONLY USED OBJECT CLASSES AND ATTRIBUTES

Object Class	Meaning	Example Entry	Req. Attr.
<u>domain</u>	name components of the domain	example	display-Name
<u>organizationalUnit</u>	organizational unit	doc	ou
<u>nsPerson</u>	person-related data for the intranet or Internet	Geeko Linux	cn

Example 5.1, “Excerpt from CN=schema” shows an excerpt from a Schema directive with explanations.

EXAMPLE 5.1: EXCERPT FROM CN=SCHEMA

```

attributetype (1.2.840.113556.1.2.102 NAME 'memberOf' ❶
    DESC 'Group that the entry belongs to' ❷
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 ❸
    X-ORIGIN 'Netscape Delegated Administrator') ❹

objectclass (2.16.840.1.113730.3.2.333 NAME 'nsPerson' ❺
    DESC 'A representation of a person in a directory server' ❻
    SUP top STRUCTURAL ❼
    MUST ( displayName $ cn ) ❽
    MAY ( userPassword $ seeAlso $ description $ legalName $ mail \
        $ preferredLanguage ) ❾
    X-ORIGIN '389 Directory Server Project'
    ...

```

- ❶ The name of the attribute, its unique *object identifier* (OID, numerical), and the abbreviation of the attribute.
- ❷ A brief description of the attribute with DESC. The corresponding RFC, on which the definition is based, may also mentioned here.
- ❸ The type of data that can be held in the attribute. In this case, it is a case-insensitive directory string.
- ❹ The source of the schema element (for example, the name of the project).
- ❺ The definition of the object class nsPerson begins with an OID and the name of the object class (like the definition of the attribute).
- ❻ A brief description of the object class.

- 7 The SUP top entry indicates that this object class is not subordinate to another object class.
- 8 With MUST list all attribute types that must be used with an object of the type nsPerson.
- 9 With MAY list all attribute types that are optionally permitted with this object class.

5.2 Installing the Software for 389 Directory Server

The 389-ds package contains the 389 Directory Server and the administration tools. If the package is not installed yet, install it with the following command:

```
tux > sudo zypper install 389-ds
```

After installation, you can set up the server either manually (as described in [Section 5.3](#)) or create a very basic setup with YaST (as described in [Section 5.4](#)).

5.3 Manually Configuring a 389 Directory Server

Setting up the 389 Directory Server takes the following basic steps:

1. *Creating the 389 Directory Server Instance*
2. *Using CA Certificates for TSL*
3. *Configuring Admin Credentials for Remote/Local Access*
4. *Configuring LDAP Users and Groups*
5. *Setting Up SSSD*

The 389 Directory Server is controlled by 3 primary commands:

dsctl

Manages a local instance and requires root permissions. Requires you to be connected to a terminal which is running the directory server instance. Used for starting, stopping, backing up the database and more.

dsconf

The primary tool used for administration and configuration of the server. Manages an instance's configuration via its external interfaces. This allows you to make configuration changes remotely on the instance.

dsidm

Used for identity management (manage users, groups, passwords etc.). The permissions are granted by access controls, so users can reset their own password or change details of their own account, for example.

5.3.1 Creating the 389 Directory Server Instance

You create the instance with the **dscreate** command. It can take a configuration file (*.inf) which defines the instance configuration settings. Alternatively, the command can be run in an interactive mode.



Note: Instance Name

If not specified otherwise, the default instance name is localhost. The instance name cannot be changed after the instance has been created.

Example 5.2 shows an example configuration file that you can use as a starting point. Alternatively, use **dscreate create-template** to create a template *.inf file. The template is commented and pre-filled, so you can adjust its variables to your needs. For more details, see the man page of **dscreate**.

1. If you want to set up a trial instance, start an editor and save the following as /tmp/instance.inf:

EXAMPLE 5.2: BASIC INSTANCE CONFIGURATION FILE

```
# /tmp/instance.inf
[general]
config_version = 2

[slapd]
root_password = YOUR_PASSWORD_FOR_CN=DIRECTORY_MANAGER ❶

[backend-userroot]
sample_entries = yes
suffix = dc=example,dc=com
```

- ❶ Set the root_password to the password for the directory server root user. The password is used for LDAP server administration only.

2. To create the 389 Directory Server instance from *Example 5.2*, run:

```
tux > sudo dscreate from-file /tmp/instance.inf
```

This creates a working LDAP server.

3. If **dscreate** should fail, the messages will tell you why. For more details, repeat the command with the `-v` option:

```
tux > sudo dscreate -v from-file /tmp/instance.inf
```

4. Check the status of the server with:

```
tux > sudo dsctl localhost status
instance 'Localhost' is running
```

5. In case you want to delete the instance later on:

```
tux > sudo dsctl localhost remove --do-it
```

With this command, you can also remove partially installed or corrupted instances.

5.3.2 Using CA Certificates for TSL

You can manage the CA certificates for 389 Directory Server with the following command line tools: **certutil**, **openssl**, and **pk12util**.

For testing purposes, you can create a self-signed certificate with **dscreate**. Find the certificate at `/etc/dirsrv/slapd-localhost/ca.crt`. For remote administration, copy the certificate to a readable location. For production environments, contact a CA authority of your organization's choice and request a server certificate, a client certificate, and a root certificate.

Make sure to meet the following requirements before executing the procedure below:

- You have a server certificate and a private key to use for the TSL connection.
- You have set up an NSS (Network Security Services) database (for example, with the **certutil** command).

Before you can import an existing private key and certificate into the NSS (Network Security Services) database, you need to create a bundle of the private key and the server certificate. This results in a `*.p12` file.

! Important: *.p12 File and Friendly Name

When creating the PKCS12 bundle, you must encode a friendly name in the *.p12 file.

Make sure to use Server-Cert as the friendly name. Otherwise the TLS connection will fail, because the 389 Directory Server searches for this exact string.

Keep in mind that the friendly name cannot be changed after you import the *.p12 file into the NSS database.

1. Use the following command to create the PKCS12 bundle with the required friendly name:

```
root # openssl pkcs12 -export -in SERVER.crt \  
-inkey SERVER.key -out SERVER.p12 \  
-name Server-Cert
```

Replace SERVER.crt with the server certificate and SERVER.key with the private key to be bundled. With -out, specify the name of the *.p12 file. Use -name to set the friendly name to use, Server-Cert.

2. Before you can import the file into the NSS database, you need to obtain its password. To do this, use the following command:

```
pk12util -i PATH_TO_SERVER.p12 -d sql:PATH_TO_NSS_DB -n Server-cert -  
W SERVER.p12_PASSWORD
```

You can then find the password in the pwdfile.txt file in the PATH_TO_NSS_DB directory.

3. Now import the SERVER.p12 file, into your NSS database:

```
pk12util -i SERVER.p12 -d PATH_TO_NSS_DB
```

5.3.3 Configuring Admin Credentials for Remote/Local Access

For remote or local administration of the 389 Directory Server, you can create a .dsrc configuration file in your home directory. This saves you to type your user name and connection details with every command. *Example 5.3* shows an example configuration file for remote administration, whereas *Example 5.4* shows one for local administration.

EXAMPLE 5.3: A `.dsrc` FILE FOR REMOTE ADMINISTRATION

```
# cat ~/.dsrc
[localhost]
uri = ldaps://localhost ❶
basedn = dc=example,dc=com
binddn = cn=Directory Manager
tls_cacertdir = PATH_TO_CERTDIR ❷
```

- ❶ Needs to point to the LDAP server instance. If not specified otherwise, the default instance name is `localhost`.
- ❷ Path to the certificate at a readable location or on the client machine from which you use the `ds*` commands.

If you want to administer the instance on the same host where the 389 Directory Server runs, use the configuration file in [Example 5.4](#).

EXAMPLE 5.4: A `.dsrc` FILE FOR LOCAL ADMINISTRATION

```
# cat ~/.dsrc
[localhost]
# Note that '/' is replaced with '%2f'.
uri = ldapi://%%2fvar%%2frun%%2fslapd-localhost.socket ❶
basedn = dc=example,dc=com
binddn = cn=Directory Manager
```

- ❶ When using `ldapi` on the server where the 389 Directory Server instance is running, your UID/GID will be detected. If it is `0/0` (which means you are logged in as `root` user), the `ldapi` binds the local `root` as the directory server root dn (`cn=Directory Manager`) of the instance. This allows local administration of the server, but also allows you to set a machine-generated password for `cn=Directory Manager` that no human knows. Whoever has administrator rights on the server hosting the 389 Directory Server instance, can access the instance as `cn=Directory Manager`.

5.3.4 Configuring LDAP Users and Groups

Users and groups can be created and managed with the `dsidm` command. It either runs interactively or you can use it with arguments from the command line.

In the following example, we add two users, `wilber` and `geeko`, by specifying their data via command line arguments.

PROCEDURE 5.1: CREATING LDAP USERS

1. Create the user wilber:

```
tux > sudo dsidm localhost user create --uid \  
--cn wilber --displayName 'Wilber Fox' --uidNumber 1000 --gidNumber 1000 \  
--homeDirectory /home/wilber
```

2. To look up a user's distinguished name (fully qualified name to the directory object, which is guaranteed unique):

```
tux > sudo dsidm localhost user get wilber  
dn: uid=wilber,ou=people,dc=example,dc=com  
[...]
```

The system prompts you for the directory server root user password (unless you configured remote or local access as described in [Section 5.3.3, “Configuring Admin Credentials for Remote/Local Access”](#)).

You need the distinguished name for actions such as changing the password for a user.

3. To set or change the password for wilber:

- a.

```
tux > sudo dsidm localhost account reset_password \  
uid=wilber,ou=people,dc=example,dc=com
```

The system prompts you for the directory server root user password (unless you configured remote or local access as described in [Section 5.3.3, “Configuring Admin Credentials for Remote/Local Access”](#)).

- b. Enter the new password for wilber twice.

If the action was successful, you get the following message:

```
reset password for uid=wilber,ou=people,dc=example,dc=com
```

4. Create the user geeko:

```
tux > sudo dsidm localhost user create --uid \  
--cn geeko --displayName 'Suzanne Geeko' \  
--uidNumber 1001 --gidNumber 1001 --homeDirectory /home/geeko
```

PROCEDURE 5.2: CREATING LDAP GROUPS AND ASSIGNING USERS TO THEM

In the following, we create a group, server_admins, and assign the user wilber to this group.

1. Create the group:

```
tux > sudo dsidm localhost group create
```

You will be prompted for a group name:

```
Enter value for cn :
```

2. Enter the name for the group, for example: server_admins.

3. Add the user wilber to the group:

```
tux > sudo dsidm localhost group add_member server_admins
uid=wilber,ou=people,dc=example,dc=com
added member: uid=wilber,ou=people,dc=example,dc=com
```

4. Verify if authentication works:

```
tux > sudo ldapwhoami -H ldaps://localhost -D \
uid=wilber,ou=people,dc=example,dc=com -W -x
```

If you are prompted for the LDAP password of wilber, authentication works.

If the command fails with the following error, you are probably using a self-signed certificate:

```
ldap_sasl_bind(SIMPLE): Can't contact LDAP server (-1)
```

In that case, edit /etc/openldap/ldap.conf and add the path to the certificate. For example:

```
TLS_CACERT /etc/dirsrv/slapd-localhost/ca.crt
```

Alternatively, include the path to the certificate in the whoami command:

```
tux > sudo LDAPTLS_CACERT=/etc/dirsrv/slapd-localhost/ca.crt \
ldapwhoami -H ldaps://localhost -D \
uid=wilber,ou=people,dc=example,dc=com -W -x
```

5.3.5 Setting Up SSSD

SSSD (System Security Services Daemon) is a daemon that communicates with remote identity providers and allows `pam` and `nsswitch` to consume that data. SSSD can have multiple backends, cache users and groups and provides features like SSH key distributions.

1. On a separate server, install the `sssd` package:

```
tux > sudo zypper in sssd
```

2. Disable and stop the `nscd` daemon because it conflicts with `sssd`:

```
tux > sudo systemctl disable nscd && systemctl stop nscd
```

3. Create the SSSD configuration and restrict the login to the members of the group `server_admins` that we created in *Procedure 5.2*:

```
tux > sudo dsidm localhost client_config sssd.conf server_admins
```

4. Review the output and paste (or redirect) it to `/etc/sssds/sssds.conf`. If required, edit the configuration file according to your needs.

5. To configure the certificates on your client, copy `ca.crt` from the LDAP server to your client:

```
tux > sudo mkdir -p /etc/openldap/certs  
cp [...]>/ca.crt /etc/openldap/certs/  
/usr/bin/c_rehash /etc/openldap/certs
```

6. Enable and start SSSD:

```
tux > sudo systemctl enable sssd  
systemctl start sssd
```

7. To make sure SSSD is part of PAM and NSS, follow the instructions in sections *Configure PAM (SUSE)* and *Configure NSS (SUSE)* at <http://www.port389.org/docs/389ds/howto/howto-sssds.html>.

8. Verify if the client can provide the details for user `wilber`:

```
tux > sudo id wilber  
uid=1000(wilber) gid=100(users) groups=100(users)
```


If everything is set up correctly, wilber can access the 389 Directory Server instance via SSH to the machine where you have installed and configured SSSD. However, geeko will fail to do so, because geeko does not belong to the group server_admins that we have configured in *Procedure 5.2*.

5.4 Setting Up a 389 Directory Server with YaST

You can use YaST to quickly create a very basic setup of the 389 Directory Server.

5.4.1 Creating a 389 Directory Server Instance with YaST

1. In YaST, click *Network Services* > *Create New Directory Server*. Alternatively, start the module from command line with `yast2 ldap-server`.
In the window that opens, you need to fill in all mandatory text fields.
2. Enter the *Fully qualified domain name* of the 389 Directory Server. It must be resolvable from the host.
3. In *Directory server instance name*, enter a local name for the LDAP server instance.



Note: Instance Name

The instance name *cannot* be changed after the instance has been created. If you plan for only one LDAP server, use the default instance name localhost. However, if you plan to host multiple LDAP servers, use meaningful names for the individual instances.

4. In *Directory suffix*, enter the base domain name of the LDAP tree. It is your domain name split by component. For example, example.com becomes dc=example,dc=com.
5. In the mandatory security options, enter the password for the directory manager (LDAP's root/admin account) and repeat the password in the next step. The password must be at least 8 characters long.
6. To run 389 Directory Server with a CA certificate, specify both of the following options:
 - a. Enter the path to the *Server TLS certificate authority in PEM format*, with which the server certificates have been signed.

b. Enter the path to the *Server TLS certificate and key in PKCS12 format with friendly name "Server-Cert"*. The `*.p12` file contains the server's private key and certificate. These must have been signed by the CA in PEM format that you have specified above. The *friendly name* must be `Server-Cert`, see [Section 5.3.2, "Using CA Certificates for TLS"](#) for details.

If you do not specify a CA certificate here, a self-signed certificate will be created automatically. After the instance has been created, find the related files in `/etc/dirsrv/slapd-INSTANCENAME`.

7. If you are ready to create the instance, click **OK**.

Create New Directory Instance

General options (mandatory)

Fully qualified domain name (e.g. dir.example.net)
dir.example.net

Directory server instance name (e.g. localhost)
localhost

Directory suffix (e.g. dc=example,dc=net)
dc=example,dc=net

Security options (mandatory)

"cn=Directory Manager" password
.....

Repeat "cn=Directory Manager" password
.....

Security options (optional)

Server TLS certificate authority in PEM format
/root/server.pem

Server TLS certificate and key in PKCS12 format with friendly name "Server-Cert"
/root/server.p12

OK Cancel

YaST displays a message whether the creation was successful and where to find the log files.

The setup with YaST provides only a very basic configuration of the 389 Directory Server. To fine-tune more settings, see [Section 5.3, "Manually Configuring a 389 Directory Server"](#) or the documentation mentioned in [Section 5.6, "For More Information"](#).

5.4.2 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 4.1, "Configuring an Authentication Client with YaST"](#).

Start the module by selecting *Network Services > LDAP and Kerberos Client*.

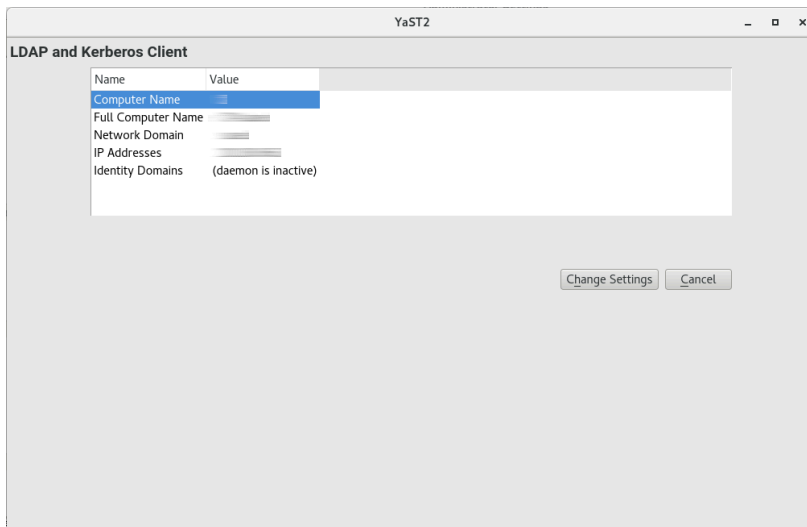
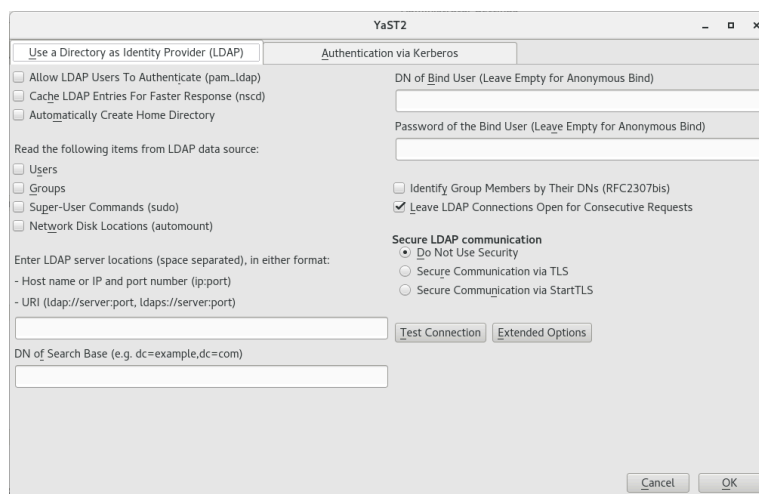


FIGURE 5.2: *LDAP AND KERBEROS CLIENT WINDOW*

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 or host names under *Enter LDAP server locations*. For security reasons, we recommend to use LDAPS:// URLs only. 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.
- If you want to cache LDAP entries locally, use *Cache LDAP Entries For Faster Response*.



Warning: Potential Security Risk with Caching

Using the cache bears security risks, depending on the used mechanism.

nscd

If you define an authorization rule (for example, members of group admin can log in), and you remove a user from that group, the client cache will not see that change until the cache expires or refreshes. So a user whose account has been revoked can still log in later.

sss

This caching mechanism constantly checks if group memberships are still valid. Thus the cache risk only exists if the sss daemon is disconnected from the LDAP server for any reason.

- 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*.

5.5 Manually Administering LDAP Data

The command line tools provided by the `openldap2-client` package (like `ldapsearch` or `ldapmodify`) can be used for administration of data in the LDAP directory. However, they are low-level tools and hard to use. For details about their use, refer to the respective man pages and documentation.

5.6 For More Information

For more information about 389 Directory Server, see the upstream documentation, available at <http://www.port389.org/docs/389ds/documentation.html>.

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 other's 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 con-

tains 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.

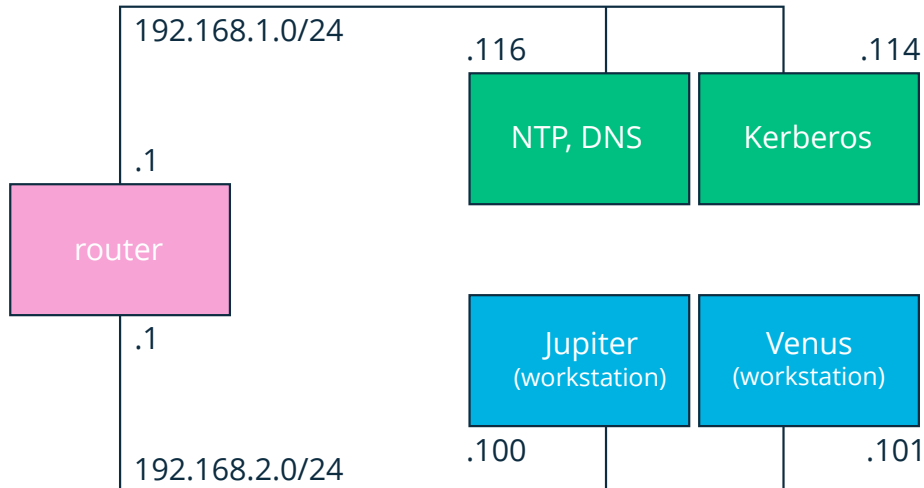


FIGURE 6.1: KERBEROS NETWORK TOPOLOGY



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. See *Section 6.5.5.1, “Configuring the Server”*.
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 6.5.5.2, “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 ad-

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 6.5.5.3, “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 6.5.5.4, “Starting the KDC”](#) for details.
7. **Create a Principal for Yourself.** You need a principal for yourself. Refer to [Section 6.5.5.3, “Creating a Principal”](#) for details.

6.5.5.1 Configuring the Server

Configuring a Kerberos server is highly variable, dependent on your network architecture, DNS and DHCP configuration, realms, and other considerations. You must have a default realm, and domain- to-realm mappings. The following example demonstrates a minimal configuration. This is not a copy-and-paste example; see https://web.mit.edu/kerberos/krb5-latest/doc/admin/conf_files/index.html for detailed information on Kerberos configuration.

EXAMPLE 6.1: EXAMPLE KDC CONFIGURATION, `/etc/krb5.conf`

```
[libdefaults]
dns_canonicalize_hostname = false
rdns = false
default_realm = example.com
ticket_lifetime = 24h
renew_lifetime = 7d

[realms]
example.com = {
kdc = kdc.example.com.:88
admin_server = kdc.example.com
default_domain = example.com
}

[logging]
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log
```

```
default = SYSLOG:NOTICE:DAEMON
```

```
[domain_realm]
```

```
.example.com = example.com
```

```
example.com = example.com
```

6.5.5.2 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: ①
Re-enter KDC database master key to verify: ②
```

- ① Type the master password.
- ② 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
```

6.5.5.3 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.
- ❷ 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.

6.5.5.4 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
<code>host</code>	Telnet, RSH, SSH
<code>nfs</code>	NFSv4 (with Kerberos support)
<code>HTTP</code>	HTTP (with Kerberos authentication)
<code>imap</code>	IMAP
<code>pop</code>	POP3
<code>ldap</code>	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 need 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

While Kerberos provides authentication, LDAP is used for authorization and identification. Both services can work together.

For secure connections, 389 Directory Server supports different ways of encrypting data: SSL/TLS connections, Start TLS connections, and SASL authentication. Simple Authentication and Security Layer (SASL) is a network protocol designed for authentication. The SASL implemen-

tation used on openSUSE Leap is `cyrus-sasl`. Kerberos authentication is performed through GSS-API (General Security Services API), provided by the `cyrus-sasl-gssapi` package. Using GSS-API, 389 Directory Server uses Kerberos tickets to authenticate sessions and encrypt data. With the SASL framework you can use different mechanisms to authenticate a user to the server. In Kerberos, authentication is always mutual. This means that not only have you authenticated yourself to the 389 Directory Server, but also the 389 Directory Server has authenticated itself to you. In particular, this means communication is with the desired server, rather than with a random service set up by an attacker.

To enable Kerberos to bind to the 389 Directory Server, create a principal `ldap/ldap.example.com` and add that to the keytab. The credentials used by the 389 Directory Server to authenticate are given to other servers by the keytab. 389 Directory Server assigns a keytab through the `KRB5_KTNAME` environment variable.

To set the variable, proceed as follows:

1.

```
tux > sudo systemctl edit dirsrv@INSTANCE
```

If you used the default name for the 389 Directory Server instance, replace `INSTANCE` with `localhost`.

2. Add the following:

```
[Service]
Environment=KRB5_KTNAME=/etc/dirsrv/slapd-INSTANCE/krb5.keytab
```

3. The keytab file needs to be readable by the account under which the 389 Directory Server runs (for example, `dirserv`):

```
tux > sudo chown dirsrv:dirsrv /etc/dirsrv/slapd-INSTANCE/krb5.keytab
tux > sudo chmod 600 /etc/dirsrv/slapd-INSTANCE/krb5.keytab
```

6.5.11.1 Using Kerberos Authentication with LDAP

To obtain and cache an initial ticket-granting ticket, use the principal that has been created in [Section 6.5.5.3, "Creating a Principal"](#):

```
tux > kinit geeko@EXAMPLE.COM
```

To check if GSSAPI authentication works, run:

```
tux > ldapwhoami -Y GSSAPI -H ldap://ldapkdc.example.com
```

```
dn: uid=testuser,ou=People,dc=example,dc=com
```

GSSAPI uses the `ccache` to authenticate the user to the 389 Directory Server without the user's password.

6.5.11.2 Configuring SASL Identity Mapping

When processing a SASL bind request, the 389 Directory Server maps the SASL authentication ID (used to authenticate to the Directory Server) with an LDAP entry stored within the server. When using Kerberos, the SASL user ID usually has the following format: `userid@REALM`, such as `tux@example.com`. This ID must be converted into the DN of the user's Directory Server entry, such as `uid=tux,ou=people,dc=example,dc=com`. The 389 Directory Server comes with some default maps for most common configurations. However, you can create customized maps. [Procedure 6.1, "Managing Maps"](#) shows how to list and display a map, how to delete a map and how to create a custom map.

PROCEDURE 6.1: MANAGING MAPS

1. To list the existing SASL maps:

```
tux > dsconf INSTANCE sasl list
Kerberos uid mapping
rfc 2829 dn syntax
rfc 2829u syntax
uid mapping
```

2. To display a map:

```
tux > sudo dsconf INSTANCE sasl get "Kerberos uid mapping"
dn: cn=Kerberos uid mapping,cn=mapping,cn=sasl,cn=config
cn: Kerberos uid mapping
nsSaslMapBaseDNTemplate: dc=\2,dc=\3
nsSaslMapFilterTemplate: (uid=\1)
nsSaslMapRegexString: \(.*\)\@\(.*\)\. \(.*\)
objectClass: top
objectClass: nsSaslMapping
```

3. The default map only works if your dc has two components. To delete the map (if it does not work for you):

```
tux > sudo dsconf INSTANCE sasl delete "Kerberos uid mapping"
```

```
Deleting SaslMapping cn=Kerberos uid mapping,cn=mapping,cn=sasl,cn=config :  
Successfully deleted cn=Kerberos uid mapping,cn=mapping,cn=sasl,cn=config
```

4. To create a new map:

```
tux > sudo dsconf localhost sasl create --cn=bhgssapi --nsSaslMapRegexString "\  
(.*\.)@EXAMPLE.NET.DE" --nsSaslMapBaseDNTemplate="dc=example,dc=net,dc=de" --  
nsSaslMapFilterTemplate="(uid=\1)"  
tux > sudo Enter value for nsSaslMapPriority :  
Successfully created bhgssapi
```

5. Display the newly created map with:

```
tux > sudo dsconf localhost sasl get "bhgssapi"  
dn: cn=bhgssapi,cn=mapping,cn=sasl,cn=config  
cn: bhgssapi  
nsSaslMapBaseDNTemplate: dc=example,dc=net,dc=de  
nsSaslMapFilterTemplate: (uid=\1)  
nsSaslMapPriority: 100  
nsSaslMapRegexString: \(.*\.)@EXAMPLE.NET.DE  
objectClass: top  
objectClass: nsSaslMapping
```

With this, you can check only the users of a specific realm and remap them to a different dc base. As you can see, the new map has 3 `dc` components, so the default maps would not have worked for this realm (`EXAMPLE.NET.DE`), only for a realm like `EXAMPLE.NET`.

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 4.1, "Configuring an Authentication Client with YaST"](#).

Start the module by selecting *Network Services > LDAP and Kerberos Client*.

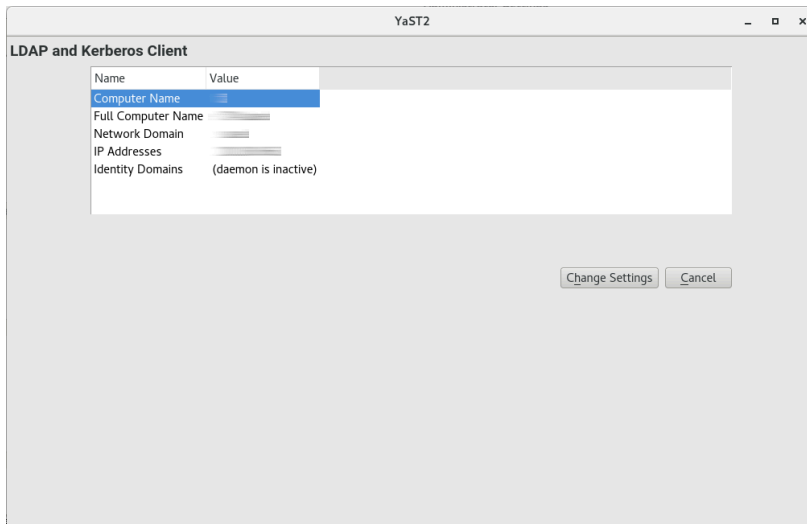
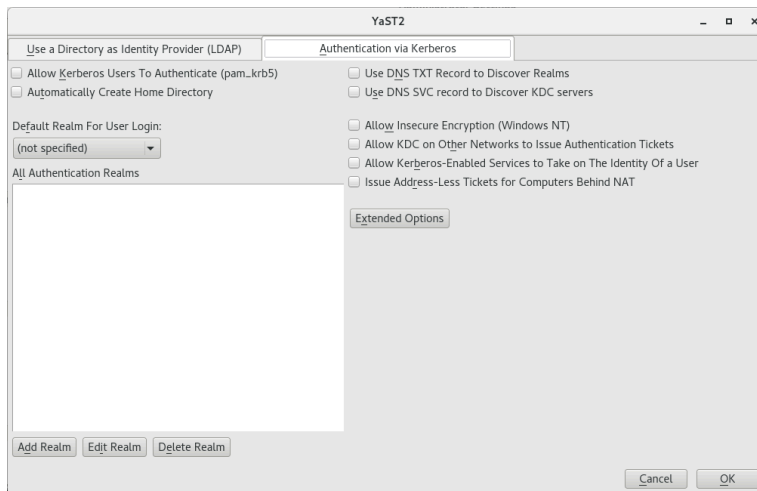


FIGURE 6.2: *LDAP AND KERBEROS CLIENT WINDOW*

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.

To check if the setup of the Kerberos back-end inside of LDAP was successful, proceed as follows:

1. Directly access the KDC database on the host of the 389 Directory Server:

```
tux > sudo kadmin.local
```

2. List the principals:

```
kadmin.local > listprincs
```

3. Create a principal:

```
kadmin.local > ank admin@EXAMPLE.COM
```

It is written to the 389 Directory Server database.

- 4.

```
tux > sudo ldapsearch -D 'cn=Directory Manager' -w password -b  
'cn=EXAMPLE.COM,cn=kdc,dc=example,dc=com' -H ldaps://localhost
```

5. Check if the principle data from Kerberos is stored in LDAP. If yes, you get an output similar to the following:

```
tux > sudo admin@EXAMPLE.COM, EXAMPLE.COM, kdc, example.com  
dn: krbprincipalname=admin@EXAMPLE.COM,cn=EXAMPLE.COM,cn=kdc,dc=example,dc=com  
krbLoginFailedCount: 0  
krbPrincipalName: admin@EXAMPLE.COM  
krbPrincipalKey:: MIG2oAMCAQGhAwIBAAIDAgEBowMCAQGkgZ8wgZwwVKAHMAWgAwIBAKFJMEeg  
AwIBEqFABD4gAKXAsMf7oV5vITzV50pcLhdomR+SdIRckouS2GeNF9lVgxjT29RpnipnLCjgG0kpr  
93d0nh82WhrrAF6bzBEoAcwBaADAgEAoTkWn6ADAgERoTAELhAAFiGRiI0yUjBteGHhTB6ESJYsYJ  
WxFa4UslUNZD1GEQGLZ/0nltLsyD2ytGc=  
krbLastPwdChange: 20190702032802Z  
krbExtraData:: AAJCzxpdcM9vdC9hZG1pbkBFWEFNUEXFLkNPTQA=
```

```
krbExtraData:: AAgBAA==
objectClass: krbprincipal
objectClass: krbprincipalaux
objectClass: krbTicketPolicyAux
objectClass: top
```

6. Obtain and cache an initial ticket-granting ticket:

```
tux > sudo kinit admin@EXAMPLE.COM
```

7. Display a list of currently cached Kerberos tickets:

```
tux > sudo klist
Ticket cache: DIR:./run/user/0/krb5cc/tkt
Default principal: admin@EXAMPLE.COM

Valid starting    Expires          Service principal
07/02/19 13:29:04  07/03/19 13:29:04  krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

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 several 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. 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.

Usually 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 does not 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 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`). 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 part 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/) (<https://blogs.technet.microsoft.com/activedirectoryua/2016/02/09/identity-management-for-unix-idmu-is-deprecated-in-windows-server/>)⁷.

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, refer to *Chapter 5, LDAP—A Directory Service*.

Kerberos

Kerberos is a third-party trusted authentication service. All its clients trust Kerberos 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_mkhome-dir`.

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_mkhome-dir`.

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.

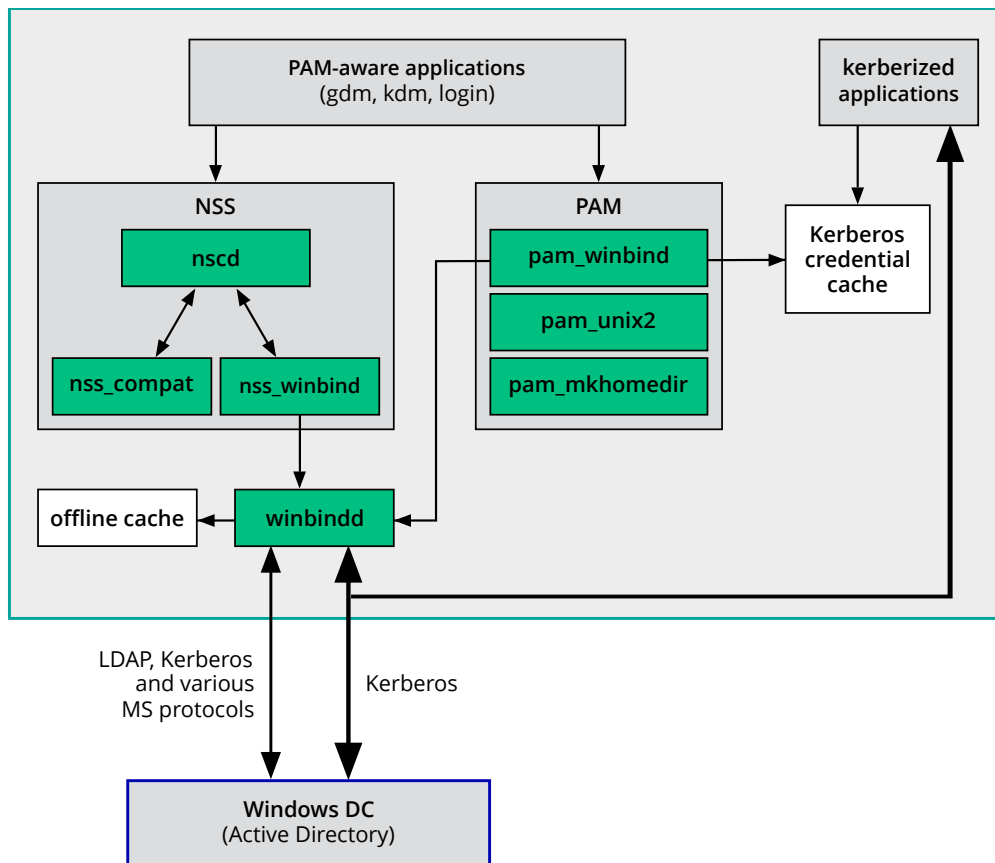


FIGURE 7.1: SCHEMA OF WINBIND-BASED ACTIVE DIRECTORY AUTHENTICATION

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 5.4.2, “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.

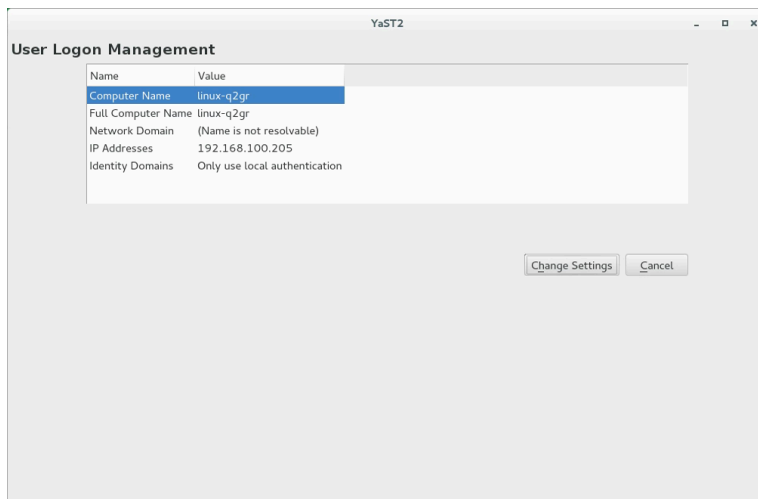


FIGURE 7.2: MAIN WINDOW OF USER LOGON MANAGEMENT

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*.

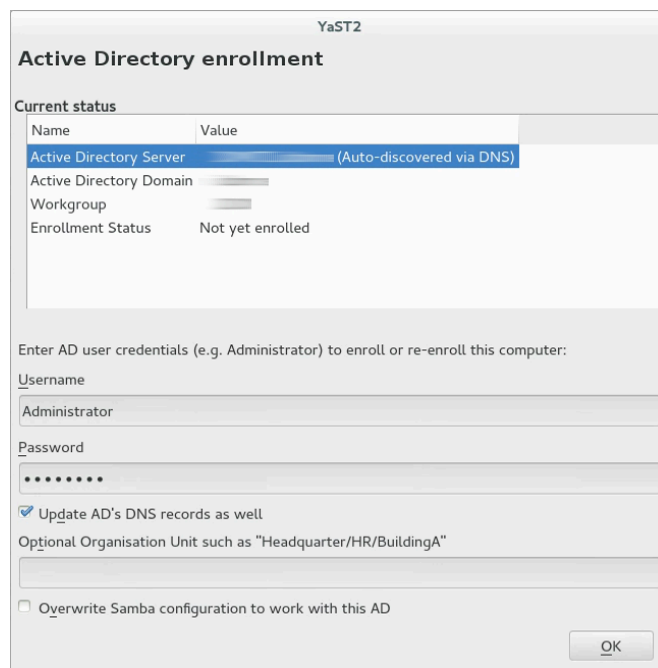


FIGURE 7.3: ENROLLING INTO A DOMAIN

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*.

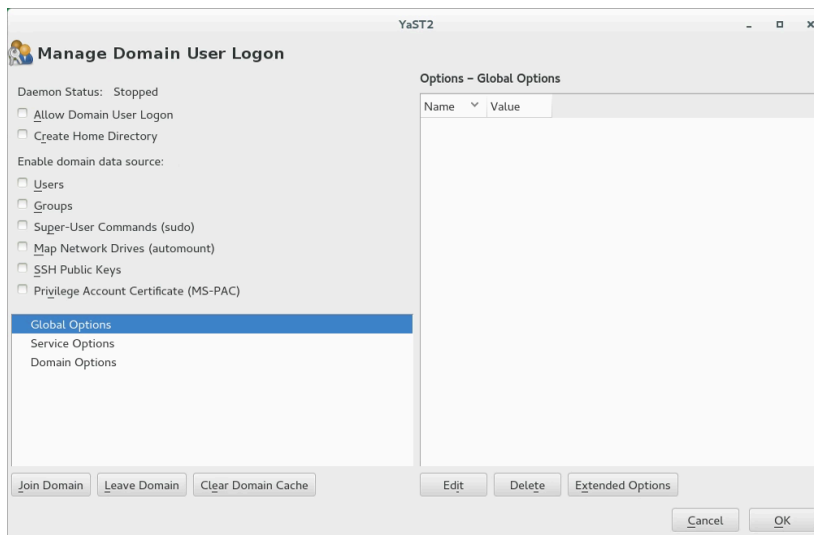


FIGURE 7.4: CONFIGURATION WINDOW OF USER LOGON MANAGEMENT

- 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.

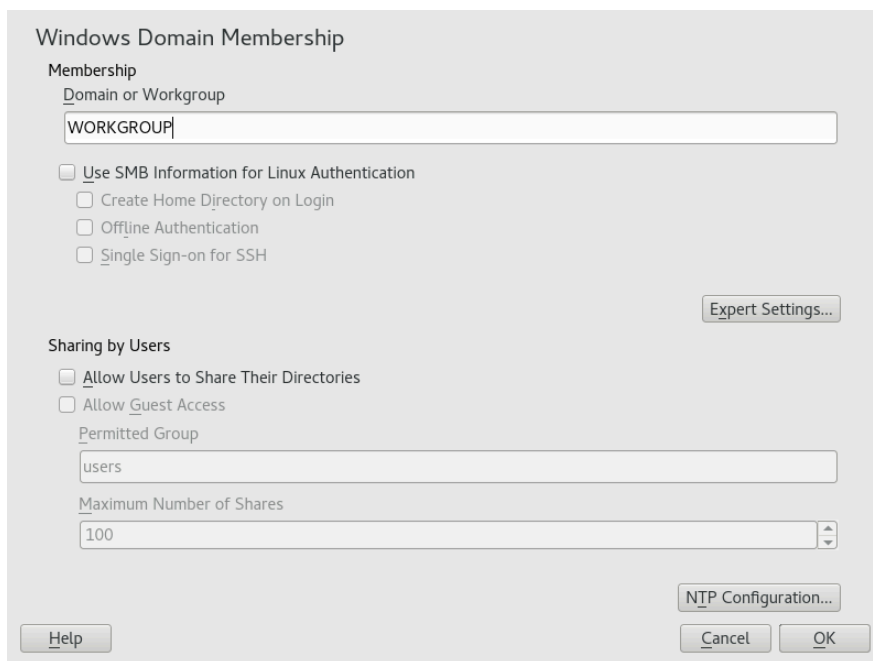


FIGURE 7.5: DETERMINING WINDOWS DOMAIN MEMBERSHIP

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"*).

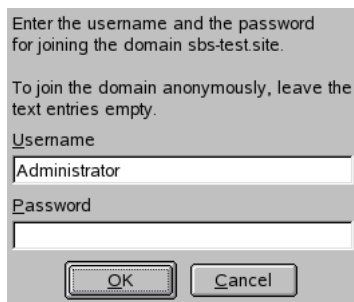


FIGURE 7.6: PROVIDING ADMINISTRATOR CREDENTIALS

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 Spectre/Meltdown Checker **96**
- 9 Configuring Security Settings with YaST **99**
- 10 Authorization with PolKit **104**
- 11 Access Control Lists in Linux **114**
- 12 Encrypting Partitions and Files **126**
- 13 Storage Encryption for Hosted Applications with cryptctl **130**
- 14 Certificate Store **138**
- 15 Intrusion Detection with AIDE **140**

8 Spectre/Meltdown Checker

`spectre-meltdown-checker` is a shell script to test if your system is vulnerable to the several speculative execution vulnerabilities that are in nearly all CPUs manufactured in the past 20 years. This is a hardware flaw that potentially allows an attacker to read all data on the system. On cloud computing services, where multiple virtual machines are on a single physical host, an attacker can gain access to all virtual machines. Fixing these vulnerabilities requires re-designing and replacing CPUs. Until this happens, there are several software patches that mitigate these vulnerabilities. If you have kept your SUSE systems updated, all of these patches should already be installed.

`spectre-meltdown-checker` generates a detailed report. It is impossible to guarantee that your system is secure, but it shows you which mitigations are in place, and potential vulnerabilities

8.1 Using `spectre-meltdown-checker`

Install the script, and then run it as root without any options:

```
root # zypper in spectre-meltdown-checker
root # spectre-meltdown-checker.sh
```

You will see colorful output like *Figure 8.1, "Output From spectre-meltdown-checker"* :

```
dreamer:/home/carla # spectre-meltdown-checker.sh
Spectre and Meltdown mitigation detection tool v0.40

Checking for vulnerabilities on current system
Kernel is Linux 4.12.14-lp151.28.13-default #1 SMP Wed Aug 7 07:20:16 UTC 2019 (0c09ad2) x86_64
CPU is Intel(R) Xeon(R) CPU E5-1620 v3 @ 3.50GHz

Hardware check
* Hardware support (CPU microcode) for mitigation techniques
* Indirect Branch Restricted Speculation (IBRS)
  * SPEC_CTRL MSR is available: YES
  * CPU indicates IBRS capability: YES (SPEC_CTRL feature bit)
* Indirect Branch Prediction Barrier (IBPB)
  * PRED_CMD MSR is available: YES
  * CPU indicates IBPB capability: YES (SPEC_CTRL feature bit)
* Single Thread Indirect Branch Predictors (STIBP)
  * SPEC_CTRL MSR is available: YES
  * CPU indicates STIBP capability: YES (Intel STIBP feature bit)
* Speculative Store Bypass Disable (SSBD)
  * CPU indicates SSBD capability: YES (Intel SSBD)
* L1 data cache invalidation
  * FLUSH_CMD MSR is available: YES
  * CPU indicates L1D flush capability: YES (L1D flush feature bit)
* Enhanced IBRS (IBRS_ALL)
  * CPU indicates ARCH_CAPABILITIES MSR availability: NO
  * ARCH_CAPABILITIES MSR advertises IBRS_ALL capability: NO
```

FIGURE 8.1: OUTPUT FROM SPECTRE-MELTDOWN-CHECKER

`spectre-meltdown-checker.sh --help` lists all options. It is useful to pipe plain text output, with no colors, to a file:

```
root # spectre-meltdown-checker.sh --no-color | tee filename.txt
```

The previous examples are on a running system, which is the default. You may also run `spectre-meltdown-checker` offline by specifying the paths to the kernel, config, and System.map files :

```
root # cd /boot
root # spectre-meltdown-checker.sh \
--no-color \
--kernel vmlinuz-4.12.14-lp151.28.13-default \
--config config-4.12.14-lp151.28.13-default \
--map System.map-4.12.14-lp151.28.13-default | tee filename.txt
```

Other useful options are:

`--verbose, -v`

Increase verbosity; repeat for more verbosity, for example `-v -v -v`

`--explain`

Print human-readable explanations

`--batch [short] [json] [nrpe] [prometheus]`

Format output in various machine-readable formats



Important: `--disclaimer` Option

`spectre-meltdown-checker.sh --disclaimer` provides important information about what the script does, and does not do.

8.2 Additional Information about Spectre/Meltdown

For more information, see the following references:

- SUSE Knowledge Base article #7022937, *Security Vulnerability: Spectre Variant 4 (Speculative Store Bypass) aka CVE-2018-3639*: <https://www.suse.com/support/kb/doc?id=7022937> ↗
- `speed47/spectre-meltdown-checker` source code on GitHub, with detailed references to relevant Common Vulnerabilities and Exposures (CVE): <https://github.com/speed47/spectre-meltdown-checker> ↗
- SUSE Blog article, *Meltdown and Spectre Performance*: <https://www.suse.com/c/meltdown-spectre-performance/> ↗
- SUSE Knowledge Base article #7022512, providing information on architectures, CVEs, and mitigations: <https://www.suse.com/support/kb/doc?id=7022512> ↗

9 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.

9.1 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.

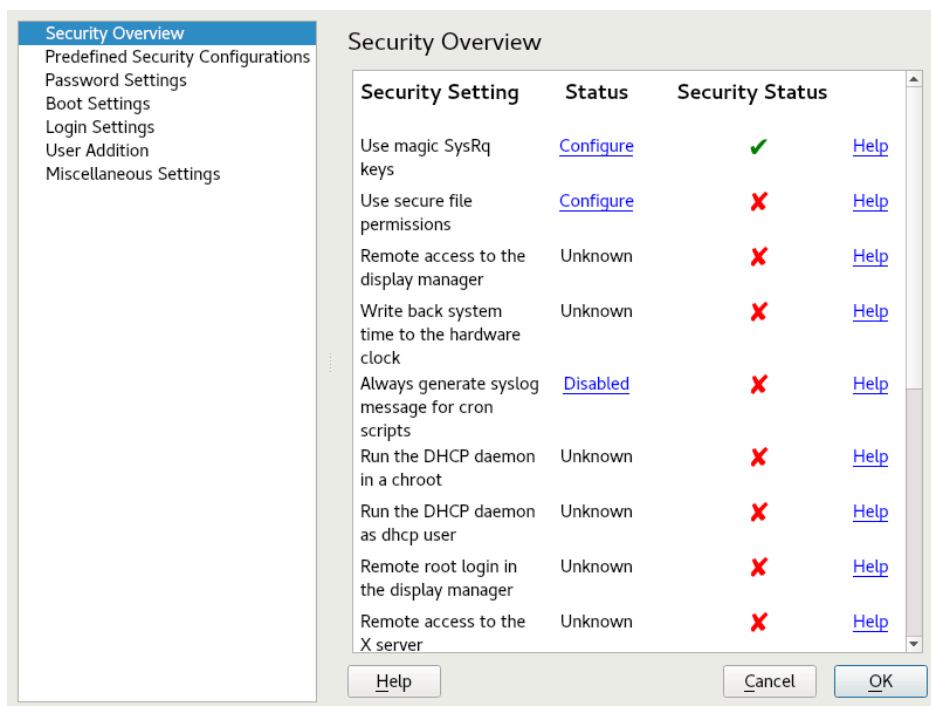


FIGURE 9.1: YAST SECURITY CENTER AND HARDENING: SECURITY OVERVIEW

9.2 *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*.

9.3 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.

9.4 *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.

9.5 *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.

9.6 *User Addition*

Set minimum and maximum values for user and group IDs. These default settings would rarely need to be changed.

9.7 *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> .

10 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.

10.1 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.

10.1.1 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.

10.1.2 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.

10.1.3 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 10.3, "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.

pktttyagent

Starts a textual authentication agent. This agent is used if a desktop environment does not have its own authentication agent.

10.1.4 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

10.2 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.

10.2.1 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.

10.2.2 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.

10.2.3 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 10.4.3, “Modifying Configuration Files for Implicit Privileges”](#).

10.3 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 10.1.3, “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`. 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 10.5, “Restoring the Default Privileges”](#).

10.4 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.

10.4.1 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> ❶

  <action id="org-opensuse-policykit-gparted"> ❷
    <message>Authentication is required to run the GParted Partition Editor</message>
    <icon_name>gparted</icon_name>
    <defaults> ❸
      <allow_any>auth_admin</allow_any>
      <allow_inactive>auth_admin</allow_inactive>
      <allow_active>auth_admin</allow_active>
    </defaults>
    <annotate ❹
      key="org.freedesktop.policykit.exec.path">/usr/sbin/gparted</annotate>
    <annotate ❹
      key="org.freedesktop.policykit.exec.allow_gui">>true</annotate>
    </action>

</policyconfig>

```

- ❶ Root element of the policy file.
- ❷ Contains one single action.
- ❸ 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.
- ❹ 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.

10.4.2 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 **gpated** to be executed by any member of the `admin` group, check for the action ID `org.opensuse.policykit.gpated`:

```
/* Allow users in admin group to run GParted without authentication */
polkit.addRule(function(action, subject) {
    if (action.id == "org.opensuse.policykit.gpated" &&
        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>.

10.4.3 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 10.2.3, "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**.

10.5 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 10.2.3, "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 10.1: 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
```

11 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.

11.1 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.

11.1.1 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`.

11.1.2 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.

11.1.3 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
```

11.2 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.

11.3 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.

11.4 Handling ACLs

Table 11.1, "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 11.2, "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 11.1: ACL ENTRY TYPES

Type	Text Form
owner	<u>user::rwx</u>
named user	<u>user:name:rwx</u>
owning group	<u>group::rwx</u>
named group	<u>group:name:rwx</u>
mask	<u>mask::rwx</u>
other	<u>other::rwx</u>

TABLE 11.2: MASKING ACCESS PERMISSIONS

Entry Type	Text Form	Permissions
named user	<u>user:geeko:r-x</u>	<u>r-x</u>
mask	<u>mask::rw-</u>	<u>rw-</u>
	effective permissions:	<u>r--</u>

11.4.1 ACL Entries and File Mode Permission Bits

Figure 11.1, “Minimum ACL: ACL Entries Compared to Permission Bits” and Figure 11.2, “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.

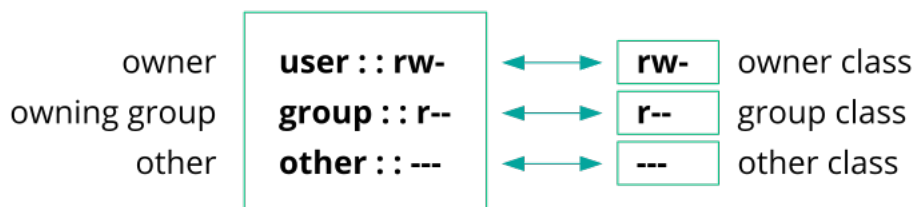


FIGURE 11.1: MINIMUM ACL: ACL ENTRIES COMPARED TO PERMISSION BITS

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 11.1, “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 11.2, “Extended ACL: ACL Entries Compared to Permission Bits”*.

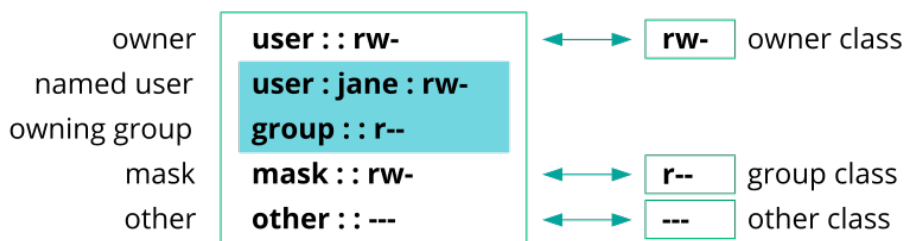


FIGURE 11.2: EXTENDED ACL: ACL ENTRIES COMPARED TO PERMISSION BITS

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.

11.4.2 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 deac-

tivate 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 11.2, “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`.

11.4.3 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.

11.4.3.1 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.

11.4.3.2 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 11.4.3.1, “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.

11.4.4 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”.

11.5 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 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.

11.6 For More Information

For more information about ACLs, see the man pages for `getfacl(1)`, `acl(5)`, and `setfacl(1)`.

12 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 12.1.1, “Creating an Encrypted Partition during Installation”](#) and [Section 12.1.2, “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 12.1.3, “Encrypting the Content of Removable Media”](#).

Encrypting Single Files with GPG

To quickly encrypt one or more files, you can use the GPG tool. See [Section 12.2, “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.

12.1 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.

12.1.1 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 12.1.2, “Creating an Encrypted Partition on a Running System”](#) and be aware that this action destroys all data on the existing partition.

12.1.2 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 12.1.1, “Creating an Encrypted Partition during Installation”](#).

12.1.3 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.

12.2 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.

13 Storage Encryption for Hosted Applications with `cryptctl`

Databases and similar applications are often hosted on external servers that are serviced by third-party staff. Certain data center maintenance tasks require third-party staff to directly access affected systems. In such cases, privacy requirements necessitate disk encryption.

`cryptctl` allows encrypting sensitive directories using LUKS and offers the following additional features:

- Encryption keys are located on a central server, which can be located on customer premises.
- Encrypted partitions are automatically remounted after an unplanned reboot.

`cryptctl` consists of two components:

- A client is a machine that has one or more encrypted partitions but does not permanently store the necessary key to decrypt those partitions. For example, clients can be cloud or otherwise hosted machines.
- The server holds encryption keys that can be requested by clients to unlock encrypted partitions.

You can also set up the `cryptctl` server to store encryption keys on a KMIP 1.3-compatible (Key Management Interoperability Protocol) server. In that case, the `cryptctl` server will not store the encryption keys of clients and is dependent upon the KMIP-compatible server to provide these.



Warning: `cryptctl` Server Maintenance

Since the `cryptctl` server manages timeouts for the encrypted disks and, depending on the configuration, can also hold encryption keys, it should be under your direct control and managed only by trusted personnel.

Additionally, it should be backed up regularly. Losing the server's data means losing access to encrypted partitions on the clients.

To handle encryption, `cryptctl` uses LUKS with aes-xts-256 encryption and 512-bit keys. Encryption keys are transferred using TLS with certificate verification.

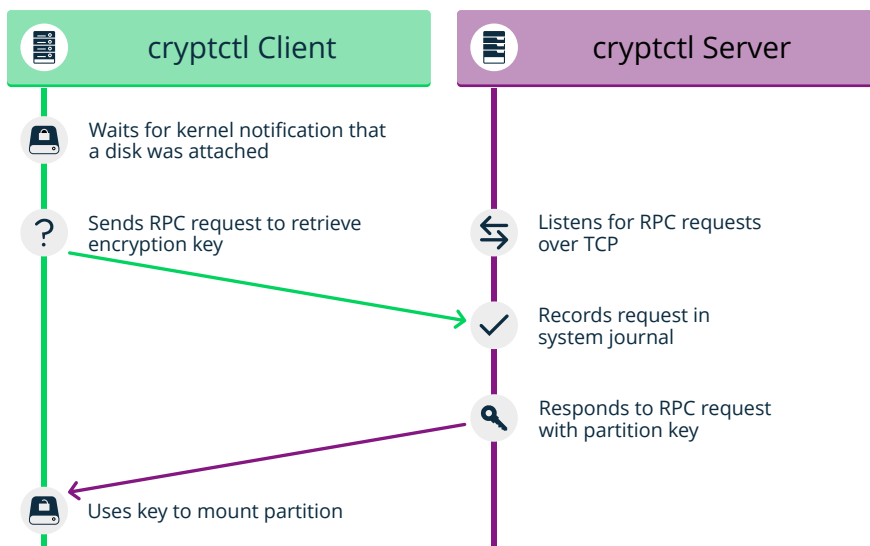


FIGURE 13.1: KEY RETRIEVAL WITH **cryptctl** (MODEL WITHOUT CONNECTION TO KMIP SERVER)

Note: Install **cryptctl**

Before continuing, make sure the package `cryptctl` is installed on all machines you intend to set up as servers or clients.

13.1 Setting Up a **cryptctl** Server

Before you can define a machine as a **cryptctl** client, you need to set up a machine as a **cryptctl** server.

Before beginning, choose whether to use a self-signed certificate to secure communication between the server and clients. If not, generate a TLS certificate for the server and have it signed by a certificate authority.

Additionally, you can have clients authenticate to the server using certificates signed by a certificate authority. To use this extra security measure, make sure to have a CA certificate at hand before starting this procedure.

1. As `root`, run:

```
root # cryptctl init-server
```

2. Answer each of the following prompts and press `Enter` after every answer. If there is a default answer, it is shown in square brackets at the end of the prompt.

- a. Choose a password with at least 10 characters and confirm it. This password assumes the role of a master password, able to unlock all partitions that are registered on the server.
- b. Specify the path to a PEM-encoded TLS certificate or certificate chain file or leave the field empty to create a self-signed certificate. If you specify a path, use an absolute path.
- c. If you want the server to be identified by a host name other than the default shown, specify a host name. `cryptctl` will then generate certificates which include the host name.
- d. Specify the IP address that belongs to the network interface that you want to listen on for decryption requests from the clients, then set a port number (the default is port 3737).
The default IP address setting, `0.0.0.0` means that `cryptctl` will listen on all network interfaces for client requests using IPv4.
- e. Specify a directory on the server that will hold the decryption keys for clients.
- f. Specify whether clients need to authenticate to the server using a TLS certificate. If you choose *No*, this means that clients authenticate using disk UUIDs only. (However, communication will be encrypted using the server certificate in any case.)
If you choose *Yes*, pick a PEM-encoded certificate authority to use for signing client certificates.
- g. Specify whether to use a KMIP 1.3-compatible server (or multiple such servers) to store encryption keys of clients. If you choose this option, provide the host names and ports for one or multiple KMIP-compatible servers.
Additionally, provide a user name, password, a CA certificate for the KMIP server, and a client identity certificate for the `cryptctl` server.

 **Important: No Easy Reconfiguration of KMIP Setting**

The setting to use a KMIP server cannot easily be changed later. To change this setting, both the `cryptctl` server and its clients need to be configured afresh.

- h. Finally, configure an SMTP server for e-mail notifications for encryption and decryption requests or leave the prompt empty to skip setting up e-mail notifications.



Note: Password-Protected Servers

cryptctl currently cannot send e-mail using authentication-protected SMTP servers. If that is necessary, set up a local SMTP proxy.

- i. When asked whether to start the **cryptctl** server, enter y.
3. To check the status of the service **cryptctl-server**, use:

```
root # systemctl status cryptctl-server
```

To reconfigure the server later, do either of the following:

- Run the command **cryptctl init-server** again. **cryptctl** will then propose the existing settings as the defaults, so that you only need to specify the values that you want to change.
- Make changes directly in the configuration file `/etc/sysconfig/cryptctl-server`. However, to avoid issues, do not change the settings `AUTH_PASSWORD_HASH` and `AUTH_PASSWORD_SALT` manually. The values of these options need to be calculated correctly.

13.2 Setting Up a **cryptctl** Client

The following interactive setup of **cryptctl** is currently the only setup method.

Make sure the following preconditions are fulfilled:

- A **cryptctl** server is available over the network.
- There is a directory to encrypt.
- The client machine has an empty partition available that is large enough to fit the directory to encrypt.
- When using a self-signed certificate, the certificate (`*.crt` file) generated on the server is available locally on the client. Otherwise, the certificate authority of the server certificate must be trusted by the client.
- If you set up the server to require clients to authenticate using a client certificate, prepare a TLS certificate for the client which is signed by the CA certificate you chose for the server.

1. As root, run:

```
root # cryptctl encrypt
```

2. Answer each of the following prompts and press `Enter` after every answer. If there is a default answer, it is shown in square brackets at the end of the prompt.
 - a. Specify the host name and port to connect to on the **cryptctl** server.
 - b. If you configured the server to have clients authenticate to it using a TLS certificate, specify a certificate and a key file for the client. The client certificate must be signed by the certificate authority chosen when setting up the server.
 - c. Specify the absolute path to the server certificate (the `*.crt` file).
 - d. Enter the encryption password that you specified when setting up the server.
 - e. Specify the path to the directory to encrypt. Specify the path to the empty partition that will contain the encrypted content of the directory.
 - f. Specify the number of machines that are allowed to decrypt the partition simultaneously.

Then specify the timeout in seconds before additional machines are allowed to decrypt the partition after the last vital sign was received from the client or clients. When a machine unexpectedly stops working and then reboots, it needs to be able to unlock its partitions again. That means this timeout should be set to a time slightly shorter than the reboot time of the client.

Important: Timeout Length

If the time is set too long, the machine cannot decrypt encrypted partitions on the first try. **cryptctl** will then continue to periodically check whether the encryption key has become available. However, this will introduce a delay.

If the timeout is set too short, machines with a copy of the encrypted partition have an increased chance of unlocking the partition first.

3. To start encryption, enter yes.

cryptctl will now encrypt the specified directory to the previously empty partition and then mount the newly encrypted partition. The file system type will be of the same type as the original unencrypted file system.

Before creating the encrypted partition, **cryptctl** moves the unencrypted content of the original directory to a location prefixed with **cryptctl-moved-**.

4. To check that the directory is indeed mounted correctly, use:

```
tux > lsblk -o NAME,MOUNTPOINT,UUID
NAME                                MOUNTPOINT          UUID
[...]
sdc
└─sdc1                               PARTITION_UUID
   └─cryptctl-unlocked-sdc1 /secret-partition  UNLOCKED_UUID
```

cryptctl identifies the encrypted partition by its UUID. For the previous example, that is the UUID displayed next to **sdc1**.

On the server, you can check whether the directory was decrypted using **cryptctl**.

```
root # cryptctl list-keys
```

For a successfully decrypted partition, you will see output like:

```
2019/06/06 15:50:00 ReloadDB: successfully loaded database of 1 records
Total: 1 records (date and time are in zone EDT)
Used By      When                UUID  Max.Users  Num.Users  Mount Point
IP_ADDRESS  2019-06-06 15:00:50  UUID  1          1          /secret-partition
```

For a partition not decrypted successfully, you will see output like:

```
2019/06/06 15:50:00 ReloadDB: successfully loaded database of 1 records
Total: 1 records (date and time are in zone EDT)
Used By      When                UUID  Max.Users  Num.Users  Mount Point
                2019-06-06 15:00:50  UUID  1          1          /secret-partition
```

See the difference in the empty **Used by** column.

Verify that the UUID shown is that of the previously encrypted partition.

5. After verifying that the encrypted partition works, delete the unencrypted content from the client. For example, use **rm**. For more safety, overwrite the content of the files before deleting them, for example, using **shred -u**.

! Important: **shred** Does Not Guarantee That Data Is Completely Erased

Depending on the type of storage media, using **shred** is not a guarantee that all data is completely removed. In particular, SSDs usually employ wear leveling strategies that render **shred** ineffective.

The configuration for the connection from client to server is stored in `/etc/sysconfig/cryptctl-client` and can be edited manually.

The server stores an encryption key for the client partition in `/var/lib/cryptctl/key-db/PARTITION_UUID`.

13.3 Checking Partition Unlock Status Using Server-side Commands

When a **cryptctl** client is active, it will send a “heartbeat” to the **cryptctl** server every 10 seconds. If the server does not receive a heartbeat from the client for the length of the timeout configured during the client setup, the server will assume that the client is offline. It will then allow another client to connect (or allow the same client to reconnect after a reboot).

To see the usage status of all keys, use:

```
root # cryptctl list-keys
```

The information under `Num.` `Users` shows whether the key is currently in use. To see more detail on a single key, use:

```
root # cryptctl show-key UUID
```

This command will show information about mount point, mount options, usage options, the last retrieval of the key and the last three heartbeats from clients.

Additionally, you can use **journalctl** to find logs of when keys were retrieved.

13.4 Unlocking Encrypted Partitions Manually

There are two ways of unlocking a partition manually, both of which are run on a client:

- **Online Unlocking.** Online unlocking allows circumventing timeout or user limitations. This method can be used when there is a network connection between client and server but the client could not (yet) unlock the partition automatically. This method will unlock all encrypted partitions on a machine.

To use it, run `cryptctl online-unlock`. Be prepared to enter the password specified when setting up the server.

- **Offline Unlocking.** This method can be used when a client cannot or must not be brought online to communicate with its server. The encryption key from the server must still be available. This method is meant as a last resort only and can only unlock a single partition at a time.

To use it, run `cryptctl offline-unlock`. The server's key file for the requisite partition (`/var/lib/cryptctl/keydb/PARTITION_UUID`) needs to be available on the client.

13.5 Maintenance Downtime Procedure

To ensure that partitions cannot be decrypted during a maintenance downtime, turn off the client and disable the `cryptctl` server. You can do so by either:

- Stopping the service `cryptctl-server`:

```
root # systemctl stop cryptctl-server
```

- Unplugging the `cryptctl` server from the network.

13.6 For More Information

For more information, also see the project home page <https://github.com/HouzuoGuo/cryptctl/>.

14 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.

14.1 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 `~/.profile`
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/`.

14.2 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.

15 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.

15.1 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.

15.2 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 15.1: 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.
S	Check for a changed file size.

Option	Description
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.

15.3 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 15.2, "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.

15.4 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 15.1: 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.

15.5 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

- 16 X Window System and X Authentication **148**
- 17 SSH: Secure Network Operations **149**
- 18 Masquerading and Firewalls **161**
- 19 Configuring a VPN Server **178**

16 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 cannot 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 17, 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.

17 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.

17.1 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"
```

17.1.1 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.

17.1.2 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 17.5.2, "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`.

17.2 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 ~ 2
```

- 1 local to remote
- 2 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.

17.3 `sftp`—Secure File Transfer

17.3.1 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'
[...]
```

17.3.2 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 17.1:

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
```

17.4 The SSH Daemon (sshd)

To work with the SSH client programs `ssh` and `sftp`, 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
```

17.4.1 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`.

17.4.2 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)

17.5 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` 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`.

17.5.1 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`.

17.5.2 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`.

17.5.3 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.

17.5.3.1 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 "$SSH_AUTH_SOCKET" -a -x "$SSH_ASKPASS"; then  
#     ssh-add < /dev/null  
# fi
```

4. When starting a new X session, you will be prompted for your SSH passphrase.

17.5.3.2 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.

17.6 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](#).

17.7 Adding and Removing Public Keys on an Installed System

In some environments, 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.

17.8 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.

18 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`. openSUSE Leap 15.0 introduces `firewalld` as the new default software firewall, replacing `SuSEfirewall2`. `SuSEfirewall2` has not been removed from openSUSE Leap 15.0 and is still part of the main repository, though not installed by default. This chapter provides guidance for configuring `firewalld`, and migrating from `SuSEfirewall2` for users who have upgraded from older openSUSE Leap releases.

18.1 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 18.1, "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.

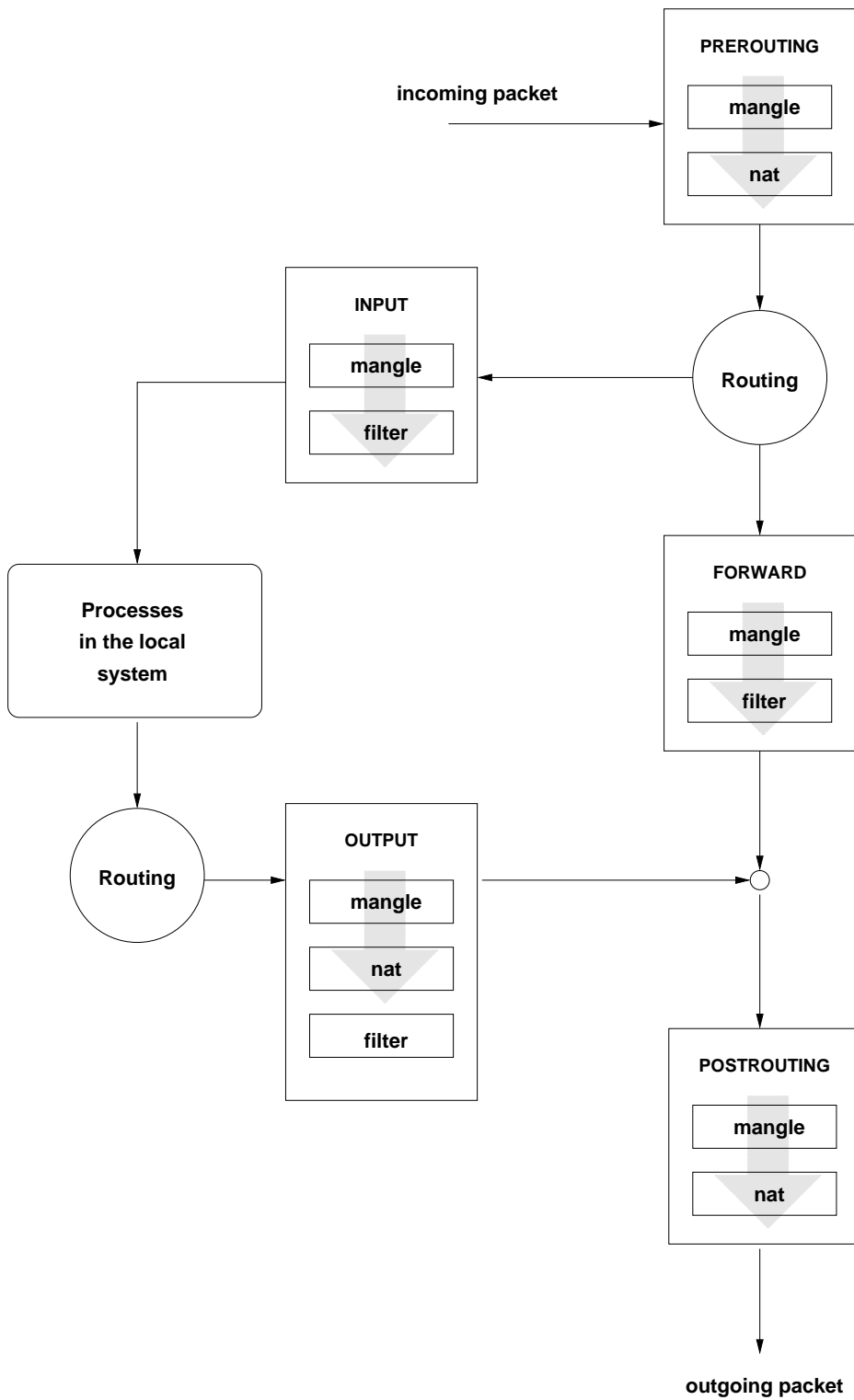


FIGURE 18.1: IPTABLES: A PACKET'S POSSIBLE PATHS

18.2 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.

18.3 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 \(http://www.tldp.org/HOWTO/Firewall-HOWTO.html\)](http://www.tldp.org/HOWTO/Firewall-HOWTO.html) ↗.

18.4 firewalld




Note: firewalld Replaces SuSEfirewall2

openSUSE Leap 15.0 introduces firewalld as the new default software firewall, replacing SuSEfirewall2. SuSEfirewall2 has not been removed from openSUSE Leap 15.0 and is still part of the main repository, though not installed by default. If you are upgrading from a release older than openSUSE Leap 15.0, SuSEfirewall2 will be unchanged and you must manually upgrade to firewalld (see [Section 18.5, “Migrating From SuSEfirewall2”](#)).

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. Several 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 are editing. How this is done is discussed in [Section 18.4.1.2, “Runtime Versus Permanent Configuration”](#).

To perform the `firewalld` configuration using the graphical user interface `firewall-config` refer to its [documentation \(http://www.firewalld.org/documentation/utilities/firewall-config.html\)](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.

18.4.1 Configuring the Firewall on the Command Line

18.4.1.1 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.

18.4.1.2 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 are 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 to make them permanent.

18.4.1.3 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.

18.4.1.4 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
```

18.4.1.5 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>.

18.4.1.6 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'
```

18.4.1.7 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 18.2, “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
```

18.4.2 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 18.1, “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 18.4.2.1, “Configuring Static Ports”](#). Alternatively, openSUSE Leap provides a helper script. For details, see [Section 18.4.2.2, “Using `firewall-rpcbind-helper` for Configuring Static Ports”](#).

18.4.2.1 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 18.1, “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 18.1: IMPORTANT SYSCONFIG VARIABLES FOR STATIC PORT CONFIGURATION

File Path	Variable Name	Example Value
<u>/etc/sysconfig/nfs</u>	MOUNT-D_PORT	21001
	STATD_PORT	21002
	LOCKD_TCP-PORT	21003
	LOCKD_UDP-PORT	21003
	RQUO-TAD_PORT	21004
<u>/etc/sysconfig/ypbind</u>	YPBIND_OPTIONS	-p 24500
<u>/etc/sysconfig/ypserv</u>	YPXFRD_ARGS	-p 24501
	YPSERV_ARGS	-p 24502

File Path	Variable Name	Example Value
	YPPASSWD-D_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 18.1, "Callback Port Configuration for the nfs Kernel Module in /etc/modprobe.d/60-nfs.conf"*.

EXAMPLE 18.1: 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 18.2, "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 18.2: 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

```

18.4.2.2 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`.

18.5 Migrating From SuSEfirewall2



Note: Creating a `firewalld` Configuration for AutoYaST

See the *Firewall Configuration* section of the *AutoYaST Guide* to learn how to create a `firewalld` configuration for AutoYaST.

When upgrading from any version of openSUSE Leap before 15.0 to openSUSE Leap 15.2, SuSEfirewall2 is not changed and remains active. There is no automatic migration, so you must migrate to `firewalld` manually. `firewalld` includes a helper migration script, `susefirewall2-to-firewalld`. Depending on the complexity of your SuSEfirewall2 configuration the script may perform a perfect migration, or it may fail. Most likely it will partially succeed and you will have to review your new `firewalld` configuration and make adjustments.

The resulting configuration will make `firewalld` behave somewhat like SuSEfirewall2. To take full advantage of `firewalld`'s features you may elect to create a new configuration, rather than trying to migrate your old configuration. It is safe to run the `susefirewall2-to-firewalld` script with no options, as it makes no permanent changes to your system. However, if you are administering the system remotely you could get locked out.

Install and run `susefirewall2-to-firewalld`:

```
root # zypper in susefirewall2-to-firewalld
root # susefirewall2-to-firewalld
INFO: Reading the /etc/sysconfig/SuSEfirewall2 file
INFO: Ensuring all firewall services are in a well-known state.
INFO: This will start/stop/restart firewall services and it's likely
INFO: to cause network disruption.
INFO: If you do not wish for this to happen, please stop the script now!
5...4...3...2...1...Lets do it!
INFO: Stopping firewalld
INFO: Restarting SuSEfirewall2_init
INFO: Restarting SuSEfirewall2
INFO: DIRECT: Adding direct rule="ipv4 -t filter -A INPUT -p udp -m udp --dport 5353 -m
pktttype
--pkt-type multicast -j ACCEPT"
[...]
INFO: Enabling direct rule=ipv6 -t filter -A INPUT -p udp -m udp --dport 546 -j ACCEPT
INFO: Enabling direct rule=ipv6 -t filter -A INPUT -p udp -m udp --dport 5353 -m pktttype
--pkt-type multicast -j ACCEPT
INFO: Enable logging for denied packets
INFO: #####
INFO:
INFO: The dry-run has been completed. Please check the above output to ensure
INFO: that everything looks good.
INFO:
INFO: #####
INFO: Stopping firewalld
INFO: Restarting SuSEfirewall2_init
INFO: Restarting SuSEfirewall2
```

This results in a lot of output, which you may wish to direct to a file for easier review:

```
root # susefirewall2-to-firewalld | tee newfirewallrules.txt
```

The script supports these options:

-c

Commit changes. The script will make changes to the system so make sure you only use this option if you are really happy with the proposed changes. This **will** reset your current `firewalld` configuration so make sure you make backups!

-d

Super noisy. Use it to file bug reports but be careful to mask sensitive information.

-h

This message.

-q

No output. Errors will not be printed either!

-v

Verbose mode. It will print warnings and other informative messages.

18.6 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.

19 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 \(http://www.openvpn.net\)](http://www.openvpn.net) implementation to link branch offices via secure wide area networks (WANs).

19.1 Conceptual Overview

This section defines some terms regarding VPN and gives a brief overview of some scenarios.

19.1.1 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 Ethernet frames). 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 IP packets). 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.

19.1.2 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 19.1, "Routed VPN"](#).

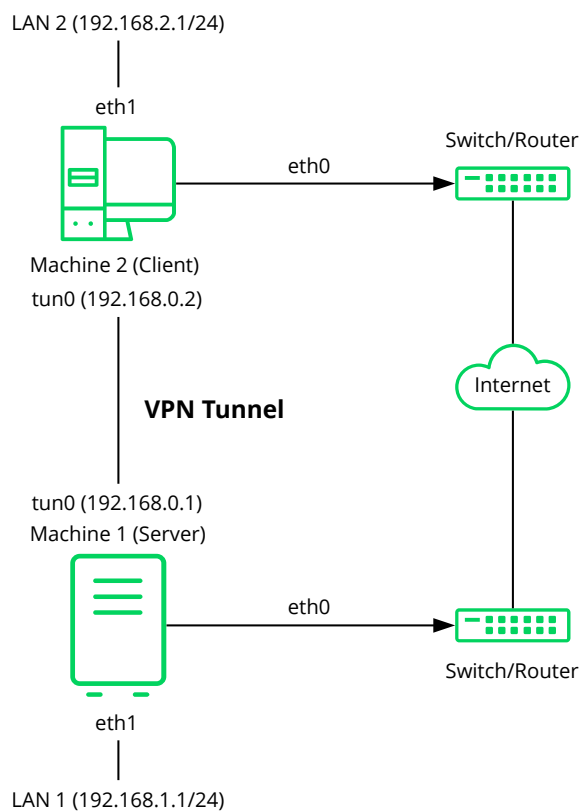


FIGURE 19.1: ROUTED VPN

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.

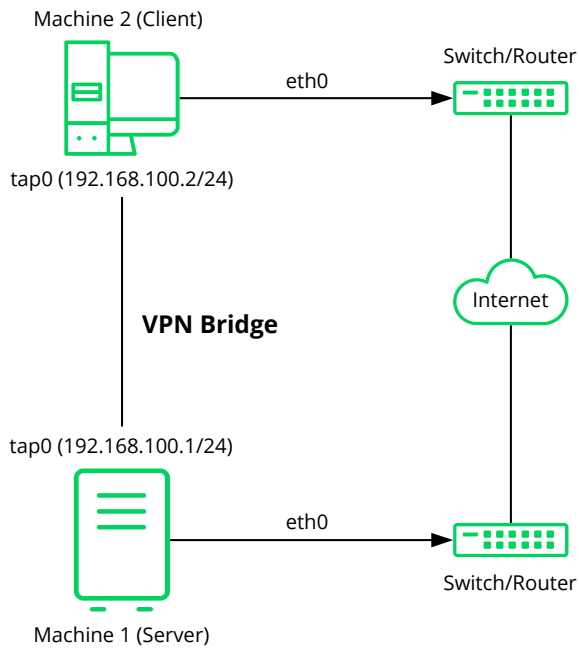


FIGURE 19.2: BRIDGED VPN - SCENARIO 1

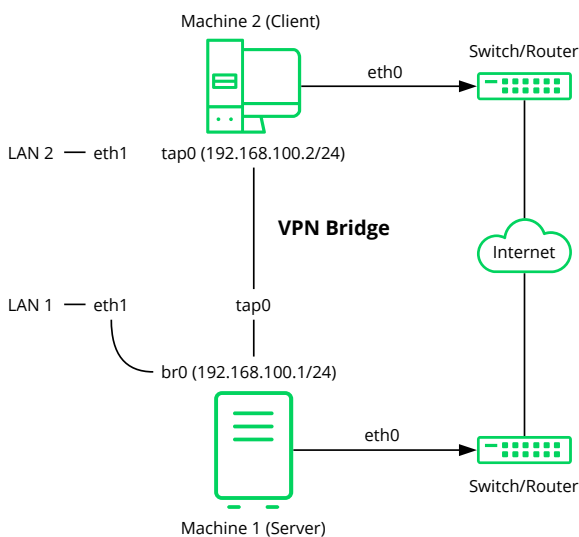


FIGURE 19.3: BRIDGED VPN - SCENARIO 2

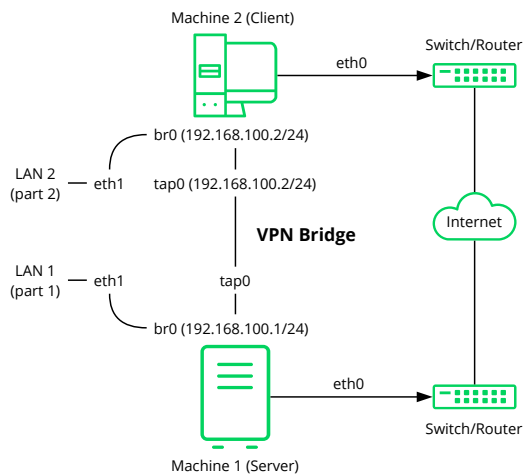


FIGURE 19.4: BRIDGED VPN - SCENARIO 3

The major difference between bridging and routing is that a routed VPN cannot IP-broadcast while a bridged VPN can.

19.2 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.

19.2.1 Configuring the VPN Server

To configure a VPN server, proceed as follows:

PROCEDURE 19.1: 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
```

19.2.2 Configuring the VPN Clients

To configure the VPN client, do the following:

PROCEDURE 19.2: 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 19.1, "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
```

19.2.3 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
```

19.3 Setting Up Your VPN Server Using a Certificate Authority

The example in *Section 19.2* 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 19.3.1, "Creating Certificates"*
2. *Section 19.3.2, "Configuring the VPN Server"*
3. *Section 19.3.3, "Configuring the VPN Clients"*

19.3.1 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*.

Creating certificates is not supported on openSUSE Leap. The following assumes you have created a CA certificate, a server certificate and a client certificate on another system.

The server certificate is required in the PEM and unencrypted key in PEM formats. Copy the PEM version to `/etc/openvpn/server_cert.pem` on the VPN server. The unencrypted version needs to go to `/etc/openvpn/server_key.pem`.

Client certificates need to be of the format PKCS12 (preferred) or PEM. The certificate in PKCS12 format needs to contain the CA chain and needs to be copied to `/etc/openvpn/CLIENT.p12`. In case you have client certificates in PEM format containing the CA chain, copy them to `/etc/openvpn/CLIENT.pem`. In case you have split the PEM certificates into client certificate (`*.ca`), client key (`*.key`), and the CA certificate (`*.ca`), copy these files to `/etc/openvpn/` on each client.

The CA certificate needs to be copied to `/etc/openvpn/vpn_ca.pem` on the server and each client.

Important: Splitting Client Certificates

If you split client certificates into client certificate, client key, and the CA certificate, you need to provide the respective filenames in the OpenVPN configuration file on the respective clients (see *Example 19.1, "VPN Server Configuration File"*).

19.3.2 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 19.1: VPN SERVER CONFIGURATION FILE

```
# /etc/openvpn/server.conf
port 1194 ①
proto udp ②
dev tun0 ③

# Security ④

ca    vpn_ca.pem
cert  server_cert.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 18, 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 19.1.1, “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 19.3.1, “Creating Certificates”](#).
- ⑤ Require that peer certificates have been signed with an explicit key usage and extended key usage based on RFC3280 TLS rules.
- 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 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.

19.3.3 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 19.2: 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 ⑥
user nobody
group nobody

# Try to preserve some state across restarts.
persist-key
persist-tun

# Security ⑦
pkcs12 client1.p12

comp-lzo ⑧
```

- ① Specifies that this machine is a client.
- ② The network device. Both clients and server must use the same device.
- ③ The protocol. Use the same settings as on the server.
- ⑤ This is security option for clients which ensures that the host they connect to is a designated server.
- ④ 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.
- ⑥ For security reasons, run the OpenVPN daemon with reduced privileges. To do so, specify that it should use the group and user `nobody`.
- ⑦ Contains the client files. For security reasons, use a separate pair of files for each client.

- 8 Turn on compression. Only use this parameter if compression is enabled on the server as well.

19.4 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 19.3: 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.

19.5 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.

IV Confining Privileges with AppArmor

- 20 Introducing AppArmor [192](#)
- 21 Getting Started [194](#)
- 22 Immunizing Programs [199](#)
- 23 Profile Components and Syntax [208](#)
- 24 AppArmor Profile Repositories [239](#)
- 25 Building and Managing Profiles with YaST [240](#)
- 26 Building Profiles from the Command Line [250](#)
- 27 Profiling Your Web Applications Using ChangeHat [278](#)
- 28 Confining Users with pam_apparmor [289](#)
- 29 Managing Profiled Applications [290](#)
- 30 Support [292](#)
- 31 AppArmor Glossary [301](#)

20 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.

20.1 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.

20.2 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 ↗.

21 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 21.3, “Choosing Applications to Profile”*.
2. Build the needed profiles as roughly outlined in *Section 21.4, “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 21.5, “Updating Your Profiles”*.

21.1 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
```

21.2 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.

21.3 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 21.1: 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 22.2](#), “Determining Programs to Immunize”.

21.4 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 26.7.3.2, "aa-complain—Entering Complain or Learning Mode"* and *Section 26.7.3.6, "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 26.7.3.9, “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 23, Profile Components and Syntax](#), [Chapter 25, Building and Managing Profiles with YaST](#), and [Chapter 26, Building Profiles from the Command Line](#).

21.5 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 26.7.3.9, “aa-logprof—Scanning the System Log”](#).

22 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 23, Profile Components and Syntax](#), [Chapter 25, Building and Managing Profiles with YaST](#), or [Chapter 26, 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 20.2, “Background Information on AppArmor Profiling”](#).

22.1 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 23.1, “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 26.7.3.12, “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 26.7.3.1, “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 26.7.3.8, "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 26.7.3.9, "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 26.7.3.2, "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 26.7.3.6, "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 26.7.3.13, “aa-notify”*.

22.2 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 22.3, “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 22.4.1, “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 22.4.2, “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.

22.3 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`.

22.4 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' (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.

22.4.1 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 27.2, “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 29, 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 27, Profiling Your Web Applications Using ChangeHat](#).

22.4.2 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.

23 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 22.2, “Determining Programs to Immunize”](#). To start building AppArmor profiles with YaST, proceed to [Chapter 25, Building and Managing Profiles with YaST](#). To build profiles using the AppArmor command line interface, proceed to [Chapter 26, Building Profiles from the Command Line](#).

For more details about creating AppArmor profiles, see [man 5 apparmor](#).

23.1 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/foo ❷ { ❸
    #include <abstractions/base> ❹

    capability setgid ❺,
    network inet tcp ❻,

    link /etc/sysconfig/foo -> /etc/foo.conf, ❼
    /bin/mount          ux,
    /dev/{,u} ❽ random  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,
    owner ❾ /shared/foo/** rw,
    /usr/bin/foobar     Cx, ❿
    /bin/**            Px -> bin_generic, ❿

    # 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 23.5, "Network Access Control"](#).
- ⑦ A link pair rule specifying the source and the target of a link. See [Section 23.7.6, "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 23.7, "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 23.7.8, "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 23.12, “Execute Modes”](#).
- 13 A named profile transition to the profile `bin_generic` located in the global scope. See [Section 23.12.7, “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 27, 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.

23.2 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.

23.2.1 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`.

23.2.2 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 23.12.7, “Named Profile Transitions”](#)) or with the `change_profile` rule (see [Section 23.2.5, “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.

23.2.3 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 23.12.2, "Discrete Local Profile Execute Mode \(Cx\)"](#)) or a named profile transition (see [Section 23.12.7, "Named Profile Transitions"](#)).

23.2.4 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 27, Profiling Your Web Applications Using ChangeHat](#) for a detailed description.

23.2.5 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.

23.3 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.

23.3.1 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](#).

23.3.2 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.

23.3.3 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](#).

23.4 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
```

23.5 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> ①][<type> ②][<protocol> ③]]
```

- ① Supported domains: inet, ax25, ipx, appletalk, netrom, bridge, x25, inet6, rose, netbeui, security, key, packet, ash, econet, atmsvc, sna, pppox, wan-pipe, bluetooth, unix, atmpvc, netlink, llc, can, tipc, iucv, rxrpc, isdn, phonet, ieee802154, caif, alg, nfc, vsock
- ② Supported types: stream, dgram, seqpacket, rdm, raw, packet
- ③ 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.

```
network ① ,  
network inet ② ,  
network inet6 ③ ,  
network inet stream ④ ,  
network inet tcp ⑤ ,  
network tcp ⑥ ,
```

- ① Allow all networking. No restrictions applied with regard to domain, type, or protocol.
- ② Allow general use of IPv4 networking.
- ③ Allow general use of IPv6 networking.
- ④ Allow the use of IPv4 TCP networking.
- ⑤ Allow the use of IPv4 TCP networking, paraphrasing the rule above.
- ⑥ Allow the use of both IPv4 and IPv6 TCP networking.

23.6 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 23.2.1, "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.

<u>*</u>	Substitutes for any number of any characters, except <u>/</u> . Example: An arbitrary number of file path elements.
----------	--

<u>**</u>	Substitutes for any number of characters, including <u>/</u> . Example: An arbitrary number of path elements, including entire directories.
<u>?</u>	Substitutes for any single character, except <u>/</u> .
<u>[abc]</u>	Substitutes for the single character <u>a</u> , <u>b</u> , or <u>c</u> . Example: a rule that matches <u>/home[01]/*/.plan</u> allows a program to access <u>.plan</u> files for users in both <u>/home0</u> and <u>/home1</u> .
<u>[a-c]</u>	Substitutes for the single character <u>a</u> , <u>b</u> , or <u>c</u> .
<u>{ab,cd}</u>	Expands to one rule to match <u>ab</u> and one rule to match <u>cd</u> . Example: a rule that matches <u>{usr,www}/pages/**</u> grants access to Web pages in both <u>/usr/pages</u> and <u>/www/pages</u> .
<u>[^a]</u>	Substitutes for any character except <u>a</u> .

23.6.1 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).

23.6.2 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.

23.6.3 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`.

23.6.4 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
```

23.6.5 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 23.6.3, "Pattern Matching"](#)). For example, the default profile

```
/** { ... }
```

can be named

```
profile default /** { ... }
```

Also, a profile with pattern matching can be named. For example:

```
/usr/lib64/firefox*/firefox-*bin { ... }
```

can be named

```
profile firefox /usr/lib64/firefox*/firefox-*bin { ... }
```


23.6.6 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.

23.7 File Permission Access Modes

File permission access modes consist of combinations of the following modes:

<u>r</u>	Read mode
----------	-----------

<u>w</u>	Write mode (mutually exclusive to <u>a</u>)
<u>a</u>	Append mode (mutually exclusive to <u>w</u>)
<u>k</u>	File locking mode
<u>l</u>	Link mode
<u>link FILE -> TARGET</u>	Link pair rule (cannot be combined with other access modes)

23.7.1 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.

23.7.2 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).

23.7.3 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.

23.7.4 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.

23.7.5 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).

23.7.6 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.

23.7.7 Optional allow and file Rules

The allow prefix is optional, and it is idiomatically implied if not specified and the deny (see [Section 23.7.9, “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
```

23.7.8 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,
```

23.7.9 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.

23.8 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
```

23.9 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,
```

23.10 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,
```

23.11 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),

```

23.12 Execute Modes

Execute modes, also named profile transitions, consist of the following modes:

<u>Px</u>	Discrete profile execute mode
<u>Cx</u>	Discrete local profile execute mode
<u>Ux</u>	Unconfined execute mode
<u>ix</u>	Inherit execute mode
<u>m</u>	Allow <code>PROT_EXEC</code> with <code>mmap(2)</code> calls

23.12.1 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.

23.12.2 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.

23.12.3 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.

23.12.4 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.

23.12.5 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.

23.12.6 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).

23.12.7 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.

23.12.8 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.

23.12.9 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

23.12.10 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
```

23.13 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.

23.14 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,
```


24 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.

25 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 26, 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.

25.1 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 25.2.1, "Adding an Entry"*, *Section 25.2.2, "Editing an Entry"*, or *Section 25.2.3, "Deleting an Entry"*.
5. When finished, click *Done*.

25.2 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
```

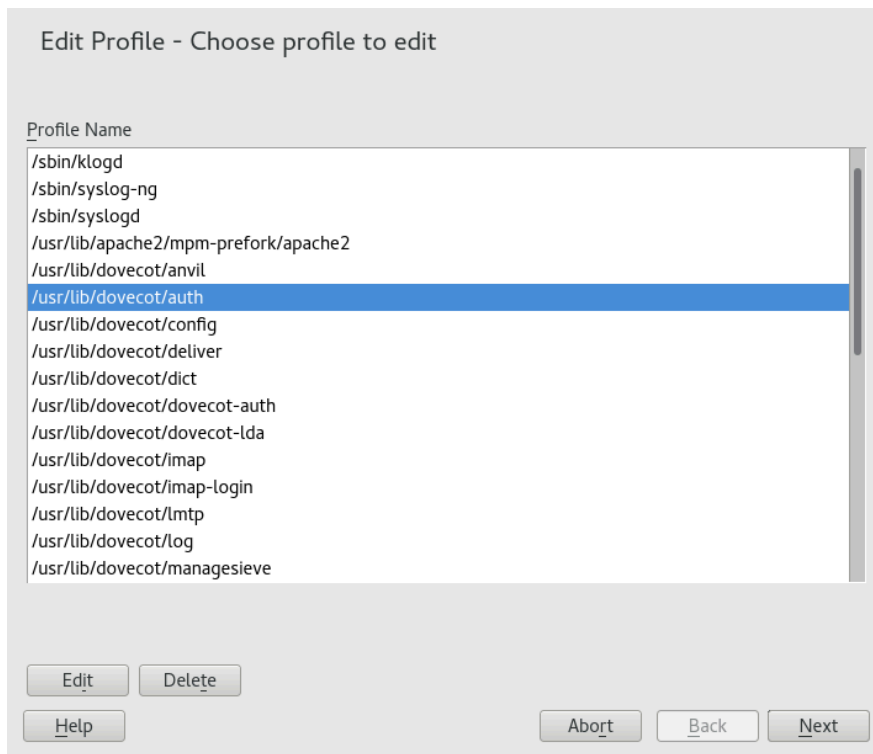


Tip

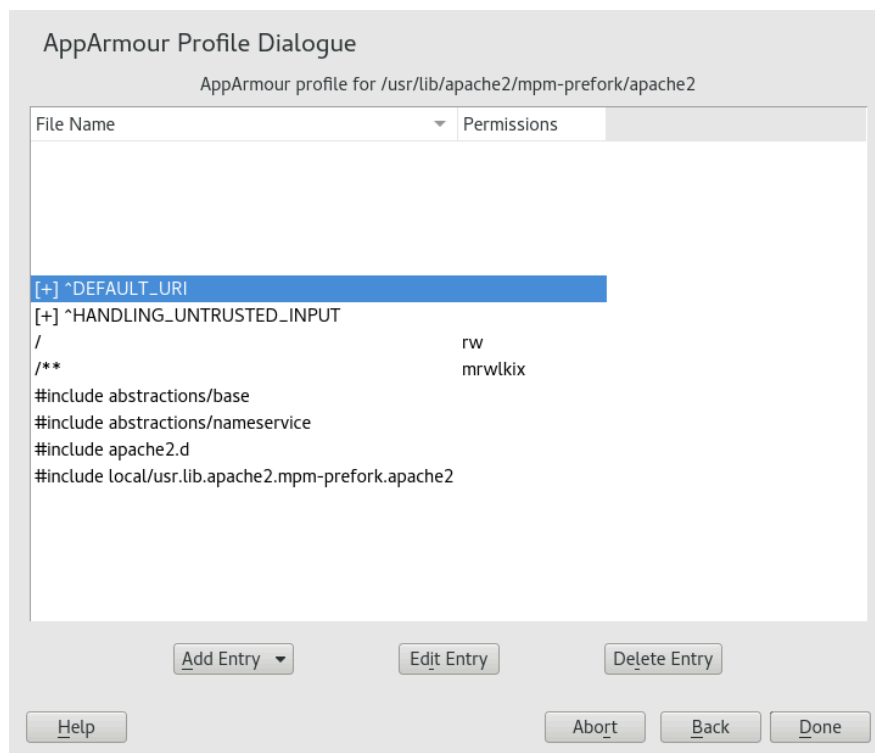
The vi editor also includes syntax (error) highlighting and syntax error highlighting, which visually warns you when the syntax of the edited AppArmor profile is wrong.

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 25.2.1, “Adding an Entry”](#), [Section 25.2.2, “Editing an Entry”](#), or [Section 25.2.3, “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**.

25.2.1 Adding an Entry

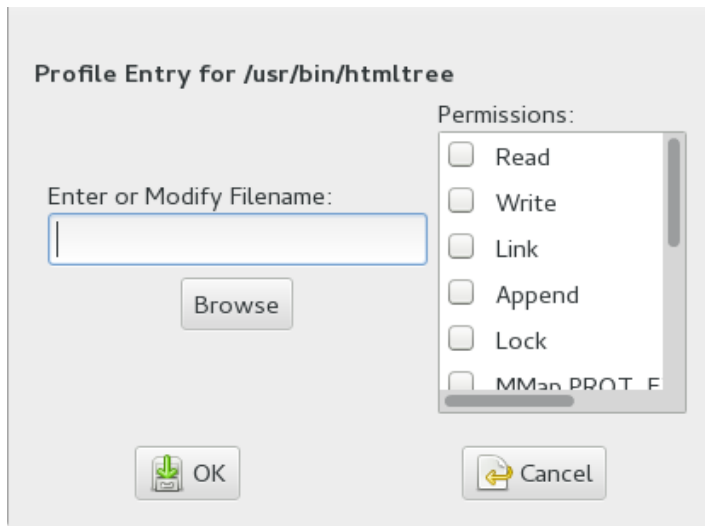
The *Add Entry* button in the *AppArmor Profile Window* lists types of entries you can add to the AppArmor profile.

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*.

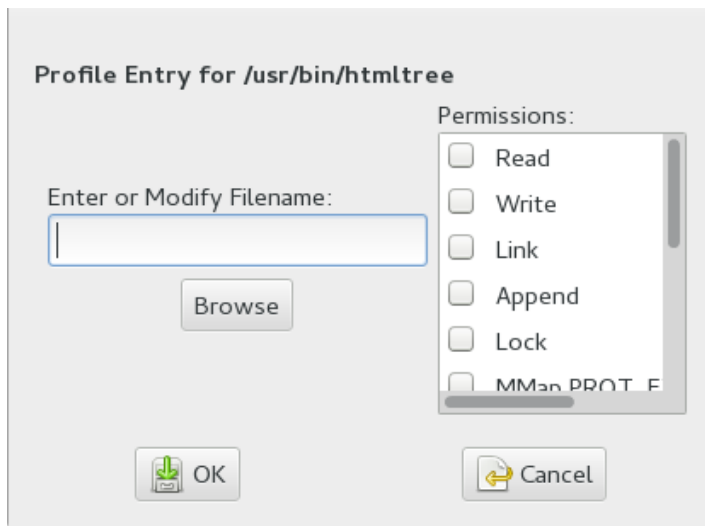
You can use globbing if necessary. For globbing information, refer to [Section 23.6, "Profile Names, Flags, Paths, and Globbing"](#). For file access permission information, refer to [Section 23.7, "File Permission Access Modes"](#).



Directory

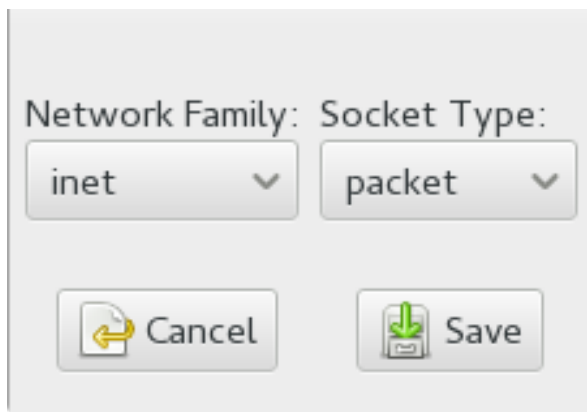
In the pop-up window, specify the absolute path of a directory, including the type of access permitted. You can use globbing if necessary. When finished, click *OK*.

For globbing information, refer to [Section 23.6, "Profile Names, Flags, Paths, and Globbing"](#). For file access permission information, refer to [Section 23.7, "File Permission Access Modes"](#).



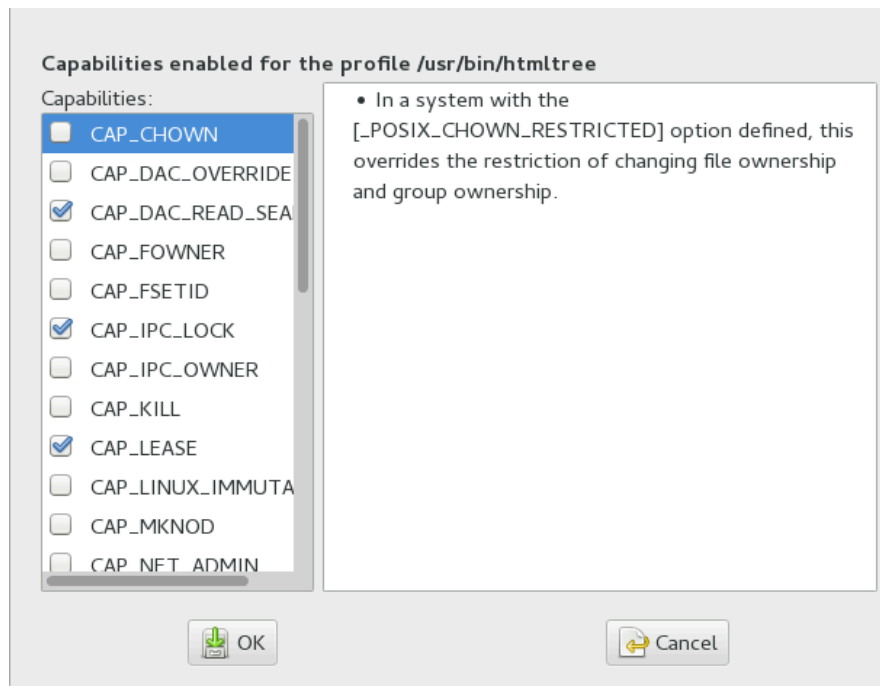
Network Rule

In the pop-up window, select the appropriate network family and the socket type. For more information, refer to [Section 23.5, "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 23.4, "Capability Entries \(POSIX.1e\)"](#) for more information about capabilities. When finished making your selections, click *OK*.

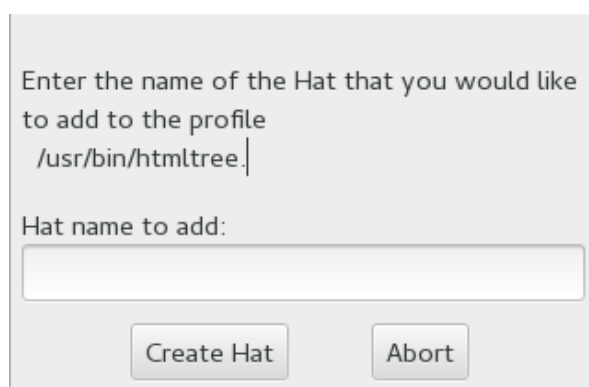


Include File

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 23.3, "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 27, Profiling Your Web Applications Using ChangeHat](#).



25.2.2 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 23.6, “Profile Names, Flags, Paths, and Globbing”](#). For access permission information, refer to [Section 23.7, “File Permission Access Modes”](#).

25.2.3 Deleting an Entry

To delete an entry in a given profile, select *Delete Entry*. AppArmor removes the selected profile entry.

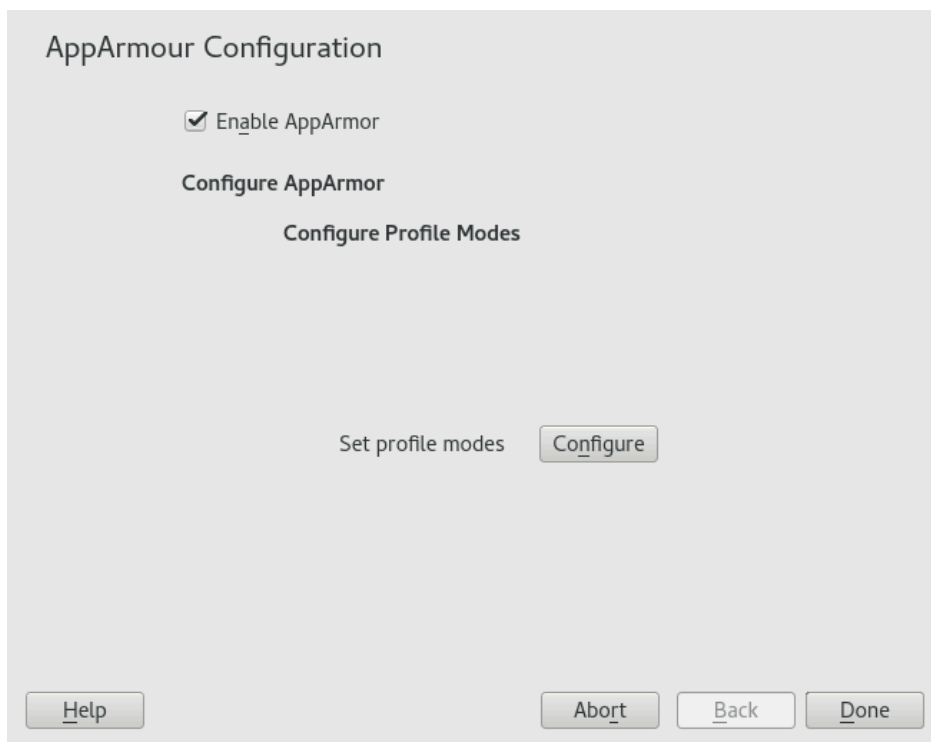
25.3 Deleting a Profile

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.

25.4 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 25.4.1, “Changing AppArmor Status”](#). To change the mode of individual profiles, continue as described in [Section 25.4.2, “Changing the Mode of Individual Profiles”](#).

25.4.1 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.

25.4.2 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 26.7.2, “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.

26 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 22, Immunizing Programs* and *Chapter 23, Profile Components and Syntax*.

26.1 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.

26.2 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 23, 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 26.3, “Adding or Creating an AppArmor Profile”](#)

Editing AppArmor Profiles

Refer to [Section 26.4, “Editing an AppArmor Profile”](#)

Deleting AppArmor Profiles

Refer to [Section 26.6, “Deleting an AppArmor Profile”](#)

26.3 Adding or Creating an AppArmor Profile

To add or create an AppArmor profile for an application, you can use a systemic or stand-alone profiling method, depending on your needs. Learn more about these two approaches in *Section 26.7, “Two Methods of Profiling”*.

26.4 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.

26.5 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
```

26.6 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
```

26.7 Two Methods of Profiling

Given the syntax for AppArmor profiles in *Chapter 23, 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 26.7.1, "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 26.7.2, "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.

26.7.1 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 26.7.3.8, “aa-genprof—Generating Profiles”](#).

26.7.2 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 26.7.3.1, “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 25.4.2, “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 26.7.3.2, “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 26.7.3.9, “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 25.4.2, “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`.

26.7.3 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.

26.7.3.1 `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.

26.7.3.2 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 25.4.2, “Changing the Mode of Individual Profiles”](#) for information.

26.7.3.3 `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.

26.7.3.4 `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.

26.7.3.5 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).

26.7.3.6 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 25.4.2, “Changing the Mode of Individual Profiles”](#) for information.

26.7.3.7 **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](#).

26.7.3.8 **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 26.7.3, "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 27, 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 26.1, “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 26.2, “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 26.1, “Learning Mode Exception: Controlling Access to Specific Resources”* and *Example 26.2, “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 26.1: 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 26.2, “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 26.2: 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 23.6, "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

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 23.7, "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 23.6, "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 26.7.3.14, "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.

26.7.3.9 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 -ltr /path/to/logfile*` | aa-logprof -f -**.

26.7.3.10 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

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 23.7, "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 23.6, "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.

26.7.3.11 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 26.1, “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 23.12, "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 23.7, "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.

26.7.3.12 `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.

26.7.3.13 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.

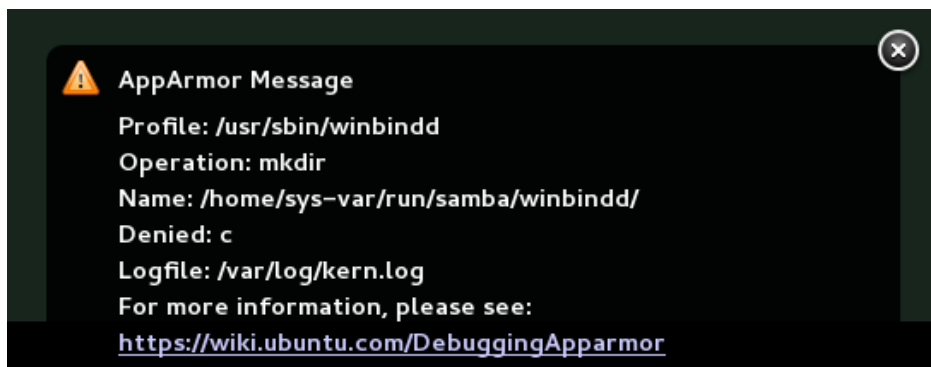


FIGURE 26.1: **aa-notify** Message in GNOME

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**.

26.7.3.14 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`.

26.8 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.

27 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 27.1.2, “Location and Directory Directives”](#).



Tip: For More Information

For more information, see the `change_hat` man page.

27.1 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.

27.1.1 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.

27.1.2 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>
```

27.2 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.

27.2.1 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 27.1: 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
```

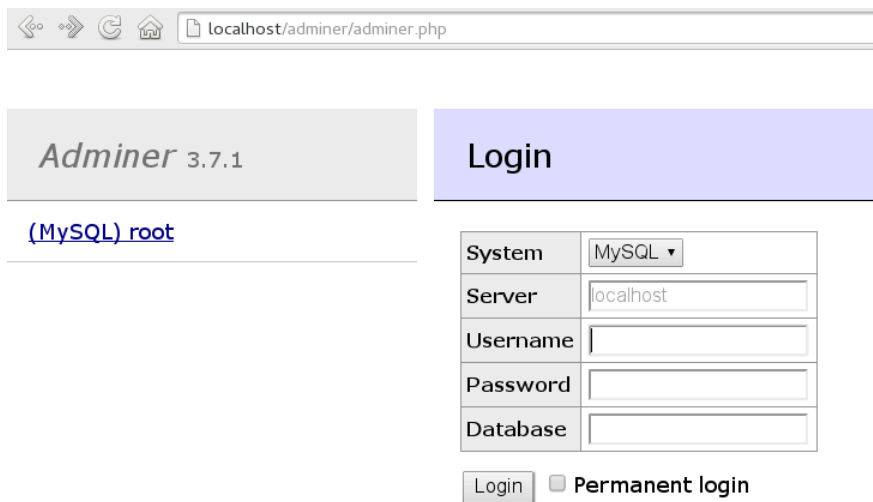
and then restart Apache with

```
tux > sudo systemctl restart apache2
```

2. Make sure MariaDB is running. If unsure, restart it with

```
tux > sudo systemctl restart mariadb
```

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.



Adminer 3.7.1

Login

[\(MySQL\) root](#)

System	MySQL ▾
Server	localhost
Username	<input type="text"/>
Password	<input type="password"/>
Database	<input type="text"/>

Login Permanent login

FIGURE 27.1: 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 27.2: 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 27.3: 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 27.4: 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 26.7.3.8, “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_3jdmii9cacj1e3jnahtopajl7p064ai242]
```

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.

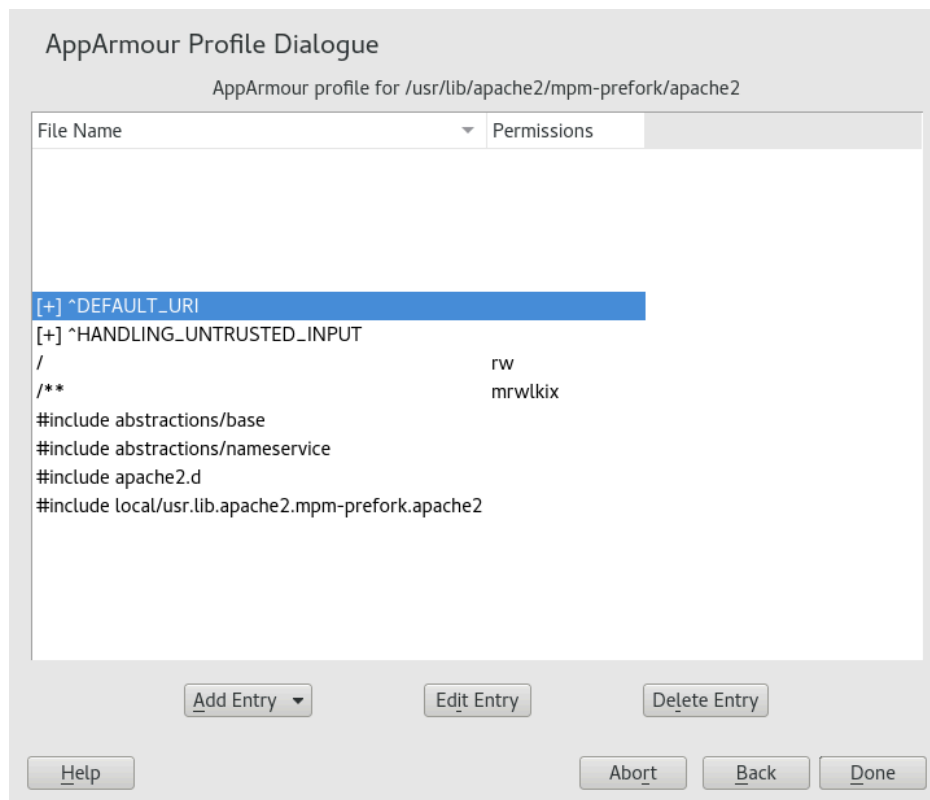


Note: Hat and Parent Profile Relationship

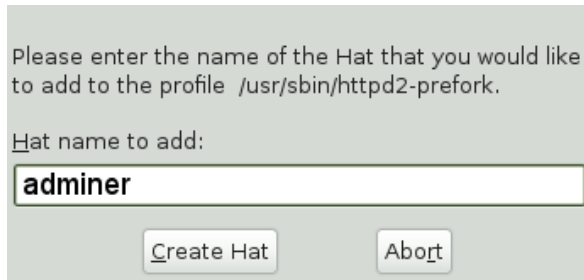
The profile `^adminer` is only available in the context of a process running under the parent profile `usr.sbin.httpd2-prefork`.

27.2.2 Adding Hats and Entries to Hats in YaST

When you use the *Edit Profile* dialog (for instructions, refer to [Section 25.2, “Editing Profiles”](#)) or when you add a new profile using *Manually Add Profile* (for instructions, refer to [Section 25.1, “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*.

28 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*.

29 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.

29.1 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.

29.2 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.

29.2.1 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.

29.2.2 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 25.2, “Editing Profiles”*.

29.2.3 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 26.7.3.9, “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 26.7.3.9, “aa-logprof—Scanning the System Log”*.

30 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.

30.1 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.

30.2 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 30.1: 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)

30.3 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.

30.4 Troubleshooting

This section lists the most common problems and error messages that may occur using AppArmor.

30.4.1 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 26.7.3.9, "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.

30.4.2 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 23.5, "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
```

30.4.3 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.

30.4.4 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`.

30.4.5 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`.

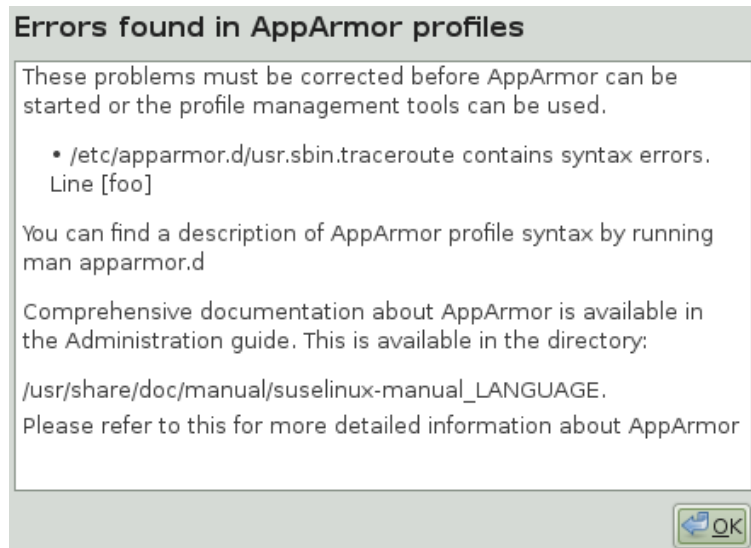
30.4.6 How to Spot and Fix AppArmor Syntax Errors

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.

```
root # systemctl start apparmor.service
```

```
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.

30.5 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.

31 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

32 Configuring SELinux **305**

32 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

32.1 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 openSUSE Leap 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 were 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.

32.1.1 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.

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.

32.1.2 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.



Warning: No Default Policy Included

No default or reference policy is provided in openSUSE Leap. SELinux will not operate without a policy, so you must build and install one. The SELinux Reference Policy Project (<https://github.com/SELinuxProject/refpolicy/wiki>) should be helpful in providing examples and detailed information on creating your own policies, and this chapter also provides guidance on managing your SELinux 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 does not support third-party policies.

32.2 Policy

The policy is the key component in SELinux. Note that no default or reference policy is included in openSUSE Leap, and you must build and install one that is customized for your needs before proceeding. (See *Warning: No Default Policy Included.*)

Your SELinux policy 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 32.1: "Security Context Settings Using ls -Z" shows the security context settings for the directories in the `/` directory of an openSUSE Leap system with an SELinux-labeled file system.

EXAMPLE 32.1: 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 `gettattr`.

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.

32.3 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.

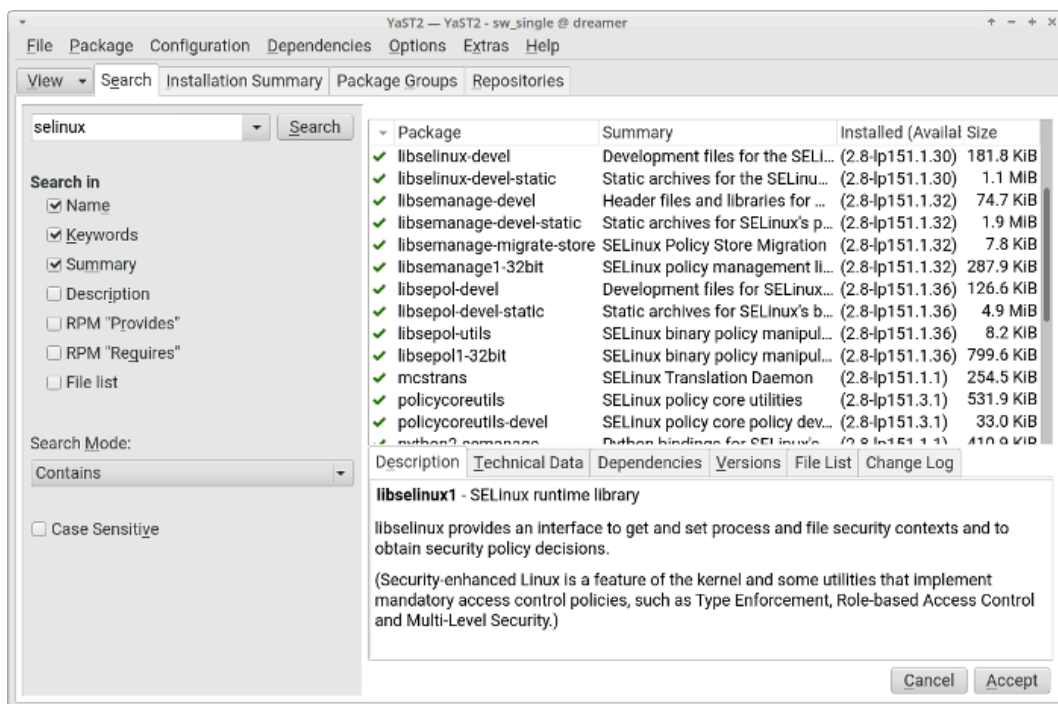


FIGURE 32.1: SELECTING ALL SELINUX PACKAGES IN YAST

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.

32.4 SELinux Policy

The policy is an essential component of SELinux. openSUSE Leap 15.2 does not include a default or reference policy, and you must build a policy that is customized for your installation. For testing and learning, see The SELinux Reference Policy Project at <https://github.com/SELinuxProject/refpolicy/wiki>. You must have a policy, as SELinux will not operate without one.

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 32.2: “Verifying that SELinux is functional”*.

EXAMPLE 32.2: 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
```

32.5 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 32.3, "Getting a List of Booleans and Verifying Policy Access"* shows part of the output of this command.

EXAMPLE 32.3: 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_ftp_full_access    -> off  allow_ftp_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 32.4: "Getting File Context Information"* for partial output of this command).

EXAMPLE 32.4: 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
```

<code>/var/run/vbe.*</code>	regular file
<code>system_u:object_r:hald_var_run_t</code>	
<code>/var/run/vmnat.*</code>	socket
<code>system_u:object_r:vmware_var_run_t</code>	
<code>/var/run/vmware.*</code>	all files
<code>system_u:object_r:vmware_var_run_t</code>	
<code>/var/run/watchdog*.pid</code>	regular file
<code>system_u:object_r:watchdog_var_run_t</code>	
<code>/var/run/winbindd(/.*)?</code>	all files
<code>system_u:object_r:winbind_var_run_t</code>	
<code>/var/run/wnn-unix(/.*)</code>	all files
<code>system_u:object_r:canna_var_run_t</code>	
<code>/var/run/wpa_supplicant(/.*)?</code>	all files
<code>system_u:object_r:NetworkManager_var_run_t</code>	
<code>/var/run/wpa_supplicant-global</code>	socket
<code>system_u:object_r:NetworkManager_var_run_t</code>	
<code>/var/run/xdmctl(/.*)?</code>	all files
<code>system_u:object_r:xdm_var_run_t</code>	
<code>/var/run/yiff-[0-9]+*.pid</code>	regular file
<code>system_u:object_r:soundd_var_run_t</code>	

32.6 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.

32.6.1 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 32.5: "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 32.5: 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 32.6: “Showing SELinux settings for processes with `ps Zaux`”](#) for example you can see the context settings for processes on your server.

EXAMPLE 32.6: SHOWING SELINUX SETTINGS FOR PROCESSES WITH `ps` `Zaux`

```
tux > sudo ps Zaux
```

LABEL	USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START
system_u:system_r:init_t 0:00 init [5]	root	1	0.0	0.0	10640	808	?	Ss	05:31
system_u:system_r:kernel_t 0:00 [kthreadd]	root	2	0.0	0.0	0	0	?	S	05:31
system_u:system_r:kernel_t 0:00 [ksoftirqd/0]	root	3	0.0	0.0	0	0	?	S	05:31
system_u:system_r:kernel_t 0:00 [migration/0]	root	6	0.0	0.0	0	0	?	S	05:31
system_u:system_r:kernel_t 0:00 [watchdog/0]	root	7	0.0	0.0	0	0	?	S	05:31
system_u:system_r:sysadm_t 0:00 /usr/sbin/mcelog --daemon --config-file /etc/mcelog/mcelog.conf	root	2344	0.0	0.0	27640	852	?	Ss	05:32
system_u:system_r:sshd_t 0:00 /usr/sbin/sshd -o PidFile=/var/run/sshd.init.pid	root	3245	0.0	0.0	69300	1492	?	Ss	05:32
system_u:system_r:cupsd_t 0:00 /usr/sbin/cupsd	root	3265	0.0	0.0	68176	2852	?	Ss	05:32
system_u:system_r:nscd_t 0:00 /usr/sbin/nscd	root	3267	0.0	0.0	772876	1380	?	Ssl	05:32
system_u:system_r:postfix_master_t 0:00 /usr/lib/postfix/master	root	3334	0.0	0.0	38320	2424	?	Ss	05:32
system_u:system_r:postfix_qmgr_t 0:00 qmgr -l -t fifo -u	postfix	3358	0.0	0.0	40216	2252	?	S	05:32
system_u:system_r:crond_t 0:00 /usr/sbin/cron	root	3415	0.0	0.0	14900	800	?	Ss	05:32
system_u:system_r:fsdaemon_t 0:00 /usr/sbin/smartd	root	3437	0.0	0.0	16468	1040	?	S	05:32
system_u:system_r:sysadm_t 0:00 login -- root	root	3441	0.0	0.0	66916	2152	?	Ss	05:32
system_u:system_r:sysadm_t 0:00 /sbin/mingetty tty2	root	3442	0.0	0.0	4596	800	tty2	Ss+	05:32

32.6.2 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.

32.6.3 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 32.7: "Viewing Default File Contexts"](#)).

EXAMPLE 32.7: 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.

32.6.4 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.

32.6.5 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_ftp_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_ftp_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_ftp_anon_write
```

32.6.6 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
```

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`.

32.7 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 32.8: "Example Lines from `/etc/audit/audit.log`"* you can see a partial example of the contents of `/var/log/audit/audit.log`

EXAMPLE 32.8: 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 sccontext=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 sccontext=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 sccontext=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 sccontext=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 32.9: 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 32.10: 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 32.11: 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 **se-module -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*

- 33 Understanding Linux Audit **327**
- 34 Setting Up the Linux Audit Framework **363**
- 35 Introducing an Audit Rule Set **375**
- 36 Useful Resources **386**

33 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 \(https://www.commoncriteriaportal.org/\)](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.

33.1 Introducing the Components of Linux Audit

The following figure illustrates how the various components of audit interact with each other:

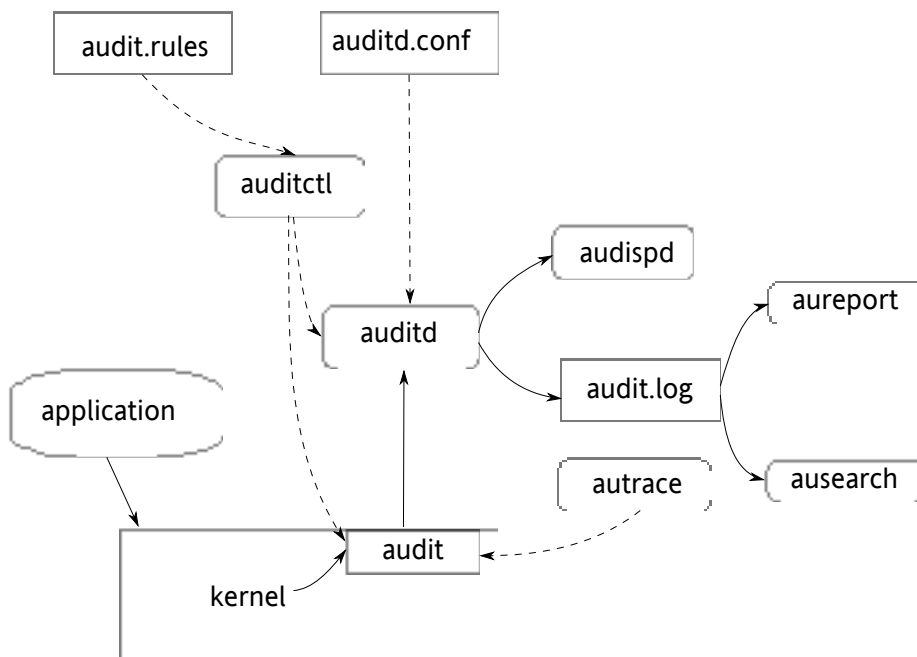


FIGURE 33.1: INTRODUCING THE COMPONENTS OF LINUX AUDIT

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 33.2, “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 33.3, “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 33.4, "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 33.5, "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 33.6, "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 33.9, "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 33.7, "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.

33.2 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 33.3](#),

*“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.

33.3 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 33.4, "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 -eFLAG`, a status message (including information on each of the above-mentioned parameters) is printed. The following example highlights the typical audit status message.

EXAMPLE 33.1: EXAMPLE OUTPUT OF `auditctl -s`

```
AUDIT_STATUS: enabled=1 flag=2 pid=3105 rate_limit=0 backlog_limit=8192 lost=0 backlog=0
```

TABLE 33.1: AUDIT STATUS FLAGS

Flag	Meaning [Possible Values]	Command
<code>enabled</code>	Set the enable flag. [0..2] 0 = disable, 1 = enable, 2 = enable and lock down the configuration	<code>auditctl -e [0 1 2]</code>
<code>flag</code>	Set the failure flag. [0..2] 0 = silent, 1 = printk, 2 = pan- ic (immediate halt without synchronizing pending data to disk)	<code>auditctl -f [0 1 2]</code>
<code>pid</code>	Process ID under which <code>au- ditd</code> is running.	—
<code>rate_limit</code>	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.	<code>auditctl -r RATE</code>
<code>backlog_limit</code>	Specify the maximum num- ber of outstanding audit buffers allowed. If all buffers	<code>auditctl -b BACKLOG</code>

Flag	Meaning [Possible Values]	Command
	are full, the action specified in the failure flag is triggered.	
<u>lost</u>	Count the current number of lost audit messages.	—
<u>backlog</u>	Count the current number of outstanding audit buffers.	—

33.4 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 33.2: EXAMPLE AUDIT RULES—AUDIT SYSTEM PARAMETERS

```
-b 1000 ①
-f 1 ②
-r 10 ③
-e 1 ④
```

- ① 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.
- ② Specify the failure flag to use. See *Table 33.1, "Audit Status Flags"* for possible values.
- ③ Specify the maximum number of messages per second that may be issued by the kernel. See *Table 33.1, "Audit Status Flags"* for details.

- 4 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 33.3: EXAMPLE AUDIT RULES—FILE SYSTEM AUDITING

```
-w /etc/shadow ❶  
-w /etc -p rx ❷  
-w /etc/passwd -k fk_passwd -p rwx 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 33.4: EXAMPLE AUDIT RULES—SYSTEM CALL AUDITING

```
-a exit,always -S mkdir ❶  
-a exit,always -S access -F a1=4 ❷  
-a exit,always -S ipc -F a0=2 ❸  
-a exit,always -S open -F success!=0 ❹  
-a task,always -F auid=0 ❺  
-a task,always -F uid=0 -F auid=501 -F gid=wheel ❻
```

- 1 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.
- 2 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.
- 3 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`.
- 4 This rule audits failed attempts to call open.
- 5 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`.
- 6 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 35.6, "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 33.5: DELETING AUDIT RULES AND EVENTS

```
-D 1  
-d exit,always -S mkdir 2  
-W /etc 3
```

- 1 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 `auditctl` to avoid having the trace rules clash with any rules present in the `audit.rules` file.
- 2 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.
- 3 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 33.6: LISTING RULES WITH `auditctl -l`

```
exit,always watch=/etc perm=rx
exit,always watch=/etc/passwd perm=rwx key=fk_passwd
exit,always watch=/etc/shadow perm=rwx
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.

33.5 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.

33.5.1 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 33.7: 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.

eid, 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 33.8, “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 33.8: AN ADVANCED AUDIT EVENT—LOGIN VIA SSH

```
type=USER_AUTH msg=audit(1234877011.791:7731): user pid=26127 uid=0 ①
aid=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 ②
aid=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 ③
aid=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 aid=4294967295 new aid=0 old ses=4294967295 new ses=1172
type=USER_START msg=audit(1234877011.799:7735): user pid=26125 uid=0 ④
aid=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 ⑤
aid=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 ⑥
aid=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)'

```

- ① PAM reports that it 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.
- ② PAM reports that it has successfully determined whether the user is authorized to log in.
- ③ PAM reports that the appropriate credentials to log in have been acquired and that the terminal changed to a normal terminal (/dev/pts/0).
- 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.

33.5.2 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 audit 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 audit 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 33.8, “Visualizing Audit Data”](#) are examples of how to further process the data generated by audit.

33.6 Querying the Audit Daemon Logs with **aureport**

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, **aureport** is the tool to use.

aureport 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 -o 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 33.5, "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`.

33.7 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**.

33.8 Visualizing Audit Data

Neither the data trail in `/var/log/audit/audit.log` nor the different report types generated by **aureport**, described in *Section 33.5.2, "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 34.6, "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
```

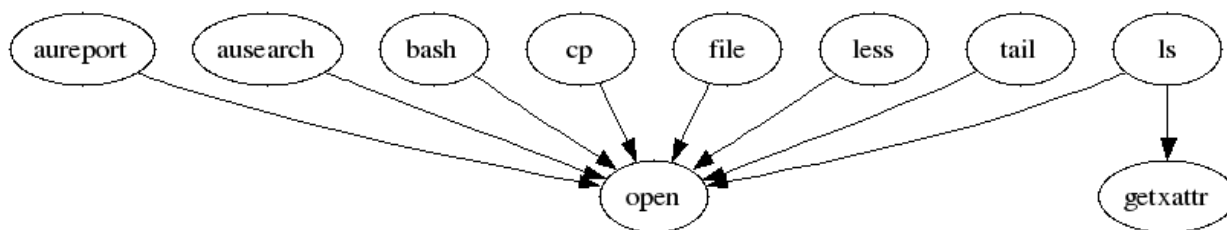


FIGURE 33.2: FLOW GRAPH—PROGRAM VERSUS SYSTEM CALL RELATIONSHIP

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
```

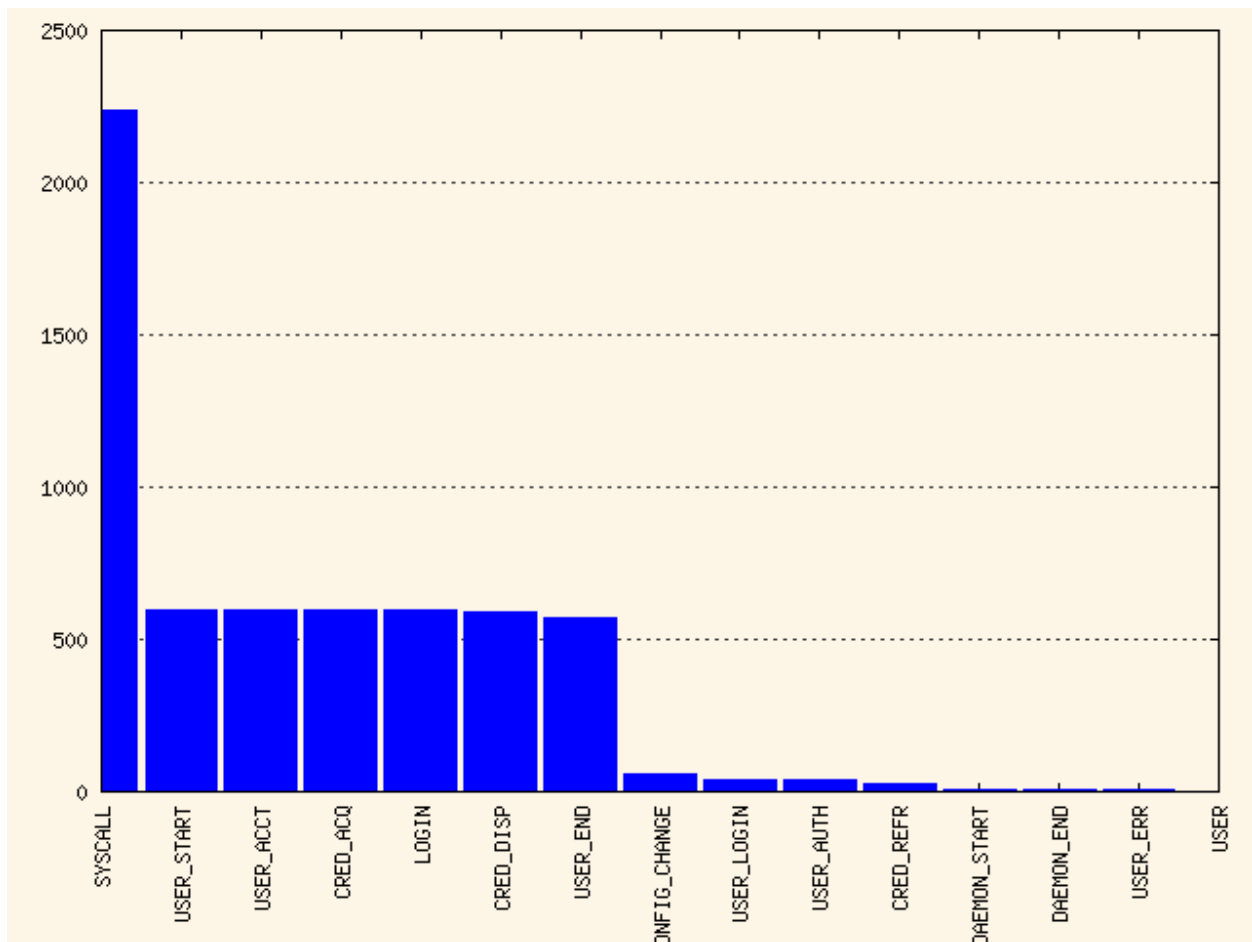


FIGURE 33.3: BAR CHART—COMMON EVENT TYPES

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>.

33.9 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 33.9: 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 33.10: [EXAMPLE /ETC/AUDISP/PLUGINS.D/SYSLOG.CONF](#)

```
active = no
direction = out
path = builtin_syslog
type = builtin
args = LOG_INFO
format = string
```

34 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 35, Introducing an Audit Rule Set*.

To set up audit on openSUSE Leap, you need to complete the following steps:

PROCEDURE 34.1: 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 34.6, "Configuring Log Visualization"*, install `gnuplot` and `graphviz` from the openSUSE Leap media.
2. Determine the components to audit. Refer to *Section 34.1, "Determining the Components to Audit"* for details.
3. Check or modify the basic audit daemon configuration. Refer to *Section 34.2, "Configuring the Audit Daemon"* for details.
4. Enable auditing for system calls. Refer to *Section 34.3, "Enabling Audit for System Calls"* for details.
5. Compose audit rules to suit your scenario. Refer to *Section 34.4, "Setting Up Audit Rules"* for details.
6. Generate logs and configure tailor-made reports. Refer to *Section 34.5, "Configuring Audit Reports"* for details.
7. Configure optional log visualization. Refer to *Section 34.6, "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`.

34.1 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 35, 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 34.3, "Enabling Audit for System Calls"* and proceed to *Section 34.4, "Setting Up Audit Rules"*.

34.2 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 33.2, “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/auditd
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 33.2, “Configuring the Audit Daemon”](#) for detailed background information about the `auditd.conf` configuration parameters.

34.3 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`.

34.4 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 35, 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 rwx
-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 33.4, "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.

34.5 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 33.5.2, “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 34.6, “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

```

- 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

```

- 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=7ffffed5b57d0 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 33.5.2, "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 34.6, "Configuring Log Visualization"*.

34.6 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 34.5, “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 **mk-graph** 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**.

35 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 35.1, “Adding Basic Audit Configuration Parameters”](#))
- Watches on audit log files and configuration files (see [Section 35.2, “Adding Watches on Audit Log Files and Configuration Files”](#))
- Monitoring operations on file system objects (see [Section 35.3, “Monitoring File System Objects”](#))
- Monitoring security databases (see [Section 35.4, “Monitoring Security Configuration Files and Databases”](#))
- Monitoring miscellaneous system calls ([Section 35.5, “Monitoring Miscellaneous System Calls”](#))
- Filtering system call arguments (see [Section 35.6, “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.

35.1 Adding Basic Audit Configuration Parameters

```
-D ①  
-b 8192 ②  
-f 2 ③
```

- ① Delete any preexisting rules before starting to define new ones.
- ② 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.
- ③ 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.

35.2 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 wa ②  
-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.

35.3 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 33.2, “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.

35.4 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
```

```

3
-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
4
w /etc/pam.d/
5
-w /etc/aliases -p wa
-w /etc/postfix/ -p wa

6
-w /etc/ssh/sshd_config

-w /etc/stunnel/stunnel.conf
-w /etc/stunnel/stunnel.pem

-w /etc/vsftpd.ftpusers
-w /etc/vsftpd.conf

7
-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`.

35.5 Monitoring Miscellaneous System Calls

Apart from auditing file system related system calls, as described in [Section 35.3, “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.

```
①
-a entry,always -S clone -S fork -S vfork

②
-a entry,always -S umask

③
-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.

35.6 Filtering System Call Arguments

In addition to the system call auditing introduced in [Section 35.3, “Monitoring File System Objects”](#) and [Section 35.5, “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 33.2, “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 ❶  
-a entry,always -S access -F a1=6 ❷  
-a entry,always -S access -F a1=7 ❸
```

- ❶ 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.
- ❷ 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.
- ❸ 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 ❶
```



```

## Use this line on x86_64, ia64 instead
#-a entry,always -S socket -F a0=10

-a entry,always -S socketcall -F a0=5 ②
## Use this line on x86_64, ia64 instead
#-a entry, always -S accept

```

- ① 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`.
- ② 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.

```

①
## 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

②
## 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.

35.7 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"
```

36 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

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary

formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.  
Permission is granted to copy, distribute  
and/or modify this document  
under the terms of the GNU Free  
Documentation License, Version 1.2  
or any later version published by the Free  
Software Foundation;  
with no Invariant Sections, no Front-Cover  
Texts, and no Back-Cover Texts.  
A copy of the license is included in the  
section entitled "GNU  
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being LIST  
THEIR TITLES, with the  
Front-Cover Texts being LIST, and with the  
Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.



System Analysis and Tuning Guide

openSUSE Leap 15.2



System Analysis and Tuning Guide

openSUSE Leap 15.2

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.

Publication Date: July 06, 2020

SUSE LLC

1800 South Novell Place

Provo, UT 84606

USA

<https://documentation.suse.com> ↗

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

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

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

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

Contents

About This Guide xi

I BASICS 1

1 General Notes on System Tuning 2

- 1.1 Be Sure What Problem to Solve 2
- 1.2 Rule Out Common Problems 3
- 1.3 Finding the Bottleneck 4
- 1.4 Step-by-step Tuning 4

II SYSTEM MONITORING 5

2 System Monitoring Utilities 6

- 2.1 Multi-Purpose Tools 6
 - vmstat** 6 • **dstat** 9 • System Activity Information: **sar** 11
- 2.2 System Information 14
 - Device Load Information: **iostat** 14 • Processor Activity Monitoring: **mpstat** 15 • Processor Frequency Monitoring: **turbostat** 16 • Task Monitoring: **pidstat** 17 • Kernel Ring Buffer: **dmesg** 17 • List of Open Files: **lsof** 17 • Kernel and udev Event Sequence Viewer: **udevadm monitor** 18
- 2.3 Processes 19
 - Interprocess Communication: **ipcs** 19 • Process List: **ps** 20 • Process Tree: **pstree** 21 • Table of Processes: **top** 22 • A top-like I/O Monitor: **iotop** 23 • Modify a process's niceness: **nice** and **renice** 24
- 2.4 Memory 25
 - Memory Usage: **free** 25 • Detailed Memory Usage: `/proc/meminfo` 25 • Process Memory Usage: `smaps` 29 • `numaTOP` 30

- 2.5 Networking 30
 - Basic Network Diagnostics: **ip** 30 • Show the Network Usage of Processes: **nethogs** 31 • Ethernet Cards in Detail: **ethtool** 32 • Show the Network Status: **ss** 33
- 2.6 The /proc File System 34
 - procinfo** 37 • System Control Parameters: /proc/sys/ 38
- 2.7 Hardware Information 39
 - PCI Resources: **lspci** 39 • USB Devices: **lsusb** 40 • Monitoring and Tuning the Thermal Subsystem: **tmon** 40 • MCELog: Machine Check Exceptions (MCE) 41 • x86_64: **dmidecode**: DMI Table Decoder 42
- 2.8 Files and File Systems 43
 - Determine the File Type: **file** 43 • File Systems and Their Usage: **mount**, **df** and **du** 43 • Additional Information about ELF Binaries 44 • File Properties: **stat** 45
- 2.9 User Information 46
 - User Accessing Files: **fuser** 46 • Who Is Doing What: **w** 46
- 2.10 Time and Date 47
 - Time Measurement with **time** 47
- 2.11 Graph Your Data: RRDtool 48
 - How RRDtool Works 48 • A Practical Example 49 • For More Information 53
- 3 Analyzing and Managing System Log Files 54**
 - 3.1 System Log Files in /var/log/ 54
 - 3.2 Viewing and Parsing Log Files 56
 - 3.3 Managing Log Files with **logrotate** 56
 - 3.4 Monitoring Log Files with **logwatch** 57
 - 3.5 Using **logger** to Make System Log Entries 58

III KERNEL MONITORING 60

4 SystemTap—Filtering and Analyzing System Data 61

4.1 Conceptual Overview 61

SystemTap Scripts 61 • Tapsets 62 • Commands and Privileges 62 • Important Files and Directories 63

4.2 Installation and Setup 64

4.3 Script Syntax 65

Probe Format 66 • SystemTap Events (Probe Points) 67 • SystemTap Handlers (Probe Body) 68

4.4 Example Script 72

4.5 User Space Probing 73

4.6 For More Information 74

5 Kernel Probes 75

5.1 Supported Architectures 75

5.2 Types of Kernel Probes 76

Kprobes 76 • Jprobes 76 • Return Probe 76

5.3 Kprobes API 77

5.4 debugfs Interface 78

Listing Registered Kernel Probes 78 • How to Switch All Kernel Probes On or Off 78

5.5 For More Information 79

6 Hardware-Based Performance Monitoring with Perf 80

6.1 Hardware-Based Monitoring 80

6.2 Sampling and Counting 80

6.3 Installing Perf 81

6.4 Perf Subcommands 81

6.5	Counting Particular Types of Event	82
6.6	Recording Events Specific to Particular Commands	83
6.7	For More Information	83
7	OProfile—System-Wide Profiler	85
7.1	Conceptual Overview	85
7.2	Installation and Requirements	85
7.3	Available OProfile Utilities	86
7.4	Using OProfile	86
	Creating a Report	86
	Getting Event Configurations	87
7.5	Generating Reports	89
7.6	For More Information	89
IV	RESOURCE MANAGEMENT	91
8	General System Resource Management	92
8.1	Planning the Installation	92
	Partitioning	92
	Installation Scope	93
	Default Target	93
8.2	Disabling Unnecessary Services	93
8.3	File Systems and Disk Access	94
	File Systems	95
	Time Stamp Update Policy	95
	Prioritizing Disk Access with ionice	96
9	Kernel Control Groups	97
9.1	Overview	97
9.2	Setting Resource Limits	97
9.3	Preventing Fork Bombs with TasksMax	99
	Finding the Current Default TasksMax Values	100
	Overriding the DefaultTasksMax Value	100
	Default TasksMax Limit on Users	102
9.4	For More Information	102

10 Automatic Non-Uniform Memory Access (NUMA) Balancing 104

10.1 Implementation 104

10.2 Configuration 105

10.3 Monitoring 106

10.4 Impact 107

11 Power Management 109

11.1 Power Management at CPU Level 109

C-States (Processor Operating States) 109 • P-States (Processor Performance States) 110 • Turbo Features 111

11.2 In-Kernel Governors 111

11.3 The `cpupower` Tools 112

Viewing Current Settings with `cpupower` 113 • Viewing Kernel Idle Statistics with `cpupower` 113 • Monitoring Kernel and Hardware Statistics with `cpupower` 114 • Modifying Current Settings with `cpupower` 116

11.4 Special Tuning Options 116

Tuning Options for P-States 116

11.5 Troubleshooting 116

11.6 For More Information 117

11.7 Monitoring Power Consumption with `powerTOP` 118

V KERNEL TUNING 121

12 Tuning I/O Performance 122

12.1 Switching I/O Scheduling 122

12.2 Available I/O Elevators with `blk-mq` I/O path 122

MQ-DEADLINE 123 • NONE 124 • BFQ (Budget Fair Queueing) 124 • KYBER 126

12.3 I/O Barrier Tuning 126

13 Tuning the Task Scheduler 127

- 13.1 Introduction 127
 - Preemption 127 • Timeslice 128 • Process Priority 128
- 13.2 Process Classification 128
- 13.3 Completely Fair Scheduler 129
 - How CFS Works 130 • Grouping Processes 130 • Kernel Configuration Options 130 • Terminology 131 • Changing Real-time Attributes of Processes with **chrt** 132 • Runtime Tuning with **sysctl** 132 • Debugging Interface and Scheduler Statistics 136
- 13.4 For More Information 138

14 Tuning the Memory Management Subsystem 139

- 14.1 Memory Usage 139
 - Anonymous Memory 140 • Pagecache 140 • Buffercache 140 • Buffer Heads 140 • Writeback 140 • Readahead 141 • VFS caches 141
- 14.2 Reducing Memory Usage 142
 - Reducing malloc (Anonymous) Usage 142 • Reducing Kernel Memory Overheads 142 • Memory Controller (Memory Cgroups) 142
- 14.3 Virtual Memory Manager (VM) Tunable Parameters 143
 - Reclaim Ratios 143 • Writeback Parameters 144 • Readahead Parameters 145 • Transparent Huge Page Parameters 146 • khugepaged Parameters 147 • Further VM Parameters 148
- 14.4 Monitoring VM Behavior 148

15 Tuning the Network 150

- 15.1 Configurable Kernel Socket Buffers 150
- 15.2 Detecting Network Bottlenecks and Analyzing Network Traffic 152
- 15.3 Netfilter 152
- 15.4 Improving the Network Performance with Receive Packet Steering (RPS) 152

VI	HANDLING SYSTEM DUMPS	155
16	Tracing Tools	156
16.1	Tracing System Calls with strace	156
16.2	Tracing Library Calls with ltrace	160
16.3	Debugging and Profiling with Valgrind	161
	General Information	161
	Default Options	162
	How Valgrind Works	162
	Messages	163
	Error Messages	165
16.4	For More Information	165
17	Kexec and Kdump	166
17.1	Introduction	166
17.2	Required Packages	166
17.3	Kexec Internals	167
17.4	Calculating crashkernel Allocation Size	168
17.5	Basic Kexec Usage	171
17.6	How to Configure Kexec for Routine Reboots	172
17.7	Basic Kdump Configuration	172
	Manual Kdump Configuration	173
	YaST Configuration	175
	Kdump over SSH	177
17.8	Analyzing the Crash Dump	178
	Kernel Binary Formats	179
17.9	Advanced Kdump Configuration	182
17.10	For More Information	183
18	Using systemd-coredump to Debug Application Crashes	185
18.1	Use and Configuration	185

VII SYNCHRONIZED CLOCKS WITH PRECISION TIME PROTOCOL 188

19 Precision Time Protocol 189

19.1 Introduction to PTP 189

PTP Linux Implementation 189

19.2 Using PTP 190

Network Driver and Hardware Support 190 • Using **ptp4l** 191 • **ptp4l**
Configuration File 192 • Delay Measurement 192 • PTP Management
Client: **pmc** 193

19.3 Synchronizing the Clocks with **phc2sys** 194

Verifying Time Synchronization 195

19.4 Examples of Configurations 196

19.5 PTP and NTP 197

NTP to PTP Synchronization 197 • Configuring PTP-NTP Bridge 198

A GNU Licenses 199

A.1 GNU Free Documentation License 199

About This Guide

openSUSE Leap is used for a broad range of usage scenarios in enterprise and scientific data centers. SUSE has ensured openSUSE Leap is set up in a way that it accommodates different operation purposes with optimal performance. However, openSUSE Leap must meet very different demands when employed on a number crunching server compared to a file server, for example. It is not possible to ship a distribution that is optimized for all workloads. Different workloads vary substantially in some aspects. Most important among those are I/O access patterns, memory access patterns, and process scheduling. A behavior that perfectly suits a certain workload might reduce performance of another workload. For example, I/O-intensive tasks, such as handling database requests, usually have completely different requirements than CPU-intensive tasks, such as video encoding. The versatility of Linux makes it possible to configure your system in a way that it brings out the best in each usage scenario.

This manual introduces you to means to monitor and analyze your system. It describes methods to manage system resources and to tune your system. This guide does *not* offer recipes for special scenarios, because each server has got its own different demands. It rather enables you to thoroughly analyze your servers and make the most out of them.

Part I, "Basics"

Tuning a system requires a carefully planned proceeding. Learn which steps are necessary to successfully improve your system.

Part II, "System Monitoring"

Linux offers a large variety of tools to monitor almost every aspect of the system. Learn how to use these utilities and how to read and analyze the system log files.

Part III, "Kernel Monitoring"

The Linux kernel itself offers means to examine every nut, bolt and screw of the system. This part introduces you to SystemTap, a scripting language for writing kernel modules that can be used to analyze and filter data. Collect debugging information and find bottlenecks by using kernel probes and Perf. Last, monitor applications with Oprofile.

Part IV, "Resource Management"

Learn how to set up a tailor-made system fitting exactly the server's need. Get to know how to use power management while at the same time keeping the performance of a system at a level that matches the current requirements.

Part V, "Kernel Tuning"

The Linux kernel can be optimized either by using `sysctl`, via the `/proc` and `/sys` file systems or by kernel command line parameters. This part covers tuning the I/O performance and optimizing the way how Linux schedules processes. It also describes basic principles of memory management and shows how memory management can be fine-tuned to suit needs of specific applications and usage patterns. Furthermore, it describes how to optimize network performance.

Part VI, "Handling System Dumps"

This part enables you to analyze and handle application or system crashes. It introduces tracing tools such as `strace` or `ltrace` and describes how to handle system crashes using `Kexec` and `Kdump`.

1 Available Documentation



Note: Online Documentation and Latest Updates

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

The following documentation is available for this product:

Book "Start-Up"

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

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.

System Analysis and Tuning Guide

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

Book “GNOME User Guide”

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

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

2 Giving Feedback

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

Bug Reports

Report issues with the documentation at <https://bugzilla.opensuse.org/>. To simplify this process, you can use the *Report Documentation Bug* links next to headlines in the HTML version of this document. These preselect the right product and category in Bugzilla and add a link to the current section. You can start typing your bug report right away. A Bugzilla account is required.

Contributions

To contribute to this documentation, use the *Edit Source* links next to headlines in the HTML version of this document. They take you to the source code on GitHub, where you can open a pull request. A GitHub account is required.

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

Mail

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

Help

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

3 Documentation Conventions

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

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

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

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

```
tux > command
```

- Notices



Warning: Warning Notice

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



Important: Important Notice

Important information you should be aware of before proceeding.



Note: Note Notice

Additional information, for example about differences in software versions.



Tip: Tip Notice

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

I Basics

- 1 General Notes on System Tuning [2](#)

1 General Notes on System Tuning

This manual discusses how to find the reasons for performance problems and provides means to solve these problems. Before you start tuning your system, you should make sure you have ruled out common problems and have found the cause for the problem. You should also have a detailed plan on how to tune the system, because applying random tuning tips often will not help and could make things worse.

PROCEDURE 1.1: GENERAL APPROACH WHEN TUNING A SYSTEM

1. Specify the problem that needs to be solved.
2. In case the degradation is new, identify any recent changes to the system.
3. Identify why the issue is considered a performance problem.
4. Specify a metric that can be used to analyze performance. This metric could for example be latency, throughput, the maximum number of users that are simultaneously logged in, or the maximum number of active users.
5. Measure current performance using the metric from the previous step.
6. Identify the subsystem(s) where the application is spending the most time.
7.
 - a. Monitor the system and/or the application.
 - b. Analyze the data, categorize where time is being spent.
8. Tune the subsystem identified in the previous step.
9. Remeasure the current performance without monitoring using the same metric as before.
10. If performance is still not acceptable, start over with *Step 3*.

1.1 Be Sure What Problem to Solve

Before starting to tuning a system, try to describe the problem as exactly as possible. A statement like “The system is slow!” is not a helpful problem description. For example, it could make a difference whether the system speed needs to be improved in general or only at peak times.

Furthermore, make sure you can apply a measurement to your problem, otherwise you cannot verify if the tuning was a success or not. You should always be able to compare “before” and “after”. Which metrics to use depends on the scenario or application you are looking into. Relevant Web server metrics, for example, could be expressed in terms of:

Latency

The time to deliver a page

Throughput

Number of pages served per second or megabytes transferred per second

Active Users

The maximum number of users that can be downloading pages while still receiving pages within an acceptable latency

1.2 Rule Out Common Problems

A performance problem often is caused by network or hardware problems, bugs, or configuration issues. Make sure to rule out problems such as the ones listed below before attempting to tune your system:

- Check the output of the systemd journal (see *Book “Reference”, Chapter 11 “journalctl: Query the systemd Journal”*) for unusual entries.
- Check (using top or ps) whether a certain process misbehaves by eating up unusual amounts of CPU time or memory.
- Check for network problems by inspecting /proc/net/dev.
- In case of I/O problems with physical disks, make sure it is not caused by hardware problems (check the disk with the smartmontools) or by a full disk.
- Ensure that background jobs are scheduled to be carried out in times the server load is low. Those jobs should also run with low priority (set via nice).
- If the machine runs several services using the same resources, consider moving services to another server.
- Last, make sure your software is up-to-date.

1.3 Finding the Bottleneck

Finding the bottleneck very often is the hardest part when tuning a system. openSUSE Leap offers many tools to help you with this task. See *Part II, "System Monitoring"* for detailed information on general system monitoring applications and log file analysis. If the problem requires a long-time in-depth analysis, the Linux kernel offers means to perform such analysis. See *Part III, "Kernel Monitoring"* for coverage.

Once you have collected the data, it needs to be analyzed. First, inspect if the server's hardware (memory, CPU, bus) and its I/O capacities (disk, network) are sufficient. If these basic conditions are met, the system might benefit from tuning.

1.4 Step-by-step Tuning

Make sure to carefully plan the tuning itself. It is of vital importance to only do one step at a time. Only by doing so can you measure whether the change made an improvement or even had a negative impact. Each tuning activity should be measured over a sufficient time period to ensure you can do an analysis based on significant data. If you cannot measure a positive effect, do not make the change permanent. Chances are, that it might have a negative effect in the future.

II System Monitoring

- 2 System Monitoring Utilities 6
- 3 Analyzing and Managing System Log Files 54

2 System Monitoring Utilities

There are number of programs, tools, and utilities which you can use to examine the status of your system. This chapter introduces some and describes their most important and frequently used parameters.

For each of the described commands, examples of the relevant outputs are presented. In the examples, the first line is the command itself (after the `tux >` or `root #`). Omissions are indicated with square brackets (`[...]`) and long lines are wrapped where necessary. Line breaks for long lines are indicated by a backslash (`\`).

```
tux > command -x -y
output line 1
output line 2
output line 3 is annoyingly long, so long that \
    we need to break it
output line 4
[...]
output line 98
output line 99
```

The descriptions have been kept short so that we can include as many utilities as possible. Further information for all the commands can be found in the manual pages. Most of the commands also understand the parameter `--help`, which produces a brief list of possible parameters.

2.1 Multi-Purpose Tools

While most Linux system monitoring tools monitor only a single aspect of the system, there are a few tools with a broader scope. To get an overview and find out which part of the system to examine further, use these tools first.

2.1.1 **vmstat**

`vmstat` collects information about processes, memory, I/O, interrupts and CPU:

```
vmstat [options] [delay [count]]
```

When called without values for delay and count, it displays average values since the last reboot. When called with a value for delay (in seconds), it displays values for the given period (two seconds in the examples below). The value for count specifies the number of updates vmstat should perform. If not specified, it will run until manually stopped.

EXAMPLE 2.1: **vmstat** OUTPUT ON A LIGHTLY USED MACHINE

```
tux > vmstat 2
procs -----memory----- ---swap-- -----io----- -system-- -----cpu-----
 r b  swpd  free  buff  cache  si  so  bi  bo  in  cs  us  sy  id  wa  st
 1 0  44264 81520  424 935736  0  0  12  25  27  34  1  0  98  0  0
 0 0  44264 81552  424 935736  0  0  0   0  38  25  0  0 100  0  0
 0 0  44264 81520  424 935732  0  0  0   0  23  15  0  0 100  0  0
 0 0  44264 81520  424 935732  0  0  0   0  36  24  0  0 100  0  0
 0 0  44264 81552  424 935732  0  0  0   0  51  38  0  0 100  0  0
```

EXAMPLE 2.2: **vmstat** OUTPUT ON A HEAVILY USED MACHINE (CPU BOUND)

```
tux > vmstat 2
procs -----memory----- ---swap-- -----io----- -system-- -----cpu-----
 r b  swpd  free  buff  cache  si  so  bi  bo  in  cs  us  sy  id  wa  st
32 1  26236 459640 110240 6312648  0  0 9944   2 4552 6597 95  5  0  0  0
23 1  26236 396728 110336 6136224  0  0 9588   0 4468 6273 94  6  0  0  0
35 0  26236 554920 110508 6166508  0  0 7684 27992 4474 4700 95  5  0  0  0
28 0  26236 518184 110516 6039996  0  0 10830   4 4446 4670 94  6  0  0  0
21 5  26236 716468 110684 6074872  0  0  8734 20534 4512 4061 96  4  0  0  0
```

 **Tip: First Line of Output**

The first line of the vmstat output always displays average values since the last reboot.

The columns show the following:

r

Shows the number of processes in a runnable state. These processes are either executing or waiting for a free CPU slot. If the number of processes in this column is constantly higher than the number of CPUs available, this may be an indication of insufficient CPU power.

b

Shows the number of processes waiting for a resource other than a CPU. A high number in this column may indicate an I/O problem (network or disk).

swpd

The amount of swap space (KB) currently used.

free

The amount of unused memory (KB).

inact

Recently unused memory that can be reclaimed. This column is only visible when calling **vmstat** with the parameter -a (recommended).

active

Recently used memory that normally does not get reclaimed. This column is only visible when calling **vmstat** with the parameter -a (recommended).

buff

File buffer cache (KB) in RAM that contains file system metadata. This column is not visible when calling **vmstat** with the parameter -a.

cache

Page cache (KB) in RAM with the actual contents of files. This column is not visible when calling **vmstat** with the parameter -a.

si / so

Amount of data (KB) that is moved from swap to RAM (si) or from RAM to swap (so) per second. High so values over a long period of time may indicate that an application is leaking memory and the leaked memory is being swapped out. High si values over a long period of time could mean that an application that was inactive for a very long time is now active again. Combined high si and so values for prolonged periods of time are evidence of swap thrashing and may indicate that more RAM needs to be installed in the system because there is not enough memory to hold the working set size.

bi

Number of blocks per second received from a block device (for example, a disk read). Note that swapping also impacts the values shown here. The block size may vary between file systems but can be determined using the `stat` utility. If throughput data is required then `iostat` may be used.

bo

Number of blocks per second sent to a block device (for example, a disk write). Note that swapping also impacts the values shown here.

in

Interrupts per second. A high value may indicate a high I/O level (network and/or disk), but could also be triggered for other reasons such as inter-processor interrupts triggered by another activity. Make sure to also check `/proc/interrupts` to identify the source of interrupts.

cs

Number of context switches per second. This is the number of times that the kernel replaces executable code of one program in memory with that of another program.

us

Percentage of CPU usage executing application code.

sy

Percentage of CPU usage executing kernel code.

id

Percentage of CPU time spent idling. If this value is zero over a longer time, your CPU(s) are working to full capacity. This is not necessarily a bad sign—rather refer to the values in columns *r* and *b* to determine if your machine is equipped with sufficient CPU power.

wa

If "wa" time is non-zero, it indicates throughput lost because of waiting for I/O. This may be inevitable, for example, if a file is being read for the first time, background writeback cannot keep up, and so on. It can also be an indicator for a hardware bottleneck (network or hard disk). Lastly, it can indicate a potential for tuning the virtual memory manager (refer to [Chapter 14, Tuning the Memory Management Subsystem](#)).

st

Percentage of CPU time stolen from a virtual machine.

See `vmstat --help` for more options.

2.1.2 **dstat**

dstat is a replacement for tools such as `vmstat`, `iostat`, `netstat`, or `ifstat`. **dstat** displays information about the system resources in real time. For example, you can compare disk usage in combination with interrupts from the IDE controller, or compare network bandwidth with the disk throughput (in the same interval).

By default, its output is presented in readable tables. Alternatively, CSV output can be produced which is suitable as a spreadsheet import format.

It is written in Python and can be enhanced with plug-ins.

This is the general syntax:

```
dstat [-afv] [OPTIONS..] [DELAY [COUNT]]
```

All options and parameters are optional. Without any parameter, `dstat` displays statistics about CPU (`-c`, `--cpu`), disk (`-d`, `--disk`), network (`-n`, `--net`), paging (`-g`, `--page`), and the interrupts and context switches of the system (`-y`, `--sys`); it refreshes the output every second ad infinitum:

```
root # dstat
You did not select any stats, using -cdngy by default.
----total-cpu-usage---- -dsk/total- -net/total- ---paging-- ---system--
usr sys idl wai hiq siq| read  writ| recv  send| in   out | int  csw
  0   0 100   0   0   0| 15k  44k|   0   0 |   0  82B| 148  194
  0   0 100   0   0   0|   0   0 |5430B 170B|   0   0 | 163  187
  0   0 100   0   0   0|   0   0 |6363B 842B|   0   0 | 196  185
```

`-a`, `--all`
equal to `-cdngy` (default)

`-f`, `--full`
expand `-C`, `-D`, `-I`, `-N` and `-S` discovery lists

`-v`, `--vmstat`
equal to `-pmgdsc`, `-D total`

`DELAY`
delay in seconds between each update

`COUNT`
the number of updates to display before exiting

The default delay is 1 and the count is unspecified (unlimited).

For more information, see the man page of `dstat` and its Web page at <http://dag.wieer.com/home-made/dstat/>.

2.1.3 System Activity Information: **sar**

sar can generate extensive reports on almost all important system activities, among them CPU, memory, IRQ usage, IO, or networking. It can also generate reports on the fly. **sar** gathers all their data from the `/proc` file system.



Note: sysstat Package

sar is a part of the `sysstat` package either with YaST, or with **zypper in sysstat**.

2.1.3.1 Generating reports with **sar**

To generate reports on the fly, call **sar** with an interval (seconds) and a count. To generate reports from files specify a file name with the option `-f` instead of interval and count. If file name, interval and count are not specified, **sar** attempts to generate a report from `/var/log/sa/saDD`, where `DD` stands for the current day. This is the default location to where **sadc** (the system activity data collector) writes its data. Query multiple files with multiple `-f` options.

```
sar 2 10 # on-the-fly report, 10 times every 2 seconds
sar -f ~/reports/sar_2014_07_17 # queries file sar_2014_07_17
sar # queries file from today in /var/log/sa/
cd /var/log/sa && \
sar -f sa01 -f sa02 # queries files /var/log/sa/0[12]
```

Find examples for useful **sar** calls and their interpretation below. For detailed information on the meaning of each column, refer to the `man (1) of sar`.



Note: sysstat reporting when the service stops

When the `sysstat` service is stopped (for example, during reboot or shutdown), the tool still collects last-minute statistics by automatically running the `/usr/lib64/sa/sa1 -S ALL 1 1` command. The collected binary data is stored in the system activity data file.

2.1.3.1.1 CPU Usage Report: **sar**

When called with no options, **sar** shows a basic report about CPU usage. On multi-processor machines, results for all CPUs are summarized. Use the option `-P ALL` to also see statistics for individual CPUs.

```

root # sar 10 5
Linux 4.4.21-64-default (jupiter)          10/12/16          _x86_64_          (2 CPU)

17:51:29      CPU      %user      %nice      %system      %iowait      %steal      %idle
17:51:39      all      57,93      0,00      9,58      1,01      0,00      31,47
17:51:49      all      32,71      0,00      3,79      0,05      0,00      63,45
17:51:59      all      47,23      0,00      3,66      0,00      0,00      49,11
17:52:09      all      53,33      0,00      4,88      0,05      0,00      41,74
17:52:19      all      56,98      0,00      5,65      0,10      0,00      37,27
Average:      all      49,62      0,00      5,51      0,24      0,00      44,62

```

%iowait displays the percentage of time that the CPU was idle while waiting for an I/O request. If this value is significantly higher than zero over a longer time, there is a bottleneck in the I/O system (network or hard disk). If the *%idle* value is zero over a longer time, your CPU is working at capacity.

2.1.3.1.2 Memory Usage Report: `sar -r`

Generate an overall picture of the system memory (RAM) by using the option `-r`:

```

root # sar -r 10 5
Linux 4.4.21-64-default (jupiter)          10/12/16          _x86_64_          (2 CPU)

17:55:27 kbmemfree kbmemused %memused kbuffers kbcached kbcommit %commit kbactive kbinact kbdirty
17:55:37  104232  1834624  94.62      20   627340  2677656  66.24  802052  828024  1744
17:55:47   98584  1840272  94.92      20   624536  2693936  66.65  808872  826932  2012
17:55:57   87088  1851768  95.51      20   605288  2706392  66.95  827260  821304  1588
17:56:07   86268  1852588  95.55      20   599240  2739224  67.77  829764  820888  3036
17:56:17  104260  1834596  94.62      20   599864  2730688  67.56  811284  821584  3164
Average:   96086  1842770  95.04      20   611254  2709579  67.03  815846  823746  2309

```

The columns *kbcommit* and *%commit* show an approximation of the maximum amount of memory (RAM and swap) that the current workload could need. While *kbcommit* displays the absolute number in kilobytes, *%commit* displays a percentage.

2.1.3.1.3 Paging Statistics Report: `sar -B`

Use the option `-B` to display the kernel paging statistics.

```

root # sar -B 10 5
Linux 4.4.21-64-default (jupiter)          10/12/16          _x86_64_          (2 CPU)

18:23:01 ppggin/s ppggout/s fault/s majflt/s pgfree/s pgscank/s pgscand/s pgsteal/s %vmeff
18:23:11  366.80   11.60  542.50    1.10  4354.80    0.00    0.00    0.00  0.00
18:23:21   0.00   333.30 1522.40    0.00 18132.40    0.00    0.00    0.00  0.00
18:23:31  47.20   127.40 1048.30    0.10 11887.30    0.00    0.00    0.00  0.00

```

18:23:41	46.40	2.50	336.10	0.10	7945.00	0.00	0.00	0.00	0.00
18:23:51	0.00	583.70	2037.20	0.00	17731.90	0.00	0.00	0.00	0.00
Average:	92.08	211.70	1097.30	0.26	12010.28	0.00	0.00	0.00	0.00

The *majflt/s* (major faults per second) column shows how many pages are loaded from disk into memory. The source of the faults may be file accesses or faults. At times, many major faults are normal. For example, during application start-up time. If major faults are experienced for the entire lifetime of the application it may be an indication that there is insufficient main memory, particularly if combined with large amounts of direct scanning (*pgscand/s*).

The *%vmeff* column shows the number of pages scanned (*pgscand/s*) in relation to the ones being reused from the main memory cache or the swap cache (*pgsteal/s*). It is a measurement of the efficiency of page reclaim. Healthy values are either near 100 (every inactive page swapped out is being reused) or 0 (no pages have been scanned). The value should not drop below 30.

2.1.3.1.4 Block Device Statistics Report: `sar -d`

Use the option `-d` to display the block device (hard disk, optical drive, USB storage device, etc.). Make sure to use the additional option `-p` (pretty-print) to make the *DEV* column readable.

```
root # sar -d -p 10 5
Linux 4.4.21-64-default (jupiter)      10/12/16      _x86_64_      (2 CPU)
```

18:46:09 DEV	tps	rd_sec/s	wr_sec/s	avgrq-sz	avgqu-sz	await	svctm	%util
18:46:19 sda	1.70	33.60	0.00	19.76	0.00	0.47	0.47	0.08
18:46:19 sr0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18:46:19 DEV	tps	rd_sec/s	wr_sec/s	avgrq-sz	avgqu-sz	await	svctm	%util
18:46:29 sda	8.60	114.40	518.10	73.55	0.06	7.12	0.93	0.80
18:46:29 sr0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18:46:29 DEV	tps	rd_sec/s	wr_sec/s	avgrq-sz	avgqu-sz	await	svctm	%util
18:46:39 sda	40.50	3800.80	454.90	105.08	0.36	8.86	0.69	2.80
18:46:39 sr0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18:46:39 DEV	tps	rd_sec/s	wr_sec/s	avgrq-sz	avgqu-sz	await	svctm	%util
18:46:49 sda	1.40	0.00	204.90	146.36	0.00	0.29	0.29	0.04
18:46:49 sr0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18:46:49 DEV	tps	rd_sec/s	wr_sec/s	avgrq-sz	avgqu-sz	await	svctm	%util
18:46:59 sda	3.30	0.00	503.80	152.67	0.03	8.12	1.70	0.56
18:46:59 sr0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Average: DEV	tps	rd_sec/s	wr_sec/s	avgrq-sz	avgqu-sz	await	svctm	%util
Average: sda	11.10	789.76	336.34	101.45	0.09	8.07	0.77	0.86
Average: sr0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Compare the *Average* values for *tps*, *rd_sec/s*, and *wr_sec/s* of all disks. Constantly high values in the *svctm* and *%util* columns could be an indication that I/O subsystem is a bottleneck.

If the machine uses multiple disks, then it is best if I/O is interleaved evenly between disks of equal speed and capacity. It will be necessary to take into account whether the storage has multiple tiers. Furthermore, if there are multiple paths to storage then consider what the link saturation will be when balancing how storage is used.

2.1.3.1.5 Network Statistics Reports: **sar -n** *KEYWORD*

The option `-n` lets you generate multiple network related reports. Specify one of the following keywords along with the `-n`:

- *DEV*: Generates a statistic report for all network devices
- *EDEV*: Generates an error statistics report for all network devices
- *NFS*: Generates a statistic report for an NFS client
- *NFSD*: Generates a statistic report for an NFS server
- *SOCK*: Generates a statistic report on sockets
- *ALL*: Generates all network statistic reports

2.1.3.2 Visualizing **sar** Data

`sar` reports are not always easy to parse for humans. kSar, a Java application visualizing your `sar` data, creates easy-to-read graphs. It can even generate PDF reports. kSar takes data generated on the fly and past data from a file. kSar is licensed under the BSD license and is available from <https://sourceforge.net/projects/ksar/>.

2.2 System Information

2.2.1 Device Load Information: **iostat**

To monitor the system device load, use `iostat`. It generates reports that can be useful for better balancing the load between physical disks attached to your system.

To be able to use `iostat`, install the package `sysstat`.

The first **iostat** report shows statistics collected since the system was booted. Subsequent reports cover the time since the previous report.

```
tux > iostat
Linux 4.4.21-64-default (jupiter)          10/12/16      _x86_64_      (4 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           17.68    4.49   4.24   0.29   0.00   73.31

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sdb                 2.02         36.74         45.73     3544894     4412392
sda                 1.05          5.12         13.47       493753     1300276
sdc                 0.02          0.14          0.00        13641         37
```

Invoking **iostat** in this way will help you find out whether throughput is different from your expectation, but not why. Such questions can be better answered by an extended report which can be generated by invoking **iostat -x**. Extended reports additionally include, for example, information on average queue sizes and average wait times. It may also be easier to evaluate the data if idle block devices are excluded using the **-z** switch. Find definitions for each of the displayed column titles in the man page of **iostat** (**man 1 iostat**).

You can also specify that a certain device should be monitored at specified intervals. For example, to generate five reports at three-second intervals for the device **sda**, use:

```
tux > iostat -p sda 3 5
```

To show statistics of network file systems (NFS), there are two similar utilities:

- **nfsiostat-sysstat** is included with the package **sysstat**.
- **nfsiostat** is included with the package **nfs-client**.

2.2.2 Processor Activity Monitoring: **mpstat**

The utility **mpstat** examines activities of each available processor. If your system has one processor only, the global average statistics will be reported.

The timing arguments work the same way as with the **iostat** command. Entering **mpstat 2 5** prints five reports for all processors in two-second intervals.

```
root # mpstat 2 5
Linux 4.4.21-64-default (jupiter)          10/12/16      _x86_64_      (2 CPU)
```

13:51:10	CPU	%usr	%nice	%sys	%iowait	%irq	%soft	%steal	%guest	%gnice	%idle
13:51:12	all	8,27	0,00	0,50	0,00	0,00	0,00	0,00	0,00	0,00	91,23
13:51:14	all	46,62	0,00	3,01	0,00	0,00	0,25	0,00	0,00	0,00	50,13
13:51:16	all	54,71	0,00	3,82	0,00	0,00	0,51	0,00	0,00	0,00	40,97
13:51:18	all	78,77	0,00	5,12	0,00	0,00	0,77	0,00	0,00	0,00	15,35
13:51:20	all	51,65	0,00	4,30	0,00	0,00	0,51	0,00	0,00	0,00	43,54
Average:	all	47,85	0,00	3,34	0,00	0,00	0,40	0,00	0,00	0,00	48,41

From the `mpstat` data, you can see:

- The ratio between the `%usr` and `%sys`. For example, a ratio of 10:1 indicates the workload is mostly running application code and analysis should focus on the application. A ratio of 1:10 indicates the workload is mostly kernel-bound and tuning the kernel is worth considering. Alternatively, determine why the application is kernel-bound and see if that can be alleviated.
- Whether there is a subset of CPUs that are nearly fully utilized even if the system is lightly loaded overall. Few hot CPUs can indicate that the workload is not parallelized and could benefit from executing on a machine with a smaller number of faster processors.

2.2.3 Processor Frequency Monitoring: `turbostat`

`turbostat` shows frequencies, load, temperature, and power of AMD64/Intel 64 processors. It can operate in two modes: If called with a command, the command process is forked and statistics are displayed upon command completion. When run without a command, it will display updated statistics every five seconds. Note that `turbostat` requires the kernel module `msr` to be loaded.

```
tux > sudo turbostat find /etc -type d -exec true {} \;
0.546880 sec
  CPU Avg_MHz  Busy% Bzy_MHz  TSC_MHz
  -      416    28.43   1465    3215
  0      631    37.29   1691    3215
  1      416    27.14   1534    3215
  2      270    24.30   1113    3215
  3      406    26.57   1530    3214
  4      505    32.46   1556    3214
  5      270    22.79   1184    3214
```

The output depends on the CPU type and may vary. To display more details such as temperature and power, use the `--debug` option. For more command line options and an explanation of the field descriptions, refer to `man 8 turbostat`.

2.2.4 Task Monitoring: **pidstat**

If you need to see what load a particular task applies to your system, use **pidstat** command. It prints activity of every selected task or all tasks managed by Linux kernel if no task is specified. You can also set the number of reports to be displayed and the time interval between them.

For example, **pidstat -C firefox 2 3** prints the load statistic for tasks whose command name includes the string “firefox”. There will be three reports printed at two second intervals.

```
root # pidstat -C firefox 2 3
Linux 4.4.21-64-default (jupiter)          10/12/16          _x86_64_          (2 CPU)

14:09:11      UID      PID    %usr %system %guest    %CPU   CPU   Command
14:09:13      1000     387    22,77  0,99   0,00   23,76    1   firefox

14:09:13      UID      PID    %usr %system %guest    %CPU   CPU   Command
14:09:15      1000     387    46,50  3,00   0,00   49,50    1   firefox

14:09:15      UID      PID    %usr %system %guest    %CPU   CPU   Command
14:09:17      1000     387    60,50  7,00   0,00   67,50    1   firefox

Average:      UID      PID    %usr %system %guest    %CPU   CPU   Command
Average:      1000     387    43,19  3,65   0,00   46,84    -   firefox
```

Similarly, **pidstat -d** can be used to estimate how much I/O tasks are doing, whether they are sleeping on that I/O and how many clock ticks the task was stalled.

2.2.5 Kernel Ring Buffer: **dmesg**

The Linux kernel keeps certain messages in a ring buffer. To view these messages, enter the command **dmesg -T**.

Older events are logged in the **systemd** journal. See *Book “Reference”, Chapter 11 “journalctl: Query the systemd Journal”* for more information on the journal.

2.2.6 List of Open Files: **lsof**

To view a list of all the files open for the process with process ID **PID**, use **-p**. For example, to view all the files used by the current shell, enter:

```
root # lsof -p $$
```



```

COMMAND  PID USER  FD   TYPE DEVICE SIZE/OFF  NODE NAME
bash     8842 root   cwd   DIR   0,32    222   6772 /root
bash     8842 root   rtd   DIR   0,32    166    256 /
bash     8842 root   txt   REG   0,32   656584 31066 /bin/bash
bash     8842 root   mem   REG   0,32  1978832 22993 /lib64/libc-2.19.so
[...]
bash     8842 root    2u   CHR  136,2    0t0    5 /dev/pts/2
bash     8842 root   255u  CHR  136,2    0t0    5 /dev/pts/2

```

The special shell variable `$$`, whose value is the process ID of the shell, has been used.

When used with `-i`, `lsuf` lists currently open Internet files as well:

```

root # lsuf -i
COMMAND  PID USER  FD   TYPE DEVICE SIZE/OFF  NODE NAME
wickedd-d 917 root   8u   IPv4  16627    0t0   UDP *:bootpc
wickedd-d 918 root   8u   IPv6  20752    0t0   UDP [fe80::5054:ff:fe72:5ead]:dhcpv6-client
sshd     3152 root   3u   IPv4  18618    0t0   TCP *:ssh (LISTEN)
sshd     3152 root   4u   IPv6  18620    0t0   TCP *:ssh (LISTEN)
master   4746 root   13u  IPv4  20588    0t0   TCP localhost:smtp (LISTEN)
master   4746 root   14u  IPv6  20589    0t0   TCP localhost:smtp (LISTEN)
sshd     8837 root   5u   IPv4  293709   0t0   TCP jupiter.suse.de:ssh->venus.suse.de:33619 (ESTABLISHED)
sshd     8837 root   9u   IPv6  294830   0t0   TCP localhost:x11 (LISTEN)
sshd     8837 root   10u  IPv4  294831   0t0   TCP localhost:x11 (LISTEN)

```

2.2.7 Kernel and udev Event Sequence Viewer: `udevadm monitor`

`udevadm monitor` listens to the kernel uevents and events sent out by a udev rule and prints the device path (DEVPATH) of the event to the console. This is a sequence of events while connecting a USB memory stick:



Note: Monitoring udev Events

Only root user is allowed to monitor udev events by running the `udevadm` command.

```

UEVENT[1138806687] add@/devices/pci0000:00/0000:00:1d.7/usb4/4-2/4-2.2
UEVENT[1138806687] add@/devices/pci0000:00/0000:00:1d.7/usb4/4-2/4-2.2/4-2.2
UEVENT[1138806687] add@/class/scsi_host/host4
UEVENT[1138806687] add@/class/usb_device/usbdev4.10
UDEV [1138806687] add@/devices/pci0000:00/0000:00:1d.7/usb4/4-2/4-2.2
UDEV [1138806687] add@/devices/pci0000:00/0000:00:1d.7/usb4/4-2/4-2.2/4-2.2
UDEV [1138806687] add@/class/scsi_host/host4
UDEV [1138806687] add@/class/usb_device/usbdev4.10
UEVENT[1138806692] add@/devices/pci0000:00/0000:00:1d.7/usb4/4-2/4-2.2/4-2.2
UEVENT[1138806692] add@/block/sdb

```

```

UEVENT[1138806692] add@/class/scsi_generic/sg1
UEVENT[1138806692] add@/class/scsi_device/4:0:0:0
UDEV [1138806693] add@/devices/pci0000:00/0000:00:1d.7/usb4/4-2/4-2.2/4-2.2
UDEV [1138806693] add@/class/scsi_generic/sg1
UDEV [1138806693] add@/class/scsi_device/4:0:0:0
UDEV [1138806693] add@/block/sdb
UEVENT[1138806694] add@/block/sdb/sdb1
UDEV [1138806694] add@/block/sdb/sdb1
UEVENT[1138806694] mount@/block/sdb/sdb1
UEVENT[1138806697] umount@/block/sdb/sdb1

```

2.3 Processes

2.3.1 Interprocess Communication: `ipcs`

The command `ipcs` produces a list of the IPC resources currently in use:

```

root # ipcs
----- Message Queues -----
key          msqid      owner      perms      used-bytes  messages

----- Shared Memory Segments -----
key          shmids    owner      perms      bytes       nattch     status
0x00000000  65536    tux        600        524288     2          dest
0x00000000  98305    tux        600        4194304    2          dest
0x00000000  884738   root       600        524288     2          dest
0x00000000  786435   tux        600        4194304    2          dest
0x00000000  12058628 tux        600        524288     2          dest
0x00000000  917509   root       600        524288     2          dest
0x00000000  12353542 tux        600        196608     2          dest
0x00000000  12451847 tux        600        524288     2          dest
0x00000000  11567114 root       600        262144     1          dest
0x00000000  10911763 tux        600        2097152    2          dest
0x00000000  11665429 root       600        2336768    2          dest
0x00000000  11698198 root       600        196608     2          dest
0x00000000  11730967 root       600        524288     2          dest

----- Semaphore Arrays -----
key          semid     owner      perms      nsems
0xa12e0919  32768    tux        666        2

```

2.3.2 Process List: **ps**

The command **ps** produces a list of processes. Most parameters must be written without a minus sign. Refer to **ps --help** for a brief help or to the man page for extensive help.

To list all processes with user and command line information, use **ps axu**:

```
tux > ps axu
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.3  34376  4608 ?        Ss   Jul24   0:02 /usr/lib/systemd/systemd
root         2  0.0  0.0     0     0 ?        S    Jul24   0:00 [kthreadd]
root         3  0.0  0.0     0     0 ?        S    Jul24   0:00 [ksoftirqd/0]
root         5  0.0  0.0     0     0 ?        S<   Jul24   0:00 [kworker/0:0H]
root         6  0.0  0.0     0     0 ?        S    Jul24   0:00 [kworker/u2:0]
root         7  0.0  0.0     0     0 ?        S    Jul24   0:00 [migration/0]
[...]
tux      12583  0.0  0.1 185980  2720 ?        Sl   10:12   0:00 /usr/lib/gvfs/gvfs-mtp-volume-monitor
tux      12587  0.0  0.1 198132  3044 ?        Sl   10:12   0:00 /usr/lib/gvfs/gvfs-gphoto2-volume-monitor
tux      12591  0.0  0.1 181940  2700 ?        Sl   10:12   0:00 /usr/lib/gvfs/gvfs-goa-volume-monitor
tux      12594  8.1 10.6 1418216 163564 ?       Sl   10:12   0:03 /usr/bin/gnome-shell
tux      12600  0.0  0.3 393448  5972 ?        Sl   10:12   0:00 /usr/lib/gnome-settings-daemon-3.0/gsd-printer
tux      12625  0.0  0.6 227776 10112 ?        Sl   10:12   0:00 /usr/lib/gnome-control-center-search-provider
tux      12626  0.5  1.5 890972 23540 ?        Sl   10:12   0:00 /usr/bin/nautilus --no-default-window
[...]
```

To check how many **sshd** processes are running, use the option **-p** together with the command **pidof**, which lists the process IDs of the given processes.

```
tux > ps -p $(pidof sshd)
  PID TTY      STAT   TIME COMMAND
 1545 ?        Ss     0:00 /usr/sbin/sshd -D
 4608 ?        Ss     0:00 sshd: root@pts/0
```

The process list can be formatted according to your needs. The option **-L** returns a list of all keywords. Enter the following command to issue a list of all processes sorted by memory usage:

```
tux > ps ax --format pid,rss,cmd --sort rss
  PID  RSS CMD
  PID  RSS CMD
    2    0 [kthreadd]
    3    0 [ksoftirqd/0]
    4    0 [kworker/0:0]
    5    0 [kworker/0:0H]
    6    0 [kworker/u2:0]
    7    0 [migration/0]
    8    0 [rcu_bh]
[...]
12518 22996 /usr/lib/gnome-settings-daemon-3.0/gnome-settings-daemon
12626 23540 /usr/bin/nautilus --no-default-window
```

```
12305 32188 /usr/bin/Xorg :0 -background none -verbose
12594 164900 /usr/bin/gnome-shell
```

USEFUL ps CALLS

ps aux --sort COLUMN

Sort the output by COLUMN. Replace COLUMN with

pmem for physical memory ratio

pcpu for CPU ratio

rss for resident set size (non-swapped physical memory)

ps axo pid,%cpu, rss, vsz, args, wchan

Shows every process, their PID, CPU usage ratio, memory size (resident and virtual), name, and their syscall.

ps axfo pid, args

Show a process tree.

2.3.3 Process Tree: **ps tree**

The command **ps tree** produces a list of processes in the form of a tree:

```
tux > ps tree
systemd---accounts-daemon---{gdbus}
      |
      |---{gmain}
      |---at-spi-bus-laun---dbus-daemon
      |
      |---{dconf worker}
      |
      |---{gdbus}
      |
      |---{gmain}
      |---at-spi2-registr---{gdbus}
      |---cron
      |---2*[dbus-daemon]
      |---dbus-launch
      |---dconf-service---{gdbus}
      |
      |---{gmain}
      |---gconfd-2
      |---gdm---gdm-simple-slav---Xorg
      |   |
      |   |---gdm-session-wor---gnome-session---gnome-setti+
      |   |
      |   |---{gdbus}
      |   |
      |   |---{gmain}
      |   |
      |   |---{gdbus}
      |   |
      |   |---{gmain}
```

```

|      |      |      | -{gdbus}
|      |      |      | -{gmain}
|      | -{gdbus}
|      | -{gmain}
[...]
```

The parameter `-p` adds the process ID to a given name. To have the command lines displayed as well, use the `-a` parameter:

2.3.4 Table of Processes: `top`

The command `top` (an abbreviation of “table of processes”) displays a list of processes that is refreshed every two seconds. To terminate the program, press `Q`. The parameter `-n 1` terminates the program after a single display of the process list. The following is an example output of the command `top -n 1`:

```

tux > top -n 1
Tasks: 128 total, 1 running, 127 sleeping, 0 stopped, 0 zombie
%Cpu(s): 2.4 us, 1.2 sy, 0.0 ni, 96.3 id, 0.1 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 1535508 total, 699948 used, 835560 free, 880 buffers
KiB Swap: 1541116 total, 0 used, 1541116 free. 377000 cached Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM     TIME+ COMMAND
    1 root        20   0 116292  4660 2028  S   0.000 0.303   0:04.45 systemd
    2 root        20   0     0     0     0   S   0.000 0.000   0:00.00 kthreadd
    3 root        20   0     0     0     0   S   0.000 0.000   0:00.07 ksoftirqd+
    5 root         0 -20     0     0     0   S   0.000 0.000   0:00.00 kworker/0+
    6 root        20   0     0     0     0   S   0.000 0.000   0:00.00 kworker/u+
    7 root        rt    0     0     0     0   S   0.000 0.000   0:00.00 migration+
    8 root        20   0     0     0     0   S   0.000 0.000   0:00.00 rcu_bh
    9 root        20   0     0     0     0   S   0.000 0.000   0:00.24 rcu_sched
   10 root        rt    0     0     0     0   S   0.000 0.000   0:00.01 watchdog/0
   11 root         0 -20     0     0     0   S   0.000 0.000   0:00.00 khelper
   12 root        20   0     0     0     0   S   0.000 0.000   0:00.00 kdevtmpfs
   13 root         0 -20     0     0     0   S   0.000 0.000   0:00.00 netns
   14 root         0 -20     0     0     0   S   0.000 0.000   0:00.00 writeback
   15 root         0 -20     0     0     0   S   0.000 0.000   0:00.00 kintegrit+
   16 root         0 -20     0     0     0   S   0.000 0.000   0:00.00 bioset
   17 root         0 -20     0     0     0   S   0.000 0.000   0:00.00 crypto
   18 root         0 -20     0     0     0   S   0.000 0.000   0:00.00 kblockd
```

By default the output is sorted by CPU usage (column `%CPU`, shortcut `Shift-P`). Use the following key combinations to change the sort field:

`Shift-M`: Resident Memory (`RES`)

Shift-N: Process ID (*PID*)

Shift-T: Time (*TIME* +)

To use any other field for sorting, press **F** and select a field from the list. To toggle the sort order, Use **Shift-R**.

The parameter `-U UID` monitors only the processes associated with a particular user. Replace *UID* with the user ID of the user. Use `top -U $(id -u)` to show processes of the current user

2.3.5 A top-like I/O Monitor: **iotop**

The **iotop** utility displays a table of I/O usage by processes or threads.



Note: Installing **iotop**

iotop is not installed by default. You need to install it manually with `zypper in iotop` as `root`.

iotop displays columns for the I/O bandwidth read and written by each process during the sampling period. It also displays the percentage of time the process spent while swapping in and while waiting on I/O. For each process, its I/O priority (class/level) is shown. In addition, the total I/O bandwidth read and written during the sampling period is displayed at the top of the interface.

- The **←** and **→** keys change the sorting.
- **R** reverses the sort order.
- **O** toggles between showing all processes and threads (default view) and showing only those doing I/O. (This function is similar to adding `--only` on command line.)
- **P** toggles between showing threads (default view) and processes. (This function is similar to `--only`.)
- **A** toggles between showing the current I/O bandwidth (default view) and accumulated I/O operations since **iotop** was started. (This function is similar to `--accumulated`.)
- **I** lets you change the priority of a thread or a process's threads.
- **Q** quits **iotop**.
- Pressing any other key will force a refresh.

Following is an example output of the command `iotop --only`, while `find` and `emacs` are running:

```
root # iotop --only
Total DISK READ: 50.61 K/s | Total DISK WRITE: 11.68 K/s
  TID  PRI0  USER      DISK READ  DISK WRITE  SWAPIN      IO>    COMMAND
 3416 be/4  tux        50.61 K/s   0.00 B/s   0.00 %    4.05 % find /
  275 be/3  root       0.00 B/s   3.89 K/s   0.00 %    2.34 % [jbd2/sda2-8]
 5055 be/4  tux        0.00 B/s   3.89 K/s   0.00 %    0.04 % emacs
```

`iotop` can be also used in a batch mode (`-b`) and its output stored in a file for later analysis. For a complete set of options, see the manual page (`man 8 iotop`).

2.3.6 Modify a process's niceness: `nice` and `renice`

The kernel determines which processes require more CPU time than others by the process's nice level, also called niceness. The higher the “nice” level of a process is, the less CPU time it will take from other processes. Nice levels range from -20 (the least “nice” level) to 19. Negative values can only be set by `root`.

Adjusting the niceness level is useful when running a non time-critical process that lasts long and uses large amounts of CPU time. For example, compiling a kernel on a system that also performs other tasks. Making such a process “nicer”, ensures that the other tasks, for example a Web server, will have a higher priority.

Calling `nice` without any parameters prints the current niceness:

```
tux > nice
0
```

Running `nice COMMAND` increments the current nice level for the given command by 10. Using `nice -n LEVEL COMMAND` lets you specify a new niceness relative to the current one.

To change the niceness of a running process, use `renice PRIORITY -p PROCESS_ID`, for example:

```
tux > renice +5 3266
```

To `renice` all processes owned by a specific user, use the option `-u USER`. Process groups are `reniced` by the option `-g PROCESS_GROUP_ID`.

2.4 Memory

2.4.1 Memory Usage: `free`

The utility `free` examines RAM and swap usage. Details of both free and used memory and swap areas are shown:

```
tux > free
```

	total	used	free	shared	buffers	cached
Mem:	32900500	32703448	197052	0	255668	5787364
-/+ buffers/cache:		26660416	6240084			
Swap:	2046972	304680	1742292			

The options `-b`, `-k`, `-m`, `-g` show the output in bytes, KB, MB, or GB, respectively. The parameter `-s delay` ensures that the display is refreshed every `DELAY` seconds. For example, `free -s 1.5` produces an update every 1.5 seconds.

2.4.2 Detailed Memory Usage: `/proc/meminfo`

Use `/proc/meminfo` to get more detailed information on memory usage than with `free`. Actually `free` uses some data from this file. See an example output from a 64-bit system below. Note that it slightly differs on 32-bit systems because of different memory management:

```
MemTotal:      1942636 kB
MemFree:       1294352 kB
MemAvailable:  1458744 kB
Buffers:       876 kB
Cached:        278476 kB
SwapCached:    0 kB
Active:        368328 kB
Inactive:      199368 kB
Active(anon):  288968 kB
Inactive(anon): 10568 kB
Active(file):  79360 kB
Inactive(file): 188800 kB
Unevictable:   80 kB
Mlocked:       80 kB
SwapTotal:     2103292 kB
SwapFree:      2103292 kB
Dirty:         44 kB
Writeback:     0 kB
```



```

AnonPages:      288592 kB
Mapped:         70444 kB
Shmem:          11192 kB
Slab:           40916 kB
SReclaimable:  17712 kB
SUnreclaim:    23204 kB
KernelStack:   2000 kB
PageTables:    10996 kB
NFS_Unstable:  0 kB
Bounce:        0 kB
WritebackTmp:  0 kB
CommitLimit:   3074608 kB
Committed_AS:  1407208 kB
VmallocTotal:  34359738367 kB
VmallocUsed:   145996 kB
VmallocChunk:  34359588844 kB
HardwareCorrupted: 0 kB
AnonHugePages: 86016 kB
HugePages_Total: 0
HugePages_Free: 0
HugePages_Rsvd: 0
HugePages_Surp: 0
Hugepagesize:  2048 kB
DirectMap4k:   79744 kB
DirectMap2M:  2017280 kB

```

These entries stand for the following:

MemTotal

Total amount of RAM.

MemFree

Amount of unused RAM.

MemAvailable

Estimate of how much memory is available for starting new applications without swapping.

Buffers

File buffer cache in RAM containing file system metadata.

Cached

Page cache in RAM. This excludes buffer cache and swap cache, but includes *Shmem* memory.

SwapCached

Page cache for swapped-out memory.

Active, Active(anon), Active(file)

Recently used memory that will not be reclaimed unless necessary or on explicit request.

Active is the sum of *Active(anon)* and *Active(file)*:

- *Active(anon)* tracks swap-backed memory. This includes private and shared anonymous mappings and private file pages after copy-on-write.
- *Active(file)* tracks other file system backed memory.

Inactive, Inactive(anon), Inactive(file)

Less recently used memory that will usually be reclaimed first. *Inactive* is the sum of *Inactive(anon)* and *Inactive(file)*:

- *Inactive(anon)* tracks swap backed memory. This includes private and shared anonymous mappings and private file pages after copy-on-write.
- *Inactive(file)* tracks other file system backed memory.

Unevictable

Amount of memory that cannot be reclaimed (for example, because it is *Mlocked* or used as a RAM disk).

Mlocked

Amount of memory that is backed by the `mlock` system call. `mlock` allows processes to define which part of physical RAM their virtual memory should be mapped to. However, `mlock` does not guarantee this placement.

SwapTotal

Amount of swap space.

SwapFree

Amount of unused swap space.

Dirty

Amount of memory waiting to be written to disk, because it contains changes compared to the backing storage. Dirty data can be explicitly synchronized either by the application or by the kernel after a short delay. A large amount of dirty data may take considerable time to write to disk resulting in stalls. The total amount of dirty data that can exist at any time can be controlled with the `sysctl` parameters `vm.dirty_ratio` or `vm.dirty_bytes` (refer to [Section 14.1.5, "Writeback"](#) for more details).

Writeback

Amount of memory that is currently being written to disk.

Mapped

Memory claimed with the `mmap` system call.

Shmem

Memory shared between groups of processes, such as IPC data, `tmpfs` data, and shared anonymous memory.

Slab

Memory allocation for internal data structures of the kernel.

SReclaimable

Slab section that can be reclaimed, such as caches (inode, dentry, etc.).

SUnreclaim

Slab section that cannot be reclaimed.

KernelStack

Amount of kernel space memory used by applications (through system calls).

PageTables

Amount of memory dedicated to page tables of all processes.

NFS_Unstable

NFS pages that have already been sent to the server, but are not yet committed there.

Bounce

Memory used for bounce buffers of block devices.

WritebackTmp

Memory used by FUSE for temporary writeback buffers.

CommitLimit

Amount of memory available to the system based on the overcommit ratio setting. This is only enforced if strict overcommit accounting is enabled.

Committed_AS

An approximation of the total amount of memory (RAM and swap) that the current workload would need in the worst case.

VmallocTotal

Amount of allocated kernel virtual address space.

VmallocUsed

Amount of used kernel virtual address space.

VmallocChunk

The largest contiguous block of available kernel virtual address space.

HardwareCorrupted

Amount of failed memory (can only be detected when using ECC RAM).

AnonHugePages

Anonymous hugepages that are mapped into user space page tables. These are allocated transparently for processes without being specifically requested, therefore they are also known as *transparent hugepages* (THP).

HugePages_Total

Number of preallocated hugepages for use by SHM_HUGETLB and MAP_HUGETLB or through the hugetlbfs file system, as defined in /proc/sys/vm/nr_hugepages.

HugePages_Free

Number of hugepages available.

HugePages_Rsvd

Number of hugepages that are committed.

HugePages_Surp

Number of hugepages available beyond *HugePages_Total* (“surplus”), as defined in /proc/sys/vm/nr_overcommit_hugepages.

Hugepagesize

Size of a hugepage—on AMD64/Intel 64 the default is 2048 KB.

DirectMap4k etc.

Amount of kernel memory that is mapped to pages with a given size (in the example: 4 kB).

2.4.3 Process Memory Usage: smaps

Exactly determining how much memory a certain process is consuming is not possible with standard tools like **top** or **ps**. Use the smaps subsystem, introduced in kernel 2.6.14, if you need exact data. It can be found at /proc/PID/smaps and shows you the number of clean and dirty memory pages the process with the ID PID is using at that time. It differentiates

between shared and private memory, so you can see how much memory the process is using without including memory shared with other processes. For more information see </usr/src/linux/Documentation/filesystems/proc.txt> (requires the package `kernel-source` to be installed).

`smaps` is expensive to read. Therefore it is not recommended to monitor it regularly, but only when closely monitoring a certain process.

2.4.4 numaTOP

numaTOP is a tool for NUMA (Non_uniform Memory Access) systems. The tool helps to identify NUMA-related performance bottlenecks by providing real-time analysis of a NUMA system.

Generally speaking, numaTOP allows you to identify and investigate processes and threads with poor locality (that is poor ratio of local versus remote memory usage) by analyzing number of Remote Memory Accesses (RMA), number of Local Memory Accesses (LMA), and RMA/LMA ratio.

numaTOP is supported on PowerPC and the following Intel Xeon processors: 5500-series, 6500/7500-series, 5600 series, E7-x8xx-series, and E5-16xx/24xx/26xx/46xx-series.

numaTOP is available in the official software repositories, and you can install the tool using the `sudo zypper in numatop` command. To launch numaTOP, run the `numatop` command. To get an overview of numaTOP functionality and usage, use the `man numatop` command.

2.5 Networking



Tip: Traffic Shaping

In case the network bandwidth is lower than expected, you should first check if any traffic shaping rules are active for your network segment.

2.5.1 Basic Network Diagnostics: `ip`

`ip` is a powerful tool to set up and control network interfaces. You can also use it to quickly view basic statistics about network interfaces of the system. For example, whether the interface is up or how many errors, dropped packets, or packet collisions there are.

If you run **ip** with no additional parameter, it displays a help output. To list all network interfaces, enter **ip addr show** (or abbreviated as **ip a**). **ip addr show up** lists only running network interfaces. **ip -s link show DEVICE** lists statistics for the specified interface only:

```
root # ip -s link show br0
6: br0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT
    link/ether 00:19:d1:72:d4:30 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped  overrun  mcast
    6346104756 9265517  0      10860   0        0
    TX: bytes  packets  errors  dropped  carrier  collsns
    3996204683 3655523  0      0       0        0
```

ip can also show interfaces (link), routing tables (route), and much more—refer to man 8 ip for details.

```
root # ip route
default via 192.168.2.1 dev eth1
192.168.2.0/24 dev eth0 proto kernel scope link src 192.168.2.100
192.168.2.0/24 dev eth1 proto kernel scope link src 192.168.2.101
192.168.2.0/24 dev eth2 proto kernel scope link src 192.168.2.102
```

```
root # ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 52:54:00:44:30:51 brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 52:54:00:a3:c1:fb brd ff:ff:ff:ff:ff:ff
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 52:54:00:32:a4:09 brd ff:ff:ff:ff:ff:ff
```

2.5.2 Show the Network Usage of Processes: **nethogs**

In some cases, for example if the network traffic suddenly becomes very high, it is desirable to quickly find out which application(s) is/are causing the traffic. **nethogs**, a tool with a design similar to top, shows incoming and outgoing traffic for all relevant processes:

PID	USER	PROGRAM	DEV	SENT	RECEIVED
27145	root	zypper	eth0	5.719	391.749 KB/sec
?	root	..0:113:80c0:8080:10:160:0:100:30015		0.102	2.326 KB/sec
26635	tux	/usr/lib64/firefox/firefox	eth0	0.026	0.026 KB/sec
?	root	..0:113:80c0:8080:10:160:0:100:30045		0.000	0.021 KB/sec

?	root	..0:113:80c0:8080:10:160:0:100:30045	0.000	0.018 KB/sec
?	root	..0:113:80c0:8080:10:160:0:100:30015	0.000	0.018 KB/sec
?	root	..0:113:80c0:8080:10:160:0:100:30045	0.000	0.017 KB/sec
?	root	..0:113:80c0:8080:10:160:0:100:30045	0.000	0.017 KB/sec
?	root	..0:113:80c0:8080:10:160:0:100:30045	0.069	0.000 KB/sec
?	root	unknown TCP	0.000	0.000 KB/sec
TOTAL			5.916	394.192 KB/sec

Like in top, nethogs features interactive commands:

[M]: cycle between display modes (kb/s, kb, b, mb)

[R]: sort by *RECEIVED*

[S]: sort by *SENT*

[Q]: quit

2.5.3 Ethernet Cards in Detail: ethtool

ethtool can display and change detailed aspects of your Ethernet network device. By default it prints the current setting of the specified device.

```
root # ethtool eth0
Settings for eth0:
Supported ports: [ TP ]
Supported link modes:   10baseT/Half 10baseT/Full
                       100baseT/Half 100baseT/Full
                       1000baseT/Full
Supports auto-negotiation: Yes
Advertised link modes:  10baseT/Half 10baseT/Full
                       100baseT/Half 100baseT/Full
                       1000baseT/Full
Advertised pause frame use: No
[...]
Link detected: yes
```

The following table shows **ethtool** options that you can use to query the device for specific information:

TABLE 2.1: LIST OF QUERY OPTIONS OF **ethtool**

ethtool option	it queries the device for
-a	pause parameter information

<u>ethtool</u> option	it queries the device for
-c	interrupt coalescing information
-g	Rx/Tx (receive/transmit) ring parameter information
-i	associated driver information
-k	offload information
-S	NIC and driver-specific statistics

2.5.4 Show the Network Status: **ss**

ss is a tool to dump socket statistics and replaces the **netstat** command. To list all connections use **ss** without parameters:

```

root # ss
Netid State      Recv-Q Send-Q   Local Address:Port      Peer Address:Port
u_str ESTAB      0      0      * 14082                  * 14083
u_str ESTAB      0      0      * 18582                  * 18583
u_str ESTAB      0      0      * 19449                  * 19450
u_str ESTAB      0      0      @/tmp/dbus-gmUUwXABPV 18784 * 18783
u_str ESTAB      0      0      /var/run/dbus/system_bus_socket 19383 * 19382
u_str ESTAB      0      0      @/tmp/dbus-gmUUwXABPV 18617 * 18616
u_str ESTAB      0      0      @/tmp/dbus-58TPPDv8qv 19352 * 19351
u_str ESTAB      0      0      * 17658                  * 17657
u_str ESTAB      0      0      * 17693                  * 17694
[..]

```

To show all network ports currently open, use the following command:

```

root # ss -l
Netid State      Recv-Q Send-Q   Local Address:Port      Peer Address:Port
nl     UNCONN    0      0      rtnl:4195117            *
nl     UNCONN    0      0      rtnl:wickedd-auto4/811 *
nl     UNCONN    0      0      rtnl:wickedd-dhcp4/813 *
nl     UNCONN    0      0      rtnl:4195121            *
nl     UNCONN    0      0      rtnl:4195115            *
nl     UNCONN    0      0      rtnl:wickedd-dhcp6/814 *
nl     UNCONN    0      0      rtnl:kernel              *
nl     UNCONN    0      0      rtnl:wickedd/817        *

```



```
n1 UNCONN 0 0 rtnl:4195118 *
n1 UNCONN 0 0 rtnl:nscd/706 *
n1 UNCONN 4352 0 tcpdiag:ss/2381 *
[...]
```

When displaying network connections, you can specify the socket type to display: TCP (-t) or UDP (-u) for example. The -p option shows the PID and name of the program to which each socket belongs.

The following example lists all TCP connections and the programs using these connections. The -a option make sure all established connections (listening and non-listening) are shown. The -p option shows the PID and name of the program to which each socket belongs.

```
root # ss -t -a -p
State Recv-Q Send-Q Local Address:Port Peer Address:Port
LISTEN 0 128 *:ssh *:users(("sshd",1551,3))
LISTEN 0 100 127.0.0.1:smtp *:users(("master",1704,13))
ESTAB 0 132 10.120.65.198:ssh 10.120.4.150:55715 users(("sshd",2103,5))
LISTEN 0 128 :::ssh :::users(("sshd",1551,4))
LISTEN 0 100 :::1:smtp :::users(("master",1704,14))
```

2.6 The /proc File System

The /proc file system is a pseudo file system in which the kernel reserves important information in the form of virtual files. For example, display the CPU type with this command:

```
tux > cat /proc/cpuinfo
processor      : 0
vendor_id    : GenuineIntel
cpu family   : 6
model        : 30
model name   : Intel(R) Core(TM) i5 CPU          750 @ 2.67GHz
stepping     : 5
microcode    : 0x6
cpu MHz      : 1197.000
cache size   : 8192 KB
physical id  : 0
siblings     : 4
core id      : 0
cpu cores    : 4
apicid       : 0
initial apicid : 0
fpu          : yes
fpu_exception : yes
cpuid level  : 11
```

```

wp                : yes
flags             : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx rdtscp lm constant_tsc
arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf pni dtes64 monitor
ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm sse4_1 sse4_2 popcnt lahf_lm ida dtherm
tpr_shadow vnmi flexpriority ept vpid
bogomips         : 5333.85
clflush size     : 64
cache_alignment  : 64
address sizes    : 36 bits physical, 48 bits virtual
power management:
[...]

```



Tip: Detailed Processor Information

Detailed information about the processor on the AMD64/Intel 64 architecture is also available by running **x86info**.

Query the allocation and use of interrupts with the following command:

```

tux > cat /proc/interrupts

```

	CPU0	CPU1	CPU2	CPU3		
0:	121	0	0	0	IO-APIC-edge	timer
8:	0	0	0	1	IO-APIC-edge	rtc0
9:	0	0	0	0	IO-APIC-fasteoi	acpi
16:	0	11933	0	0	IO-APIC-fasteoi	ehci_hcd:+
18:	0	0	0	0	IO-APIC-fasteoi	i801_smbus
19:	0	117978	0	0	IO-APIC-fasteoi	ata_piix,+
22:	0	0	3275185	0	IO-APIC-fasteoi	enp5s1
23:	417927	0	0	0	IO-APIC-fasteoi	ehci_hcd:+
40:	2727916	0	0	0	HPET_MSI-edge	hpet2
41:	0	2749134	0	0	HPET_MSI-edge	hpet3
42:	0	0	2759148	0	HPET_MSI-edge	hpet4
43:	0	0	0	2678206	HPET_MSI-edge	hpet5
45:	0	0	0	0	PCI-MSI-edge	aerdrv, P+
46:	0	0	0	0	PCI-MSI-edge	PCIe PME,+
47:	0	0	0	0	PCI-MSI-edge	PCIe PME,+
48:	0	0	0	0	PCI-MSI-edge	PCIe PME,+
49:	0	0	0	387	PCI-MSI-edge	snd_hda_i+
50:	933117	0	0	0	PCI-MSI-edge	nvidia
NMI:	2102	2023	2031	1920	Non-maskable interrupts	
LOC:	92	71	57	41	Local timer interrupts	
SPU:	0	0	0	0	Spurious interrupts	
PMI:	2102	2023	2031	1920	Performance monitoring int+	
IWI:	47331	45725	52464	46775	IRQ work interrupts	

RTR:	2	0	0	0	APIC ICR read retries
RES:	472911	396463	339792	323820	Rescheduling interrupts
CAL:	48389	47345	54113	50478	Function call interrupts
TLB:	28410	26804	24389	26157	TLB shootdowns
TRM:	0	0	0	0	Thermal event interrupts
THR:	0	0	0	0	Threshold APIC interrupts
MCE:	0	0	0	0	Machine check exceptions
MCP:	40	40	40	40	Machine check polls
ERR:	0				
MIS:	0				

The address assignment of executables and libraries is contained in the maps file:

```
tux > cat /proc/self/maps
08048000-0804c000 r-xp 00000000 03:03 17753      /bin/cat
0804c000-0804d000 rw-p 00004000 03:03 17753      /bin/cat
0804d000-0806e000 rw-p 0804d000 00:00 0          [heap]
b7d27000-b7d5a000 r--p 00000000 03:03 11867      /usr/lib/locale/en_GB.utf8/
b7d5a000-b7e32000 r--p 00000000 03:03 11868      /usr/lib/locale/en_GB.utf8/
b7e32000-b7e33000 rw-p b7e32000 00:00 0
b7e33000-b7f45000 r-xp 00000000 03:03 8837       /lib/libc-2.3.6.so
b7f45000-b7f46000 r--p 00112000 03:03 8837       /lib/libc-2.3.6.so
b7f46000-b7f48000 rw-p 00113000 03:03 8837       /lib/libc-2.3.6.so
b7f48000-b7f4c000 rw-p b7f48000 00:00 0
b7f52000-b7f53000 r--p 00000000 03:03 11842      /usr/lib/locale/en_GB.utf8/
[...]
b7f5b000-b7f61000 r--s 00000000 03:03 9109       /usr/lib/gconv/gconv-module
b7f61000-b7f62000 r--p 00000000 03:03 9720       /usr/lib/locale/en_GB.utf8/
b7f62000-b7f76000 r-xp 00000000 03:03 8828       /lib/ld-2.3.6.so
b7f76000-b7f78000 rw-p 00013000 03:03 8828       /lib/ld-2.3.6.so
bfd61000-bfd76000 rw-p bfd61000 00:00 0          [stack]
ffffe000-fffff000 ---p 00000000 00:00 0          [vdso]
```

A lot more information can be obtained from the `/proc` file system. Some important files and their contents are:

/proc/devices

Available devices

/proc/modules

Kernel modules loaded

/proc/cmdline

Kernel command line

/proc/meminfo

Detailed information about memory usage

/proc/config.gz

gzip-compressed configuration file of the kernel currently running

/proc/PID/

Find information about processes currently running in the /proc/NNN directories, where NNN is the process ID (PID) of the relevant process. Every process can find its own characteristics in /proc/self/.

Further information is available in the text file /usr/src/linux/Documentation/filesystems/proc.txt (this file is available when the package kernel-source is installed).

2.6.1 **procinfo**

Important information from the /proc file system is summarized by the command **procinfo**:

```
tux > procinfo
Linux 3.11.10-17-desktop (geeko@buildhost) (gcc 4.8.1 20130909) #1 4CPU
[jupiter.example.com]

Memory:      Total      Used      Free      Shared    Buffers    Cached
Mem:         8181908   8000632   181276    0         85472     2850872
Swap:        10481660   1576     10480084

Bootup: Mon Jul 28 09:54:13 2014    Load average: 1.61 0.85 0.74 2/904 25949

user  :      1:54:41.84  12.7%  page in :    2107312  disk 1:    52212r   20199w
nice  :      0:00:00.46  0.0%  page out:    1714461  disk 2:    19387r   10928w
system:    0:25:38.00  2.8%  page act:    466673   disk 3:     548r    10w
IOWait:    0:04:16.45  0.4%  page dea:    272297
hw irq:    0:00:00.42  0.0%  page flt: 105754526
sw irq:    0:01:26.48  0.1%  swap in :      0
idle  :    12:14:43.65 81.5%  swap out:    394
guest  :    0:02:18.59  0.2%
uptime:    3:45:22.24          context : 99809844

irq 0:      121 timer          irq 41: 3238224 hpet3
irq 8:        1 rtc0          irq 42: 3251898 hpet4
irq 9:         0 acpi          irq 43: 3156368 hpet5
irq 16: 14589 ehci_hcd:usb1    irq 45:      0 aerdrv, PCIe PME
irq 18:         0 i801_smbus    irq 46:      0 PCIe PME, pciehp
irq 19: 124861 ata_piix, ata_piix, f irq 47:      0 PCIe PME, pciehp
irq 22: 3742817 enp5s1          irq 48:      0 PCIe PME, pciehp
```

```
irq 23:    479248 ehci_hcd:usb2      irq 49:    387 snd_hda_intel
irq 40:    3216894 hpet2           irq 50:    1088673 nvidia
```

To see all the information, use the parameter `-a`. The parameter `-nN` produces updates of the information every `N` seconds. In this case, terminate the program by pressing `Q`.

By default, the cumulative values are displayed. The parameter `-d` produces the differential values. `procinfo -dn5` displays the values that have changed in the last five seconds:

2.6.2 System Control Parameters: `/proc/sys/`

System control parameters are used to modify the Linux kernel parameters at runtime. They reside in `/proc/sys/` and can be viewed and modified with the `sysctl` command. To list all parameters, run `sysctl -a`. A single parameter can be listed with `sysctl PARAMETER_NAME`. Parameters are grouped into categories and can be listed with `sysctl CATEGORY` or by listing the contents of the respective directories. The most important categories are listed below. The links to further readings require the installation of the package `kernel-source`.

`sysctl dev (/proc/sys/dev/)`

Device-specific information.

`sysctl fs (/proc/sys/fs/)`

Used file handles, quotas, and other file system-oriented parameters. For details see </usr/src/linux/Documentation/sysctl/fs.txt>.

`sysctl kernel (/proc/sys/kernel/)`

Information about the task scheduler, system shared memory, and other kernel-related parameters. For details see </usr/src/linux/Documentation/sysctl/kernel.txt>

`sysctl net (/proc/sys/net/)`

Information about network bridges, and general network parameters (mainly the `ipv4/` subdirectory). For details see </usr/src/linux/Documentation/sysctl/net.txt>

`sysctl vm (/proc/sys/vm/)`

Entries in this path relate to information about the virtual memory, swapping, and caching. For details see </usr/src/linux/Documentation/sysctl/vm.txt>

To set or change a parameter for the current session, use the command `sysctl -w PARAMETER = VALUE`. To permanently change a setting, add a line `PARAMETER = VALUE` to `/etc/sysctl.conf`.

2.7 Hardware Information

2.7.1 PCI Resources: `lspci`



Note: Accessing PCI configuration.

Most operating systems require root user privileges to grant access to the computer's PCI configuration.

The command `lspci` lists the PCI resources:

```
root # lspci
00:00.0 Host bridge: Intel Corporation 82845G/GL[Brookdale-G]/GE/PE \
  DRAM Controller/Host-Hub Interface (rev 01)
00:01.0 PCI bridge: Intel Corporation 82845G/GL[Brookdale-G]/GE/PE \
  Host-to-AGP Bridge (rev 01)
00:1d.0 USB Controller: Intel Corporation 82801DB/DBL/DBM \
  (ICH4/ICH4-L/ICH4-M) USB UHCI Controller #1 (rev 01)
00:1d.1 USB Controller: Intel Corporation 82801DB/DBL/DBM \
  (ICH4/ICH4-L/ICH4-M) USB UHCI Controller #2 (rev 01)
00:1d.2 USB Controller: Intel Corporation 82801DB/DBL/DBM \
  (ICH4/ICH4-L/ICH4-M) USB UHCI Controller #3 (rev 01)
00:1d.7 USB Controller: Intel Corporation 82801DB/DBM \
  (ICH4/ICH4-M) USB2 EHCI Controller (rev 01)
00:1e.0 PCI bridge: Intel Corporation 82801 PCI Bridge (rev 81)
00:1f.0 ISA bridge: Intel Corporation 82801DB/DBL (ICH4/ICH4-L) \
  LPC Interface Bridge (rev 01)
00:1f.1 IDE interface: Intel Corporation 82801DB (ICH4) IDE \
  Controller (rev 01)
00:1f.3 SMBus: Intel Corporation 82801DB/DBL/DBM (ICH4/ICH4-L/ICH4-M) \
  SMBus Controller (rev 01)
00:1f.5 Multimedia audio controller: Intel Corporation 82801DB/DBL/DBM \
  (ICH4/ICH4-L/ICH4-M) AC'97 Audio Controller (rev 01)
01:00.0 VGA compatible controller: Matrox Graphics, Inc. G400/G450 (rev 85)
02:08.0 Ethernet controller: Intel Corporation 82801DB PRO/100 VE (LOM) \
  Ethernet Controller (rev 81)
```

Using `-v` results in a more detailed listing:

```
root # lspci -v
[...]
00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet \
  Controller (rev 02)
```

```
Subsystem: Intel Corporation PRO/1000 MT Desktop Adapter
Flags: bus master, 66MHz, medium devsel, latency 64, IRQ 19
Memory at f0000000 (32-bit, non-prefetchable) [size=128K]
I/O ports at d010 [size=8]
Capabilities: [dc] Power Management version 2
Capabilities: [e4] PCI-X non-bridge device
Kernel driver in use: e1000
Kernel modules: e1000
```

Information about device name resolution is obtained from the file `/usr/share/pci.ids`. PCI IDs not listed in this file are marked “Unknown device.”

The parameter `-vv` produces all the information that could be queried by the program. To view the pure numeric values, use the parameter `-n`.

2.7.2 USB Devices: `lsusb`

The command `lsusb` lists all USB devices. With the option `-v`, print a more detailed list. The detailed information is read from the directory `/proc/bus/usb/`. The following is the output of `lsusb` with these USB devices attached: hub, memory stick, hard disk and mouse.

```
root # lsusb
Bus 004 Device 007: ID 0ea0:2168 Ours Technology, Inc. Transcend JetFlash \
  2.0 / Astone USB Drive
Bus 004 Device 006: ID 04b4:6830 Cypress Semiconductor Corp. USB-2.0 IDE \
  Adapter
Bus 004 Device 005: ID 05e3:0605 Genesys Logic, Inc.
Bus 004 Device 001: ID 0000:0000
Bus 003 Device 001: ID 0000:0000
Bus 002 Device 001: ID 0000:0000
Bus 001 Device 005: ID 046d:c012 Logitech, Inc. Optical Mouse
Bus 001 Device 001: ID 0000:0000
```

2.7.3 Monitoring and Tuning the Thermal Subsystem: `tmon`

`tmon` is a tool to help visualize, tune, and test the complex thermal subsystem. When started without parameters, `tmon` runs in monitoring mode:

```
┌─── THERMAL ZONES (SENSORS) ───┐
| Thermal Zones:                acpitz00 |
| Trip Points:                  PC       |
└────────────────────────────────┘

┌─── COOLING DEVICES ───┐
```

```

|ID  Cooling Dev  Cur   Max  Thermal Zone Binding |
|00  Processor    0     3   |||||
|01  Processor    0     3   |||||
|02  Processor    0     3   |||||
|03  Processor    0     3   |||||
|04 intel_powerc -1    50   |||||

          10     20     30     40
|acpitz 0:[ 8][>>>>>>>>>P9          C31

----- CONTROLS -----
|PID gain: kp=0.36 ki=5.00 kd=0.19 Output 0.00
|Target Temp: 65.0C, Zone: 0, Control Device: None

Ctrl-c - Quit  TAB - Tuning

```

For detailed information on how to interpret the data, how to log thermal data and how to use **tmon** to test and tune cooling devices and sensors, refer to the man page: **man 8 tmon**. The package **tmon** is not installed by default.

2.7.4 MCELog: Machine Check Exceptions (MCE)

The **mcelog** package logs and parses/translates Machine Check Exceptions (MCE) on hardware errors, including I/O, CPU, and memory errors. In addition, mcelog handles predictive bad page offlining and automatic core offlining when cache errors happen. Formerly this was managed by a cron job executed hourly. Now hardware errors are immediately processed by a mcelog daemon.



Note: Support for AMD Scalable MCA

openSUSE Leap 15.0 and higher support AMD's Scalable Machine Check Architecture (Scalable MCA). Scalable MCA improves hardware error reporting in AMD Zen processors. It information logged in MCA banks for improved error handling and better diagnosability.

mcelog captures MCA messages (**rasdaemon** and **dmesg** also capture MCA messages). See Section 3.1 “Machine Check Architecture”, of *Processor Programming Reference (PPR) for AMD Family 17h Model 01h, Revision B1 Processors* for detailed information (http://developer.amd.com/wordpress/media/2017/11/54945_PPR_Family_17h_Models_00h-0Fh.pdf).

mcelog is configured in `/etc/mcelog/mcelog.conf`. Configuration options are documented in **man mcelog**, and at <http://mcelog.org/>. The following example shows only changes to the default file:

```
daemon = yes
filter = yes
filter-memory-errors = yes
no-syslog = yes
logfile = /var/log/mcelog
run-credentials-user = root
run-credentials-group = nobody
client-group = root
socket-path = /var/run/mcelog-client
```

The mcelog service is not enabled by default. The service can either be enabled and started via the YaST system services editor, or via command line:

```
root # systemctl enable mcelog
root # systemctl start mcelog
```

2.7.5 x86_64: dmidecode: DMI Table Decoder

dmidecode shows the machine's DMI table containing information such as serial numbers and BIOS revisions of the hardware.

```
root # dmidecode
# dmidecode 2.12
SMBIOS 2.5 present.
27 structures occupying 1298 bytes.
Table at 0x000EB250.

Handle 0x0000, DMI type 4, 35 bytes
Processor Information
    Socket Designation: J1PR
    Type: Central Processor
    Family: Other
    Manufacturer: Intel(R) Corporation
    ID: E5 06 01 00 FF FB EB BF
    Version: Intel(R) Core(TM) i5 CPU          750 @ 2.67GHz
    Voltage: 1.1 V
    External Clock: 133 MHz
    Max Speed: 4000 MHz
    Current Speed: 2667 MHz
    Status: Populated, Enabled
```

```
Upgrade: Other
L1 Cache Handle: 0x0004
L2 Cache Handle: 0x0003
L3 Cache Handle: 0x0001
Serial Number: Not Specified
Asset Tag: Not Specified
Part Number: Not Specified
```

```
[..]
```

2.8 Files and File Systems

2.8.1 Determine the File Type: **file**

The command **file** determines the type of a file or a list of files by checking `/usr/share/misc/magic`.

```
tux > file /usr/bin/file
/usr/bin/file: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), \
for GNU/Linux 2.6.4, dynamically linked (uses shared libs), stripped
```

The parameter **-f** *LIST* specifies a file with a list of file names to examine. The **-z** allows **file** to look inside compressed files:

```
tux > file /usr/share/man/man1/file.1.gz
/usr/share/man/man1/file.1.gz: gzip compressed data, from Unix, max compression
tux > file -z /usr/share/man/man1/file.1.gz
/usr/share/man/man1/file.1.gz: troff or preprocessor input text \
(gzip compressed data, from Unix, max compression)
```

The parameter **-i** outputs a mime type string rather than the traditional description.

```
tux > file -i /usr/share/misc/magic
/usr/share/misc/magic: text/plain charset=utf-8
```

2.8.2 File Systems and Their Usage: **mount**, **df** and **du**

The command **mount** shows which file system (device and type) is mounted at which mount point:

```
root # mount
```

```

/dev/sda2 on / type ext4 (rw,acl,user_xattr)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
debugfs on /sys/kernel/debug type debugfs (rw)
devtmpfs on /dev type devtmpfs (rw,mode=0755)
tmpfs on /dev/shm type tmpfs (rw,mode=1777)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
/dev/sda3 on /home type ext3 (rw)
securityfs on /sys/kernel/security type securityfs (rw)
fusectl on /sys/fs/fuse/connections type fusectl (rw)
gvfs-fuse-daemon on /home/tux/.gvfs type fuse.gvfs-fuse-daemon \
(rw,nosuid,nodev,user=tux)

```

Obtain information about total usage of the file systems with the command **df**. The parameter **-h** (or **--human-readable**) transforms the output into a form understandable for common users.

```

tux > df -h

```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda2	20G	5,9G	13G	32%	/
devtmpfs	1,6G	236K	1,6G	1%	/dev
tmpfs	1,6G	668K	1,6G	1%	/dev/shm
/dev/sda3	208G	40G	159G	20%	/home

Display the total size of all the files in a given directory and its subdirectories with the command **du**. The parameter **-s** suppresses the output of detailed information and gives only a total for each argument. **-h** again transforms the output into a human-readable form:

```

tux > du -sh /opt
192M /opt

```

2.8.3 Additional Information about ELF Binaries

Read the content of binaries with the **readelf** utility. This even works with ELF files that were built for other hardware architectures:

```

tux > readelf --file-header /bin/ls
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00
  Class:                               ELF64
  Data:                                   2's complement, little endian
  Version:                               1 (current)
  OS/ABI:                                UNIX - System V
  ABI Version:                           0

```

```
Type: EXEC (Executable file)
Machine: Advanced Micro Devices X86-64
Version: 0x1
Entry point address: 0x402540
Start of program headers: 64 (bytes into file)
Start of section headers: 95720 (bytes into file)
Flags: 0x0
Size of this header: 64 (bytes)
Size of program headers: 56 (bytes)
Number of program headers: 9
Size of section headers: 64 (bytes)
Number of section headers: 32
Section header string table index: 31
```

2.8.4 File Properties: **stat**

The command `stat` displays file properties:

```
tux > stat /etc/profile
File: `/etc/profile'
Size: 9662          Blocks: 24          IO Block: 4096   regular file
Device: 802h/2050d Inode: 132349       Links: 1
Access: (0644/-rw-r--r--)  Uid: (  0/   root)   Gid: (  0/   root)
Access: 2009-03-20 07:51:17.000000000 +0100
Modify: 2009-01-08 19:21:14.000000000 +0100
Change: 2009-03-18 12:55:31.000000000 +0100
```

The parameter `--file-system` produces details of the properties of the file system in which the specified file is located:

```
tux > stat /etc/profile --file-system
File: "/etc/profile"
ID: d4fb76e70b4d1746 Namelen: 255      Type: ext2/ext3
Block size: 4096      Fundamental block size: 4096
Blocks: Total: 2581445   Free: 1717327   Available: 1586197
Inodes: Total: 655776   Free: 490312
```

2.9 User Information

2.9.1 User Accessing Files: **fuser**

It can be useful to determine what processes or users are currently accessing certain files. Suppose, for example, you want to unmount a file system mounted at `/mnt`. `umount` returns "device is busy." The command `fuser` can then be used to determine what processes are accessing the device:

```
tux > fuser -v /mnt/*
```

	USER	PID	ACCESS	COMMAND
/mnt/notes.txt	tux	26597	f....	less

Following termination of the `less` process, which was running on another terminal, the file system can successfully be unmounted. When used with `-k` option, `fuser` will terminate processes accessing the file as well.

2.9.2 Who Is Doing What: **w**

With the command `w`, find out who is logged in to the system and what each user is doing. For example:

```
tux > w
```

```
16:00:59 up 1 day, 2:41, 3 users, load average: 0.00, 0.01, 0.05
```

USER	TTY	FROM	LOGIN@	IDLE	JCPU	PCPU	WHAT
tux	:0	console	Wed13	?xdm?	8:15	0.03s	/usr/lib/gdm/gd
tux	console	:0	Wed13	26:41m	0.00s	0.03s	/usr/lib/gdm/gd
tux	pts/0	:0	Wed13	20:11	0.10s	2.89s	/usr/lib/gnome-

If any users of other systems have logged in remotely, the parameter `-f` shows the computers from which they have established the connection.

2.10 Time and Date

2.10.1 Time Measurement with `time`

Determine the time spent by commands with the `time` utility. This utility is available in two versions: as a Bash built-in and as a program (`/usr/bin/time`).

```
tux > time find . > /dev/null
```

```
real    0m4.051s ❶  
user    0m0.042s ❷  
sys     0m0.205s ❸
```

- ❶ The real time that elapsed from the command's start-up until it finished.
- ❷ CPU time of the user as reported by the `times` system call.
- ❸ CPU time of the system as reported by the `times` system call.

The output of `/usr/bin/time` is much more detailed. It is recommended to run it with the `-v` switch to produce human-readable output.

```
/usr/bin/time -v find . > /dev/null  
  Command being timed: "find ."  
  User time (seconds): 0.24  
  System time (seconds): 2.08  
  Percent of CPU this job got: 25%  
  Elapsed (wall clock) time (h:mm:ss or m:ss): 0:09.03  
  Average shared text size (kbytes): 0  
  Average unshared data size (kbytes): 0  
  Average stack size (kbytes): 0  
  Average total size (kbytes): 0  
  Maximum resident set size (kbytes): 2516  
  Average resident set size (kbytes): 0  
  Major (requiring I/O) page faults: 0  
  Minor (reclaiming a frame) page faults: 1564  
  Voluntary context switches: 36660  
  Involuntary context switches: 496  
  Swaps: 0  
  File system inputs: 0  
  File system outputs: 0  
  Socket messages sent: 0  
  Socket messages received: 0  
  Signals delivered: 0  
  Page size (bytes): 4096
```

2.11 Graph Your Data: RRDtool

There are a lot of data in the world around you, which can be easily measured in time. For example, changes in the temperature, or the number of data sent or received by your computer's network interface. RRDtool can help you store and visualize such data in detailed and customizable graphs.

RRDtool is available for most Unix platforms and Linux distributions. openSUSE® Leap ships RRDtool as well. Install it either with YaST or by entering

`zypper install rrdtool` in the command line as `root`.



Tip: Bindings

There are Perl, Python, Ruby, and PHP bindings available for RRDtool, so that you can write your own monitoring scripts in your preferred scripting language.

2.11.1 How RRDtool Works

RRDtool is an abbreviation of *Round Robin Database tool*. *Round Robin* is a method for manipulating with a constant amount of data. It uses the principle of a circular buffer, where there is no end nor beginning to the data row which is being read. RRDtool uses Round Robin Databases to store and read its data.

As mentioned above, RRDtool is designed to work with data that change in time. The ideal case is a sensor which repeatedly reads measured data (like temperature, speed etc.) in constant periods of time, and then exports them in a given format. Such data are perfectly ready for RRDtool, and it is easy to process them and create the desired output.

Sometimes it is not possible to obtain the data automatically and regularly. Their format needs to be pre-processed before it is supplied to RRDtool, and often you need to manipulate RRDtool even manually.

The following is a simple example of basic RRDtool usage. It illustrates all three important phases of the usual RRDtool workflow: *creating* a database, *updating* measured values, and *viewing* the output.

2.11.2 A Practical Example

Suppose we want to collect and view information about the memory usage in the Linux system as it changes in time. To make the example more vivid, we measure the currently free memory over a period of 40 seconds in 4-second intervals. Three applications that usually consume a lot of system memory are started and closed: the Firefox Web browser, the Evolution e-mail client, and the Eclipse development framework.

2.11.2.1 Collecting Data

RRDtool is very often used to measure and visualize network traffic. In such case, the Simple Network Management Protocol (SNMP) is used. This protocol can query network devices for relevant values of their internal counters. Exactly these values are to be stored with RRDtool. For more information on SNMP, see <http://www.net-snmp.org/>.

Our situation is different—we need to obtain the data manually. A helper script `free_mem.sh` repetitively reads the current state of free memory and writes it to the standard output.

```
tux > cat free_mem.sh
INTERVAL=4
for steps in {1..10}
do
    DATE=`date +%s`
    FREEMEM=`free -b | grep "Mem" | awk '{ print $4 }'`
    sleep $INTERVAL
    echo "rrdtool update free_mem.rrd $DATE:$FREEMEM"
done
```

- The time interval is set to 4 seconds, and is implemented with the `sleep` command.
- RRDtool accepts time information in a special format - so called *Unix time*. It is defined as the number of seconds since the midnight of January 1, 1970 (UTC). For example, 1272907114 represents 2010-05-03 17:18:34.
- The free memory information is reported in bytes with `free -b`. Prefer to supply basic units (bytes) instead of multiple units (like kilobytes).
- The line with the `echo ...` command contains the future name of the database file (`free_mem.rrd`), and together creates a command line for updating RRDtool values.

After running `free_mem.sh`, you see an output similar to this:

```
tux > sh free_mem.sh
```



```
rrdtool update free_mem.rrd 1272974835:1182994432
rrdtool update free_mem.rrd 1272974839:1162817536
rrdtool update free_mem.rrd 1272974843:1096269824
rrdtool update free_mem.rrd 1272974847:1034219520
rrdtool update free_mem.rrd 1272974851:909438976
rrdtool update free_mem.rrd 1272974855:832454656
rrdtool update free_mem.rrd 1272974859:829120512
rrdtool update free_mem.rrd 1272974863:1180377088
rrdtool update free_mem.rrd 1272974867:1179369472
rrdtool update free_mem.rrd 1272974871:1181806592
```

It is convenient to redirect the command's output to a file with

```
sh free_mem.sh > free_mem_updates.log
```

to simplify its future execution.

2.11.2.2 Creating the Database

Create the initial Robin Round database for our example with the following command:

```
tux > rrdtool create free_mem.rrd --start 1272974834 --step=4 \
DS:memory:GAUGE:600:U:U RRA:AVERAGE:0.5:1:24
```

POINTS TO NOTICE

- This command creates a file called `free_mem.rrd` for storing our measured values in a Round Robin type database.
- The `--start` option specifies the time (in Unix time) when the first value will be added to the database. In this example, it is one less than the first time value of the `free_mem.sh` output (1272974835).
- The `--step` specifies the time interval in seconds with which the measured data will be supplied to the database.
- The `DS:memory:GAUGE:600:U:U` part introduces a new data source for the database. It is called *memory*, its type is *gauge*, the maximum number between two updates is 600 seconds, and the *minimal* and *maximal* value in the measured range are unknown (U).
- `RRA:AVERAGE:0.5:1:24` creates Round Robin archive (RRA) whose stored data are processed with the *consolidation functions* (CF) that calculates the *average* of data points. 3 arguments of the consolidation function are appended to the end of the line.

If no error message is displayed, then `free_mem.rrd` database is created in the current directory:

```
tux > ls -l free_mem.rrd
-rw-r--r-- 1 tux users 776 May  5 12:50 free_mem.rrd
```

2.11.2.3 Updating Database Values

After the database is created, you need to fill it with the measured data. In [Section 2.11.2.1, “Collecting Data”](#), we already prepared the file `free_mem_updates.log` which consists of `rrdtool update` commands. These commands do the update of database values for us.

```
tux > sh free_mem_updates.log; ls -l free_mem.rrd
-rw-r--r-- 1 tux users 776 May  5 13:29 free_mem.rrd
```

As you can see, the size of `free_mem.rrd` remained the same even after updating its data.

2.11.2.4 Viewing Measured Values

We have already measured the values, created the database, and stored the measured value in it. Now we can play with the database, and retrieve or view its values.

To retrieve all the values from our database, enter the following on the command line:

```
tux > rrdtool fetch free_mem.rrd AVERAGE --start 1272974830 \
--end 1272974871
      memory
1272974832: nan
1272974836: 1.1729059840e+09
1272974840: 1.1461806080e+09
1272974844: 1.0807572480e+09
1272974848: 1.0030243840e+09
1272974852: 8.9019289600e+08
1272974856: 8.3162112000e+08
1272974860: 9.1693465600e+08
1272974864: 1.1801251840e+09
1272974868: 1.1799787520e+09
1272974872: nan
```

POINTS TO NOTICE

- `AVERAGE` will fetch average value points from the database, because only one data source is defined ([Section 2.11.2.2, “Creating the Database”](#)) with `AVERAGE` processing and no other function is available.

- The first line of the output prints the name of the data source as defined in [Section 2.11.2.2, “Creating the Database”](#).
- The left results column represents individual points in time, while the right one represents corresponding measured average values in scientific notation.
- The nan in the last line stands for “not a number”.

Now a graph representing the values stored in the database is drawn:

```
tux > rrdtool graph free_mem.png \
--start 1272974830 \
--end 1272974871 \
--step=4 \
DEF:free_memory=free_mem.rrd:memory:AVERAGE \
LINE2:free_memory#FF0000 \
--vertical-label "GB" \
--title "Free System Memory in Time" \
--zoom 1.5 \
--x-grid SECOND:1:SECOND:4:SECOND:10:0:%X
```

POINTS TO NOTICE

- free_mem.png is the file name of the graph to be created.
- --start and --end limit the time range within which the graph will be drawn.
- --step specifies the time resolution (in seconds) of the graph.
- The DEF:... part is a data definition called *free_memory*. Its data are read from the free_mem.rrd database and its data source called *memory*. The *average* value points are calculated, because no others were defined in [Section 2.11.2.2, “Creating the Database”](#).
- The LINE... part specifies properties of the line to be drawn into the graph. It is 2 pixels wide, its data come from the *free_memory* definition, and its color is red.
- --vertical-label sets the label to be printed along the y axis, and --title sets the main label for the whole graph.
- --zoom specifies the zoom factor for the graph. This value must be greater than zero.
- --x-grid specifies how to draw grid lines and their labels into the graph. Our example places them every second, while major grid lines are placed every 4 seconds. Labels are placed every 10 seconds under the major grid lines.

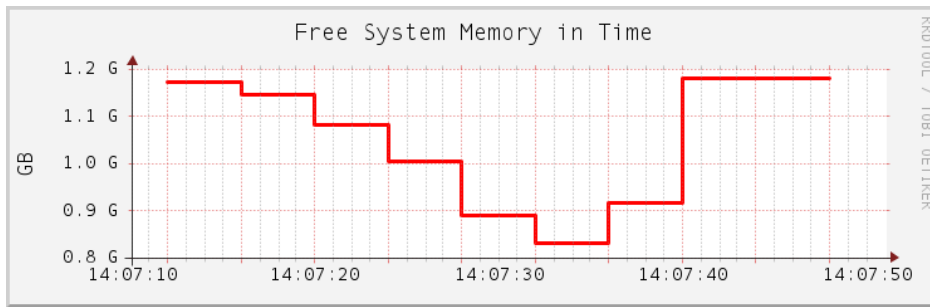


FIGURE 2.1: EXAMPLE GRAPH CREATED WITH RRDTOOL

2.11.3 For More Information

RRDtool is a very complex tool with a lot of sub-commands and command line options. Some are easy to understand, but to make it produce the results you want and fine-tune them according to your liking may require a lot of effort.

Apart from RRDtool's man page (`man 1 rrdtool`) which gives you only basic information, you should have a look at the [RRDtool home page](http://oss.oetiker.ch/rrdtool/) (<http://oss.oetiker.ch/rrdtool/>). There is a detailed [documentation](http://oss.oetiker.ch/rrdtool/doc/index.en.html) (<http://oss.oetiker.ch/rrdtool/doc/index.en.html>) of the `rrdtool` command and all its sub-commands. There are also several [tutorials](http://oss.oetiker.ch/rrdtool/tut/index.en.html) (<http://oss.oetiker.ch/rrdtool/tut/index.en.html>) to help you understand the common RRDtool workflow.

If you are interested in monitoring network traffic, have a look at [MRTG \(Multi Router Traffic Grapher\)](http://oss.oetiker.ch/mrtg/) (<http://oss.oetiker.ch/mrtg/>). MRTG can graph the activity of many network devices. It can use RRDtool.

3 Analyzing and Managing System Log Files

System log file analysis is one of the most important tasks when analyzing the system. In fact, looking at the system log files should be the first thing to do when maintaining or troubleshooting a system. openSUSE Leap automatically logs almost everything that happens on the system in detail. Since the move to `systemd`, kernel messages and messages of system services registered with `systemd` are logged in `systemd` journal (see *Book “Reference”, Chapter 11 “journalctl: Query the systemd Journal”*). Other log files (mainly those of system applications) are written in plain text and can be easily read using an editor or pager. It is also possible to parse them using scripts. This allows you to filter their content.

3.1 System Log Files in `/var/log/`

System log files are always located under the `/var/log` directory. The following list presents an overview of all system log files from openSUSE Leap present after a default installation. Depending on your installation scope, `/var/log` also contains log files from other services and applications not listed here. Some files and directories described below are “placeholders” and are only used, when the corresponding application is installed. Most log files are only visible for the user `root`.

apparmor/

AppArmor log files. For more information about AppArmor, see *Book “Security Guide”*.

audit/

Logs from the audit framework. See *Book “Security Guide”* for details.

ConsoleKit/

Logs of the `ConsoleKit` daemon (daemon for tracking what users are logged in and how they interact with the computer).

cups/

Access and error logs of the Common Unix Printing System (`cups`).

firewall

Firewall logs.

gdm/

Log files from the GNOME display manager.

krb5/

Log files from the Kerberos network authentication system.

lastlog

A database containing information on the last login of each user. Use the command **lastlog** to view. See **man 8 lastlog** for more information.

localmessages

Log messages of some boot scripts, for example the log of the DHCP client.

mail*

Mail server (postfix, sendmail) logs.

messages

This is the default place where all kernel and system log messages go and should be the first place (along with /var/log/warn) to look at in case of problems.

NetworkManager

NetworkManager log files.

news/

Log messages from a news server.

chrony/

Logs from the Network Time Protocol daemon (chrony).

pk_backend_zypp*

PackageKit (with libzypp back-end) log files.

samba/

Log files from Samba, the Windows SMB/CIFS file server.

warn

Log of all system warnings and errors. This should be the first place (along with the output of the systemd journal) to look in case of problems.

wtmp

Database of all login/logout activities, and remote connections. Use the command **last** to view. See **man 1 last** for more information.

Xorg.0.log

X.Org start-up log file. Refer to this in case you have problems starting X.Org. Copies from previous X.Org starts are numbered Xorg.?.log.

YaST2/

All YaST log files.

zypp/

libzypp log files. Refer to these files for the package installation history.

zypper.log

Logs from the command line installer zypper.

3.2 Viewing and Parsing Log Files

To view log files, you can use any text editor. There is also a simple YaST module for viewing the system log available in the YaST control center under *Miscellaneous* > *System Log*.

For viewing log files in a text console, use the commands less or more. Use head and tail to view the beginning or end of a log file. To view entries appended to a log file in real-time use tail -f. For information about how to use these tools, see their man pages.

To search for strings or regular expressions in log files use grep. awk is useful for parsing and rewriting log files.

3.3 Managing Log Files with **logrotate**

Log files under /var/log grow on a daily basis and quickly become very large. **logrotate** is a tool that helps you manage log files and their growth. It allows automatic rotation, removal, compression, and mailing of log files. Log files can be handled periodically (daily, weekly, or monthly) or when exceeding a particular size.

logrotate is usually run daily by systemd, and thus usually modifies log files only once a day. However, exceptions occur when a log file is modified because of its size, if **logrotate** is run multiple times a day, or if --force is enabled. Use /var/lib/misc/logrotate.status to find out when a particular file was last rotated.

The main configuration file of **logrotate** is /etc/logrotate.conf. System packages and programs that produce log files (for example, apache2) put their own configuration files in the /etc/logrotate.d/ directory. The content of /etc/logrotate.d/ is included via /etc/logrotate.conf.

EXAMPLE 3.1: **EXAMPLE FOR** /etc/logrotate.conf

```
# see "man logrotate" for details
```

```
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

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

# use date as a suffix of the rotated file
dateext

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

# comment these to switch compression to use gzip or another
# compression scheme
compresscmd /usr/bin/bzip2
uncompresscmd /usr/bin/bunzip2

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

! Important: Avoid Permission Conflicts

The `create` option pays heed to the modes and ownerships of files specified in `/etc/permissions*`. If you modify these settings, make sure no conflicts arise.

3.4 Monitoring Log Files with **logwatch**

logwatch is a customizable, pluggable log-monitoring script. It parses system logs, extracts the important information and presents them in a human readable manner. To use **logwatch**, install the `logwatch` package.

logwatch can either be used at the command line to generate on-the-fly reports, or via `cron` to regularly create custom reports. Reports can either be printed on the screen, saved to a file, or be mailed to a specified address. The latter is especially useful when automatically generating reports via `cron`.

On the command line, you can tell **logwatch** for which service and time span to generate a report and how much detail should be included:

```
# Detailed report on all kernel messages from yesterday
```



```
logwatch --service kernel --detail High --range Yesterday --print

# Low detail report on all sshd events recorded (incl. archived logs)
logwatch --service sshd --detail Low --range All --archives --print

# Mail a report on all smartd messages from May 5th to May 7th to root@localhost
logwatch --service smartd --range 'between 5/5/2005 and 5/7/2005' \
--mailto root@localhost --print
```

The `--range` option has got a complex syntax—see [logwatch --range help](#) for details. A list of all services that can be queried is available with the following command:

```
tux > ls /usr/share/logwatch/default.conf/services/ | sed 's/\.conf//g'
```

logwatch can be customized to great detail. However, the default configuration should usually be sufficient. The default configuration files are located under `/usr/share/logwatch/default.conf/`. Never change them because they would get overwritten again with the next update. Rather place custom configuration in `/etc/logwatch/conf/` (you may use the default configuration file as a template, though). A detailed HOWTO on customizing **logwatch** is available at [/usr/share/doc/packages/logwatch/HOWTO-Customize-LogWatch](#). The following configuration files exist:

logwatch.conf

The main configuration file. The default version is extensively commented. Each configuration option can be overwritten on the command line.

ignore.conf

Filter for all lines that should globally be ignored by **logwatch**.

services/*.conf

The service directory holds configuration files for each service you can generate a report for.

logfiles/*.conf

Specifications on which log files should be parsed for each service.

3.5 Using **logger** to Make System Log Entries

logger is a tool for making entries in the system log. It provides a shell command interface to the rsyslogd system log module. For example, the following line outputs its message in `/var/log/messages` or directly in the journal (if no logging facility is running):

```
tux > logger -t Test "This message comes from $USER"
```

Depending on the current user and host name, the log contains a line similar to this:

```
Sep 28 13:09:31 venus Test: This message comes from tux
```

III Kernel Monitoring

- 4 SystemTap—Filtering and Analyzing System Data 61
- 5 Kernel Probes 75
- 6 Hardware-Based Performance Monitoring with Perf 80
- 7 OProfile—System-Wide Profiler 85

4 SystemTap—Filtering and Analyzing System Data

SystemTap provides a command line interface and a scripting language to examine the activities of a running Linux system, particularly the kernel, in fine detail. SystemTap scripts are written in the SystemTap scripting language, are then compiled to C-code kernel modules and inserted into the kernel. The scripts can be designed to extract, filter and summarize data, thus allowing the diagnosis of complex performance problems or functional problems. SystemTap provides information similar to the output of tools like netstat, ps, top, and iostat. However, more filtering and analysis options can be used for the collected information.

4.1 Conceptual Overview

Each time you run a SystemTap script, a SystemTap session is started. Several passes are done on the script before it is allowed to run. Then, the script is compiled into a kernel module and loaded. If the script has been executed before and no system components have changed (for example, different compiler or kernel versions, library paths, or script contents), SystemTap does not compile the script again. Instead, it uses the *.c and *.ko data stored in the SystemTap cache (~/ .systemtap).

The module is unloaded when the tap has finished running. For an example, see the test run in *Section 4.2, “Installation and Setup”* and the respective explanation.

4.1.1 SystemTap Scripts

SystemTap usage is based on SystemTap scripts (*.stp). They tell SystemTap which type of information to collect, and what to do once that information is collected. The scripts are written in the SystemTap scripting language that is similar to AWK and C. For the language definition, see <http://sourceware.org/systemtap/langref/>. A lot of useful example scripts are available from <http://www.sourceware.org/systemtap/examples/>.

The essential idea behind a SystemTap script is to name events, and to give them handlers. When SystemTap runs the script, it monitors for certain events. When an event occurs, the Linux kernel runs the handler as a sub-routine, then resumes. Thus, events serve as the triggers for handlers to run. Handlers can record specified data and print it in a certain manner.

The SystemTap language only uses a few data types (integers, strings, and associative arrays of these), and full control structures (blocks, conditionals, loops, functions). It has a lightweight punctuation (semicolons are optional) and does not need detailed declarations (types are inferred and checked automatically).

For more information about SystemTap scripts and their syntax, refer to [Section 4.3, “Script Syntax”](#) and to the [stapprobes](#) and [stapfuncs](#) man pages, that are available with the [system-tap-docs](#) package.

4.1.2 Tapsets

Tapsets are a library of pre-written probes and functions that can be used in SystemTap scripts. When a user runs a SystemTap script, SystemTap checks the script's probe events and handlers against the tapset library. SystemTap then loads the corresponding probes and functions before translating the script to C. Like SystemTap scripts themselves, tapsets use the file name extension [*.stp](#).

However, unlike SystemTap scripts, tapsets are not meant for direct execution. They constitute the library from which other scripts can pull definitions. Thus, the tapset library is an abstraction layer designed to make it easier for users to define events and functions. Tapsets provide aliases for functions that users could want to specify as an event. Knowing the proper alias is often easier than remembering specific kernel functions that might vary between kernel versions.

4.1.3 Commands and Privileges

The main commands associated with SystemTap are [stap](#) and [staprun](#). To execute them, you either need [root](#) privileges or must be a member of the [stapdev](#) or [stapusr](#) group.

[stap](#)

SystemTap front-end. Runs a SystemTap script (either from file, or from standard input). It translates the script into C code, compiles it, and loads the resulting kernel module into a running Linux kernel. Then, the requested system trace or probe functions are performed.

[staprun](#)

SystemTap back-end. Loads and unloads kernel modules produced by the SystemTap front-end.

For a list of options for each command, use `--help`. For details, refer to the `stap` and the `staprun` man pages.

To avoid giving `root` access to users solely to enable them to work with SystemTap, use one of the following SystemTap groups. They are not available by default on openSUSE Leap, but you can create the groups and modify the access rights accordingly. Also adjust the permissions of the `staprun` command if the security implications are appropriate for your environment.

stapdev

Members of this group can run SystemTap scripts with `stap`, or run SystemTap instrumentation modules with `staprun`. As running `stap` involves compiling scripts into kernel modules and loading them into the kernel, members of this group still have effective `root` access.

stapusr

Members of this group are only allowed to run SystemTap instrumentation modules with `staprun`. In addition, they can only run those modules from `/lib/modules/KERNEL_VERSION/systemtap/`. This directory must be owned by `root` and must only be writable for the `root` user.

4.1.4 Important Files and Directories

The following list gives an overview of the SystemTap main files and directories.

/lib/modules/KERNEL_VERSION/systemtap/

Holds the SystemTap instrumentation modules.

/usr/share/systemtap/tapset/

Holds the standard library of tapsets.

/usr/share/doc/packages/systemtap/examples

Holds several example SystemTap scripts for various purposes. Only available if the `systemtap-docs` package is installed.

~/.systemtap/cache

Data directory for cached SystemTap files.

/tmp/stap*

Temporary directory for SystemTap files, including translated C code and kernel object.

4.2 Installation and Setup

As SystemTap needs information about the kernel, some additional kernel-related packages must be installed. For each kernel you want to probe with SystemTap, you need to install a set of the following packages. This set should exactly match the kernel version and flavor (indicated by * in the overview below).

Important: Repository for Packages with Debugging Information

If you subscribed your system for online updates, you can find “debuginfo” packages in the *-Debuginfo-Updates online installation repository relevant for openSUSE Leap 15.2. Use YaST to enable the repository.

For the classic SystemTap setup, install the following packages (using either YaST or zypper).

- systemtap
- systemtap-server
- systemtap-docs (optional)
- kernel-*-base
- kernel-*-debuginfo
- kernel-*-devel
- kernel-source-*
- gcc

To get access to the man pages and to a helpful collection of example SystemTap scripts for various purposes, additionally install the systemtap-docs package.

To check if all packages are correctly installed on the machine and if SystemTap is ready to use, execute the following command as root.

```
root # stap -v -e 'probe vfs.read {printf("read performed\n"); exit()}'
```

It probes the currently used kernel by running a script and returning an output. If the output is similar to the following, SystemTap is successfully deployed and ready to use:

```
Pass 1: parsed user script and 59 library script(s) in 80usr/0sys/214real ms.
```

```
Pass ②: analyzed script: 1 probe(s), 11 function(s), 2 embed(s), 1 global(s) in
140usr/20sys/412real ms.
Pass ③: translated to C into
"/tmp/stapDwEk76/stap_1856e21ea1c246da85ad8c66b4338349_4970.c" in 160usr/0sys/408real ms.
Pass ④: compiled C into "stap_1856e21ea1c246da85ad8c66b4338349_4970.ko" in
2030usr/360sys/10182real ms.
Pass ⑤: starting run.
read performed
Pass ⑤: run completed in 10usr/20sys/257real ms.
```

- ① Checks the script against the existing tapset library in `/usr/share/systemtap/tapset/` for any tapsets used. Tapsets are scripts that form a library of pre-written probes and functions that can be used in SystemTap scripts.
- ② Examines the script for its components.
- ③ Translates the script to C. Runs the system C compiler to create a kernel module from it. Both the resulting C code (`*.c`) and the kernel module (`*.ko`) are stored in the SystemTap cache, `~/.systemtap`.
- ④ Loads the module and enables all the probes (events and handlers) in the script by hooking into the kernel. The event being probed is a Virtual File System (VFS) read. As the event occurs on any processor, a valid handler is executed (prints the text `read performed`) and closed with no errors.
- ⑤ After the SystemTap session is terminated, the probes are disabled, and the kernel module is unloaded.

In case any error messages appear during the test, check the output for hints about any missing packages and make sure they are installed correctly. Rebooting and loading the appropriate kernel may also be needed.

4.3 Script Syntax

SystemTap scripts consist of the following two components:

SystemTap Events (Probe Points)

Name the kernel events at the associated handler should be executed. Examples for events are entering or exiting a certain function, a timer expiring, or starting or terminating a session.

SystemTap Handlers (Probe Body)

Series of script language statements that specify the work to be done whenever a certain event occurs. This normally includes extracting data from the event context, storing them into internal variables, or printing results.

An event and its corresponding handler is collectively called a probe. SystemTap events are also called probe points. A probe's handler is also called a probe body.

Comments can be inserted anywhere in the SystemTap script in various styles: using either #, /* */, or // as marker.

4.3.1 Probe Format

A SystemTap script can have multiple probes. They must be written in the following format:

```
probe EVENT {STATEMENTS}
```

Each probe has a corresponding statement block. This statement block must be enclosed in { } and contains the statements to be executed per event.

EXAMPLE 4.1: SIMPLE SYSTEMTAP SCRIPT

The following example shows a simple SystemTap script.

```
probe ① begin ②
{ ③
  printf ④ ("hello world\n") ⑤
  exit () ⑥
} ⑦
```

- ① Start of the probe.
- ② Event begin (the start of the SystemTap session).
- ③ Start of the handler definition, indicated by {.
- ④ First function defined in the handler: the printf function.
- ⑤ String to be printed by the printf function, followed by a line break (/n).
- ⑥ Second function defined in the handler: the exit() function. Note that the SystemTap script will continue to run until the exit() function executes. If you want to stop the execution of the script before, stop it manually by pressing Ctrl-C.
- ⑦ End of the handler definition, indicated by }.

The event `begin` ② (the start of the SystemTap session) triggers the handler enclosed in `{ }`. Here, that is the `printf` function ④. In this case, it prints `hello world` followed by a new line ⑤. Then, the script exits.

If your statement block holds several statements, SystemTap executes these statements in sequence—you do not need to insert special separators or terminators between multiple statements. A statement block can also be nested within another statement blocks. Generally, statement blocks in SystemTap scripts use the same syntax and semantics as in the C programming language.

4.3.2 SystemTap Events (Probe Points)

SystemTap supports several built-in events.

The general event syntax is a dotted-symbol sequence. This allows a breakdown of the event namespace into parts. Each component identifier may be parameterized by a string or number literal, with a syntax like a function call. A component may include a `*` character, to expand to other matching probe points. A probe point may be followed by a `?` character, to indicate that it is optional, and that no error should result if it fails to expand. Alternately, a probe point may be followed by a `!` character to indicate that it is both optional and sufficient.

SystemTap supports multiple events per probe—they need to be separated by a comma (`,`). If multiple events are specified in a single probe, SystemTap will execute the handler when any of the specified events occur.

In general, events can be classified into the following categories:

- Synchronous events: Occur when any process executes an instruction at a particular location in kernel code. This gives other events a reference point (instruction address) from which more contextual data may be available.

An example for a synchronous event is `vfs.FILE_OPERATION`: The entry to the `FILE_OPERATION` event for Virtual File System (VFS). For example, in [Section 4.2, “Installation and Setup”](#), `read` is the `FILE_OPERATION` event used for VFS.

- Asynchronous events: Not tied to a particular instruction or location in code. This family of probe points consists mainly of counters, timers, and similar constructs.

Examples for asynchronous events are: `begin` (start of a SystemTap session—when a SystemTap script is run), `end` (end of a SystemTap session), or timer events. Timer events specify a handler to be executed periodically, like `example timer.s(SECONDS)`, or `timer.m.s(MILLISECONDS)`.

When used together with other probes that collect information, timer events allow you to print periodic updates and see how that information changes over time.

EXAMPLE 4.2: PROBE WITH TIMER EVENT

For example, the following probe would print the text “hello world” every 4 seconds:

```
probe timer.s(4)
{
    printf("hello world\n")
}
```

For detailed information about supported events, refer to the [stapprobes](#) man page. The *See Also* section of the man page also contains links to other man pages that discuss supported events for specific subsystems and components.

4.3.3 SystemTap Handlers (Probe Body)

Each SystemTap event is accompanied by a corresponding handler defined for that event, consisting of a statement block.

4.3.3.1 Functions

If you need the same set of statements in multiple probes, you can place them in a function for easy reuse. Functions are defined by the keyword `function` followed by a name. They take any number of string or numeric arguments (by value) and may return a single string or number.

```
function FUNCTION_NAME(ARGUMENTS) {STATEMENTS}
probe EVENT {FUNCTION_NAME(ARGUMENTS)}
```

The statements in `FUNCTION_NAME` are executed when the probe for `EVENT` executes. The `ARGUMENTS` are optional values passed into the function.

Functions can be defined anywhere in the script. They may take any

One of the functions needed very often was already introduced in [Example 4.1, “Simple SystemTap Script”](#): the `printf` function for printing data in a formatted way. When using the `printf` function, you can specify how arguments should be printed by using a format string. The format string is included in quotation marks and can contain further format specifiers, introduced by a `%` character.

Which format strings to use depends on your list of arguments. Format strings can have multiple format specifiers—each matching a corresponding argument. Multiple arguments can be separated by a comma.

EXAMPLE 4.3: `printf` FUNCTION WITH FORMAT SPECIFIERS

```
printf (" ①%s ②(%d ③) open\n ④", execname(), pid())
```

- ① Start of the format string, indicated by `"`.
- ② String format specifier.
- ③ Integer format specifier.
- ④ End of the format string, indicated by `"`.

The example above prints the current executable name (`execname()`) as a string and the process ID (`pid()`) as an integer in brackets. Then, a space, the word `open` and a line break follow:

```
[...]
vmware-guestd(2206) open
hald(2360) open
[...]
```

Apart from the two functions `execname()` and `pid()` used in *Example 4.3, “printf Function with Format Specifiers”*, a variety of other functions can be used as `printf` arguments.

Among the most commonly used SystemTap functions are the following:

`tid()`

ID of the current thread.

`pid()`

Process ID of the current thread.

`uid()`

ID of the current user.

`cpu()`

Current CPU number.

`execname()`

Name of the current process.

`gettimeofday_s()`

Number of seconds since Unix epoch (January 1, 1970).

`ctime()`

Convert time into a string.

`pp()`

String describing the probe point currently being handled.

`thread_indent()`

Useful function for organizing print results. It (internally) stores an indentation counter for each thread (`tid()`). The function takes one argument, an indentation delta, indicating how many spaces to add or remove from the thread's indentation counter. It returns a string with some generic trace data along with an appropriate number of indentation spaces. The generic data returned includes a time stamp (number of microseconds since the initial indentation for the thread), a process name, and the thread ID itself. This allows you to identify what functions were called, who called them, and how long they took.

Call entries and exits often do not immediately precede each other (otherwise it would be easy to match them). In between a first call entry and its exit, usually other call entries and exits are made. The indentation counter helps you match an entry with its corresponding exit as it indents the next function call in case it is *not* the exit of the previous one.

For more information about supported SystemTap functions, refer to the [stapfuncs](#) man page.

4.3.3.2 Other Basic Constructs

Apart from functions, you can use other common constructs in SystemTap handlers, including variables, conditional statements (like `if/else`, `while` loops, `for` loops, arrays or command line arguments).

4.3.3.2.1 Variables

Variables may be defined anywhere in the script. To define one, simply choose a name and assign a value from a function or expression to it:

```
foo = gettimeofday( )
```

Then you can use the variable in an expression. From the type of values assigned to the variable, SystemTap automatically infers the type of each identifier (string or number). Any inconsistencies will be reported as errors. In the example above, `foo` would automatically be classified as a number and could be printed via `printf()` with the integer format specifier (`%d`).

However, by default, variables are local to the probe they are used in: They are initialized, used and disposed of at each handler evocation. To share variables between probes, declare them global anywhere in the script. To do so, use the `global` keyword outside of the probes:

EXAMPLE 4.4: USING GLOBAL VARIABLES

```
global count_jiffies, count_ms
probe timer.jiffies(100) { count_jiffies ++ }
probe timer.ms(100) { count_ms ++ }
probe timer.ms(12345)
{
    hz=(1000*count_jiffies) / count_ms
    printf ("jiffies:ms ratio %d:%d => CONFIG_HZ=%d\n",
        count_jiffies, count_ms, hz)
    exit ()
}
```

This example script computes the `CONFIG_HZ` setting of the kernel by using timers that count jiffies and milliseconds, then computing accordingly. (A jiffy is the duration of one tick of the system timer interrupt. It is not an absolute time interval unit, since its duration depends on the clock interrupt frequency of the particular hardware platform). With the `global` statement it is possible to use the variables `count_jiffies` and `count_ms` also in the probe `timer.ms(12345)`. With `++` the value of a variable is incremented by `1`.

4.3.3.2.2 Conditional Statements

There are several conditional statements that you can use in SystemTap scripts. The following are probably the most common:

If/Else Statements

They are expressed in the following format:

```
if (CONDITION) ① STATEMENT1 ②
else ③ STATEMENT2 ④
```

The `if` statement compares an integer-valued expression to zero. If the condition expression ① is non-zero, the first statement ② is executed. If the condition expression is zero, the second statement ④ is executed. The else clause (③ and ④) is optional. Both ② and ④ can also be statement blocks.

While Loops

They are expressed in the following format:

```
while (CONDITION) ① STATEMENT ②
```

As long as condition is non-zero, the statement ② is executed. ② can also be a statement block. It must change a value so condition will eventually be zero.

For Loops

They are a shortcut for while loops and are expressed in the following format:

```
for (INITIALIZATION ①; CONDITIONAL ②; INCREMENT ③) statement
```

The expression specified in ① is used to initialize a counter for the number of loop iterations and is executed before execution of the loop starts. The execution of the loop continues until the loop condition ② is false. (This expression is checked at the beginning of each loop iteration). The expression specified in ③ is used to increment the loop counter. It is executed at the end of each loop iteration.

Conditional Operators

The following operators can be used in conditional statements:

==: Is equal to

!=: Is not equal to

>=: Is greater than or equal to

<=: Is less than or equal to

4.4 Example Script

If you have installed the systemtap-docs package, you can find several useful SystemTap example scripts in /usr/share/doc/packages/systemtap/examples.

This section describes a rather simple example script in more detail: /usr/share/doc/packages/systemtap/examples/network/tcp_connections.stp.

EXAMPLE 4.5: MONITORING INCOMING TCP CONNECTIONS WITH tcp_connections.stp

```
#!/usr/bin/env stap  
  
probe begin {
```

```

printf("%6s %16s %6s %6s %16s\n",
       "UID", "CMD", "PID", "PORT", "IP_SOURCE")
}

probe kernel.function("tcp_accept").return?,
       kernel.function("inet_csk_accept").return? {
sock = $return
if (sock != 0)
    printf("%6d %16s %6d %6d %16s\n", uid(), execname(), pid(),
       inet_get_local_port(sock), inet_get_ip_source(sock))
}

```

This SystemTap script monitors the incoming TCP connections and helps to identify unauthorized or unwanted network access requests in real time. It shows the following information for each new incoming TCP connection accepted by the computer:

- User ID (UID)
- Command accepting the connection (CMD)
- Process ID of the command (PID)
- Port used by the connection (PORT)
- IP address from which the TCP connection originated (IP_SOURCE)

To run the script, execute

```
stap /usr/share/doc/packages/systemtap/examples/network/tcp_connections.stp
```

and follow the output on the screen. To manually stop the script, press `Ctrl-C`.

4.5 User Space Probing

For debugging user space applications (like DTrace can do), openSUSE Leap 15.2 supports user space probing with SystemTap: Custom probe points can be inserted in any user space application. Thus, SystemTap lets you use both kernel space and user space probes to debug the behavior of the whole system.

To get the required utrace infrastructure and the uprobes kernel module for user space probing, you need to install the `kernel-trace` package in addition to the packages listed in [Section 4.2, "Installation and Setup"](#).

utrace implements a framework for controlling user space tasks. It provides an interface that can be used by various tracing “engines”, implemented as loadable kernel modules. The engines register callback functions for specific events, then attach to whichever thread they want to trace. As the callbacks are made from “safe” places in the kernel, this allows for great leeway in the kinds of processing the functions can do. Various events can be watched via utrace, for example, system call entry and exit, fork(), signals being sent to the task, etc. More details about the utrace infrastructure are available at <http://sourceware.org/systemtap/wiki/utrace>.

SystemTap includes support for probing the entry into and return from a function in user space processes, probing predefined markers in user space code, and monitoring user-process events. To check if the currently running kernel provides the needed utrace support, use the following command:

```
tux > sudo grep CONFIG_UTRACE /boot/config-`uname -r`
```

For more details about user space probing, refer to https://sourceware.org/systemtap/SystemTap_Beginners_Guide/userspace-probing.html.

4.6 For More Information

This chapter only provides a short SystemTap overview. Refer to the following links for more information about SystemTap:

<http://sourceware.org/systemtap/>

SystemTap project home page.

<http://sourceware.org/systemtap/wiki/>

Huge collection of useful information about SystemTap, ranging from detailed user and developer documentation to reviews and comparisons with other tools, or Frequently Asked Questions and tips. Also contains collections of SystemTap scripts, examples and usage stories and lists recent talks and papers about SystemTap.

<http://sourceware.org/systemtap/documentation.html>

Features a *SystemTap Tutorial*, a *SystemTap Beginner's Guide*, a *Tapset Developer's Guide*, and a *SystemTap Language Reference* in PDF and HTML format. Also lists the relevant man pages.

You can also find the SystemTap language reference and SystemTap tutorial in your installed system under `/usr/share/doc/packages/systemtap`. Example SystemTap scripts are available from the `example` subdirectory.

5 Kernel Probes

Kernel probes are a set of tools to collect Linux kernel debugging and performance information. Developers and system administrators usually use them either to debug the kernel, or to find system performance bottlenecks. The reported data can then be used to tune the system for better performance.

You can insert these probes into any kernel routine, and specify a handler to be invoked after a particular break-point is hit. The main advantage of kernel probes is that you no longer need to rebuild the kernel and reboot the system after you make changes in a probe.

To use kernel probes, you typically need to write or obtain a specific kernel module. Such modules include both the *init* and the *exit* function. The *init* function (such as `register_kprobe()`) registers one or more probes, while the *exit* function unregisters them. The registration function defines *where* the probe will be inserted and *which handler* will be called after the probe is hit. To register or unregister a group of probes at one time, you can use relevant `register_<PROBE_TYPE>probes()` or `unregister_<PROBE_TYPE>probes()` functions.

Debugging and status messages are typically reported with the `printk` kernel routine. `printk` is a kernel space equivalent of a user space `printf` routine. For more information on `printk`, see [Logging kernel messages \(http://www.win.tue.nl/~aeb/linux/lk/lk-2.html#ss2.8\)](http://www.win.tue.nl/~aeb/linux/lk/lk-2.html#ss2.8). Normally, you can view these messages by inspecting the output of the `systemd` journal (see *Book "Reference", Chapter 11 "journalctl: Query the systemd Journal"*). For more information on log files, see *Chapter 3, Analyzing and Managing System Log Files*.

5.1 Supported Architectures

Kernel probes are *fully* implemented on the following architectures:

- x86
- AMD64/Intel 64
- Arm
- POWER

Kernel probes are *partially* implemented on the following architectures:

- IA64 (does not support probes on instruction `slot1`)
- sparc64 (return probes not yet implemented)

5.2 Types of Kernel Probes

There are three types of kernel probes: *Kprobes*, *Jprobes*, and *Kretprobes*. Kretprobes are sometimes called *return probes*. You can find source code examples of all three type of probes in the Linux kernel. See the directory [/usr/src/linux/samples/kprobes/](#) (package [kernel-source](#)).

5.2.1 Kprobes

Kprobes can be attached to any instruction in the Linux kernel. When Kprobes is registered, it inserts a break-point at the first byte of the probed instruction. When the processor hits this break-point, the processor registers are saved, and the processing passes to Kprobes. First, a *pre-handler* is executed, then the probed instruction is stepped, and, finally a *post-handler* is executed. The control is then passed to the instruction following the probe point.

5.2.2 Jprobes

Jprobes is implemented through the Kprobes mechanism. It is inserted on a function's entry point and allows direct access to the arguments of the function which is being probed. Its handler routine must have the same argument list and return value as the probed function. To end it, call the [jprobe_return\(\)](#) function.

When a jprobe is hit, the processor registers are saved, and the instruction pointer is directed to the jprobe handler routine. The control then passes to the handler with the same register contents as the function being probed. Finally, the handler calls the [jprobe_return\(\)](#) function, and switches the control back to the control function.

In general, you can insert multiple probes on one function. Jprobe is, however, limited to only one instance per function.

5.2.3 Return Probe

Return probes are also implemented through Kprobes. When the [register_kretprobe\(\)](#) function is called, a kprobe is attached to the entry of the probed function. After hitting the probe, the kernel probes mechanism saves the probed function return address and calls a user-defined return handler. The control is then passed back to the probed function.

Before you call `register_kretprobe()`, you need to set a `maxactive` argument, which specifies how many instances of the function can be probed at the same time. If set too low, you will miss a certain number of probes.

5.3 Kprobes API

The programming interface of Kprobes consists of functions which are used to register and unregister all used kernel probes, and associated probe handlers. For a more detailed description of these functions and their arguments, see the information sources in [Section 5.5, "For More Information"](#).

register_kprobe()

Inserts a break-point on a specified address. When the break-point is hit, the `pre_handler` and `post_handler` are called.

register_jprobe()

Inserts a break-point in the specified address. The address needs to be the address of the first instruction of the probed function. When the break-point is hit, the specified handler is run. The handler should have the same argument list and return type as the probed.

register_kretprobe()

Inserts a return probe for the specified function. When the probed function returns, a specified handler is run. This function returns 0 on success, or a negative error number on failure.

unregister_kprobe(), unregister_jprobe(), unregister_kretprobe()

Removes the specified probe. You can use it any time after the probe has been registered.

register_kprobes(), register_jprobes(), register_kretprobes()

Inserts each of the probes in the specified array.

unregister_kprobes(), unregister_jprobes(), unregister_kretprobes()

Removes each of the probes in the specified array.

disable_kprobe(), disable_jprobe(), disable_kretprobe()

Disables the specified probe temporarily.

enable_kprobe(), enable_jprobe(), enable_kretprobe()

Temporarily enables disabled probes.

5.4 debugfs Interface

In recent Linux kernels, the Kprobes instrumentation uses the kernel's `debugfs` interface. It can list all registered probes and globally switch all probes on or off.

5.4.1 Listing Registered Kernel Probes

The list of all currently registered probes is in the `/sys/kernel/debug/kprobes/list` file.

```
saturn.example.com:~ # cat /sys/kernel/debug/kprobes/list
c015d71a k  vfs_read+0x0  [DISABLED]
c011a316 j  do_fork+0x0
c03dedc5 r  tcp_v4_rcv+0x0
```

The first column lists the address in the kernel where the probe is inserted. The second column prints the type of the probe: `k` for kprobe, `j` for jprobe, and `r` for return probe. The third column specifies the symbol, offset and optional module name of the probe. The following optional columns include the status information of the probe. If the probe is inserted on a virtual address which is not valid anymore, it is marked with `[GONE]`. If the probe is temporarily disabled, it is marked with `[DISABLED]`.

5.4.2 How to Switch All Kernel Probes On or Off

The `/sys/kernel/debug/kprobes/enabled` file represents a switch with which you can globally and forcibly turn on or off all the registered kernel probes. To turn them off, simply enter

```
root # echo "0" > /sys/kernel/debug/kprobes/enabled
```

on the command line as `root`. To turn them on again, enter

```
root # echo "1" > /sys/kernel/debug/kprobes/enabled
```

Note that this way you do not change the status of the probes. If a probe is temporarily disabled, it will not be enabled automatically but will remain in the `[DISABLED]` state after entering the latter command.

5.5 For More Information

To learn more about kernel probes, look at the following sources of information:

- Thorough but more technically oriented information about kernel probes is in [/usr/src/linux/Documentation/kprobes.txt](#) (package `kernel-source`).
- Examples of all three types of probes (together with related `Makefile`) are in the [/usr/src/linux/samples/kprobes/](#) directory (package `kernel-source`).
- In-depth information about Linux kernel modules and `printk` kernel routine is in [The Linux Kernel Module Programming Guide \(http://tldp.org/LDP/lkmpg/2.6/html/lkmpg.html\)](http://tldp.org/LDP/lkmpg/2.6/html/lkmpg.html) ↗
- Practical but slightly outdated information about the use of kernel probes can be found in [Kernel debugging with Kprobes \(http://www.ibm.com/developerworks/library/l-kprobes.html\)](http://www.ibm.com/developerworks/library/l-kprobes.html) ↗

6 Hardware-Based Performance Monitoring with Perf

Perf is an interface to access the performance monitoring unit (PMU) of a processor and to record and display software events such as page faults. It supports system-wide, per-thread, and KVM virtualization guest monitoring.

You can store resulting information in a report. This report contains information about, for example, instruction pointers or what code a thread was executing.

Perf consists of two parts:

- Code integrated into the Linux kernel that is responsible for instructing the hardware.
- The `perf` user space utility that allows you to use the kernel code and helps you analyze gathered data.

6.1 Hardware-Based Monitoring

Performance monitoring means collecting information related to how an application or system performs. This information can be obtained either through software-based means or from the CPU or chipset. Perf integrates both of these methods.

Many modern processors contain a performance monitoring unit (PMU). The design and functionality of a PMU is CPU-specific. For example, the number of registers, counters and features supported will vary by CPU implementation.

Each PMU model consists of a set of registers: the performance monitor configuration (PMC) and the performance monitor data (PMD). Both can be read, but only PMCs are writable. These registers store configuration information and data.

6.2 Sampling and Counting

Perf supports several profiling modes:

- **Counting.** Count the number of occurrences of an event.
- **Event-Based Sampling.** A less exact way of counting: A sample is recorded whenever a certain threshold number of events has occurred.

- **Time-Based Sampling.** A less exact way of counting: A sample is recorded in a defined frequency.
- **Instruction-Based Sampling (AMD64 only).** The processor follows instructions appearing in a given interval and samples which events they produce. This allows following up on individual instructions and seeing which of them is critical to performance.

6.3 Installing Perf

The Perf kernel code is already included with the default kernel. To be able to use the user space utility, install the package `perf`.

6.4 Perf Subcommands

To gather the required information, the `perf` tool has several subcommands. This section gives an overview of the most often used commands.

To see help in the form of a man page for any of the subcommands, use either `perf help SUB-COMMAND` or `man perf- SUBCOMMAND`.

`perf stat`

Start a program and create a statistical overview that is displayed after the program quits. `perf stat` is used to count events.

`perf record`

Start a program and create a report with performance counter information. The report is stored as `perf.data` in the current directory. `perf record` is used to sample events.

`perf report`

Display a report that was previously created with `perf record`.

`perf annotate`

Display a report file and an annotated version of the executed code. If debug symbols are installed, you will also see the source code displayed.

`perf list`

List event types that Perf can report with the current kernel and with your CPU. You can filter event types by category—for example, to see hardware events only, use `perf list hw`.

The man page for `perf_event_open` has short descriptions for the most important events. For example, to find a description of the event `branch-misses`, search for `BRANCH_MISSES` (note the spelling differences):

```
tux > man perf_event_open | grep -A5 BRANCH_MISSES
```

Sometimes, events may be ambiguous. Note that the lowercase hardware event names are not the name of raw hardware events but instead the name of aliases created by Perf. These aliases map to differently named but similarly defined hardware events on each supported processor.

For example, the `cpu-cycles` event is mapped to the hardware event `UNHALTED_CORE_CYCLES` on Intel processors. On AMD processors, however, it is mapped to the hardware event `CPU_CLK_UNHALTED`.

Perf also allows measuring raw events specific to your hardware. To look up their descriptions, see the Architecture Software Developer's Manual of your CPU vendor. The relevant documents for AMD64/Intel 64 processors are linked to in [Section 6.7, "For More Information"](#).

`perf top`

Display system activity as it happens.

`perf trace`

This command behaves similarly to `strace`. With this subcommand, you can see which system calls are executed by a particular thread or process and which signals it receives.

6.5 Counting Particular Types of Event

To count the number of occurrences of an event, such as those displayed by `perf list`, use:

```
root # perf stat -e EVENT -a
```

To count multiple types of events at once, list them separated by commas. For example, to count `cpu-cycles` and `instructions`, use:

```
root # perf stat -e cpu-cycles,instructions -a
```

To stop the session, press `Ctrl-C`.

You can also count the number of occurrences of an event within a particular time:

```
root # perf stat -e EVENT -a -- sleep TIME
```

Replace `TIME` by a value in seconds.

6.6 Recording Events Specific to Particular Commands

There are various ways to sample events specific to a particular command:

- To create a report for a newly invoked command, use:

```
root # perf record COMMAND
```

Then, use the started process normally. When you quit the process, the Perf session will also stop.

- To create a report for the entire system while a newly invoked command is running, use:

```
root # perf record -a COMMAND
```

Then, use the started process normally. When you quit the process, the Perf session will also stop.

- To create a report for an already running process, use:

```
root # perf record -p PID
```

Replace PID with a process ID. To stop the session, press `Ctrl-C`.

Now you can view the gathered data (`perf.data`) using:

```
tux > perf report
```

This will open a pseudo-graphical interface. To receive help, press `H`. To quit, press `Q`.

If you prefer a graphical interface, try the GTK+ interface of Perf:

```
tux > perf report --gtk
```

However, note that the GTK+ interface is very limited in functionality.

6.7 For More Information

This chapter only provides a short overview. Refer to the following links for more information:

https://perf.wiki.kernel.org/index.php/Main_Page ↗

The project home page. It also features a tutorial on using `perf`.

<http://www.brendangregg.com/perf.html> ↗

Unofficial page with many one-line examples of how to use `perf`.

http://web.eece.maine.edu/~vweaver/projects/perf_events/ ↗

Unofficial page with several resources, mostly relating to the Linux kernel code of Perf and its API. This page includes, for example, a CPU compatibility table and a programming guide.

<https://www-ssl.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-3b-part-2-manual.pdf> ↗

The *Intel Architectures Software Developer's Manual, Volume 3B*.

<https://support.amd.com/TechDocs/24593.pdf> ↗

The *AMD Architecture Programmer's Manual, Volume 2*.

Chapter 7, OProfile—System-Wide Profiler

Consult this chapter for other performance optimizations.

7 OProfile—System-Wide Profiler

OProfile is a profiler for dynamic program analysis. It investigates the behavior of a running program and gathers information. This information can be viewed and gives hints for further optimization.

It is not necessary to recompile or use wrapper libraries to use OProfile. Not even a kernel patch is needed. Usually, when profiling an application, a small overhead is expected, depending on the workload and sampling frequency.

7.1 Conceptual Overview

OProfile consists of a kernel driver and a daemon for collecting data. It uses the hardware performance counters provided on many processors. OProfile is capable of profiling all code including the kernel, kernel modules, kernel interrupt handlers, system shared libraries, and other applications.

Modern processors support profiling through the hardware by performance counters. Depending on the processor, there can be many counters and each of these can be programmed with an event to count. Each counter has a value which determines how often a sample is taken. The lower the value, the more often it is used.

During the post-processing step, all information is collected and instruction addresses are mapped to a function name.

7.2 Installation and Requirements

To use OProfile, install the `oprofile` package.

It is useful to install the `*-debuginfo` package for the respective application you want to profile.

If you want to profile the kernel, you need the `debuginfo` package as well.

7.3 Available OProfile Utilities

OProfile contains several utilities to handle the profiling process and its profiled data. The following list is a short summary of programs used in this chapter:

opannotate

Outputs annotated source or assembly listings mixed with profile information. An annotated report can be used in combination with addr2line to identify the source file and line where hotspots potentially exist. See man addr2line for more information.

operf

Profiler tool. After profiling stops, the data that is by default stored in CUR_DIR/oprofile_data/samples/current can be processed by opreport, for example.

ophelp

Lists available events with short descriptions.

opimport

Converts sample database files from a foreign binary format to the native format.

opreport

Generates reports from profiled data.

7.4 Using OProfile

With OProfile, you can profile both the kernel and applications. When profiling the kernel, tell OProfile where to find the vmlinux* file. Use the --vmlinux option and point it to vmlinux* (usually in /boot). If you need to profile kernel modules, OProfile does this by default. However, make sure you read <http://oprofile.sourceforge.net/doc/kernel-profiling.html>.

Applications usually do not need to profile the kernel, therefore you should use the --no-vmlinux option to reduce the amount of information.

7.4.1 Creating a Report

Starting the daemon, collecting data, stopping the daemon, and creating a report for the application COMMAND.

1. Open a shell and log in as root.

2. Decide if you want to profile with or without the Linux kernel:

- a. **Profile With the Linux Kernel.** Execute the following commands, because **operf** can only work with uncompressed images:

```
tux > cp /boot/vmlinux-`uname -r`.gz /tmp
tux > gunzip /tmp/vmlinux*.gz
tux > operf --vmlinux=/tmp/vmlinux* COMMAND
```

- b. **Profile Without the Linux Kernel.** Use the following command:

```
root # operf --no-vmlinux COMMAND
```

To see which functions call other functions in the output, additionally use the `--callgraph` option and set a maximum `DEPTH`:

```
root # operf --no-vmlinux --callgraph
DEPTH COMMAND
```

3. **operf** writes its data to `CUR_DIR/oprofile_data/samples/current`. After the **operf** command is finished (or is aborted by `Ctrl-C`), the data can be analyzed with **oreport**:

```
root # oreport
Overflow stats not available
CPU: CPU with timer interrupt, speed 0 MHz (estimated)
Profiling through timer interrupt
      TIMER:0|
samples|    %|
-----|
      84877 98.3226 no-vmlinux
...

```

7.4.2 Getting Event Configurations

The general procedure for event configuration is as follows:

1. Use first the events `CPU-CLK_UNHALTED` and `INST_RETIRED` to find optimization opportunities.
2. Use specific events to find bottlenecks. To list them, use the command **perf list**.

If you need to profile certain events, first check the available events supported by your processor with the `ophelp` command (example output generated from Intel Core i5 CPU):

```
root # ophelp
oprofile: available events for CPU type "Intel Architectural Perfmon"

See Intel 64 and IA-32 Architectures Software Developer's Manual
Volume 3B (Document 253669) Chapter 18 for architectural perfmon events
This is a limited set of fallback events because oprofile does not know your CPU
CPU_CLK_UNHALTED: (counter: all)
    Clock cycles when not halted (min count: 6000)
INST_RETIRED: (counter: all)
    number of instructions retired (min count: 6000)
LLC_MISSES: (counter: all)
    Last level cache demand requests from this core that missed the LLC (min count:
6000)
    Unit masks (default 0x41)
    -----
    0x41: No unit mask
LLC_REFS: (counter: all)
    Last level cache demand requests from this core (min count: 6000)
    Unit masks (default 0x4f)
    -----
    0x4f: No unit mask
BR_MISS_PRED_RETIRED: (counter: all)
    number of mispredicted branches retired (precise) (min count: 500)
```

Specify the performance counter events with the option `--event`. Multiple options are possible. This option needs an event name (from `ophelp`) and a sample rate, for example:

```
root # operf --events CPU_CLK_UNHALTED:100000
```



Warning: Setting Sampling Rates with CPU_CLK_UNHALTED

Setting low sampling rates can seriously impair the system performance while high sample rates can disrupt the system to such a high degree that the data is useless. It is recommended to tune the performance metric for being monitored with and without OProfile and to experimentally determine the minimum sample rate that disrupts the performance the least.

7.5 Generating Reports

Before generating a report, make sure the **oprof** has stopped. Unless you have provided an output directory with `--session-dir`, **oprof** has written its data to `CUR_DIR/oprofile_data/samples/current`, and the reporting tools **opreport** and **opannotate** will look there by default.

Calling **opreport** without any options gives a complete summary. With an executable as an argument, retrieve profile data only from this executable. If you analyze applications written in C++, use the `--demangle smart` option.

The **opannotate** generates output with annotations from source code. Run it with the following options:

```
root # opannotate --source \  
--base-dirs=BASEDIR \  
--search-dirs=SEARCHDIR \  
--output-dir=annotated/ \  
/lib/libfoo.so
```

The option `--base-dir` contains a comma separated list of paths which is stripped from debug source files. These paths were searched prior to looking in `--search-dirs`. The `--search-dirs` option is also a comma separated list of directories to search for source files.



Note: Inaccuracies in Annotated Source

Because of compiler optimization, code can disappear and appear in a different place. Use the information in <http://oprofile.sourceforge.net/doc/debug-info.html> to fully understand its implications.

7.6 For More Information

This chapter only provides a short overview. Refer to the following links for more information:

<http://oprofile.sourceforge.net>

The project home page.

Manpages

Details descriptions about the options of the different tools.

</usr/share/doc/packages/oprofile/oprofile.html>

Contains the OProfile manual.

<http://developer.intel.com/> ↗

Architecture reference for Intel processors.

IV Resource Management

- 8 General System Resource Management **92**
- 9 Kernel Control Groups **97**
- 10 Automatic Non-Uniform Memory Access (NUMA) Balancing **104**
- 11 Power Management **109**

8 General System Resource Management

Tuning the system is not only about optimizing the kernel or getting the most out of your application, it begins with setting up a lean and fast system. The way you set up your partitions and file systems can influence the server's speed. The number of active services and the way routine tasks are scheduled also affects performance.

8.1 Planning the Installation

A carefully planned installation ensures that the system is set up exactly as you need it for the given purpose. It also saves considerable time when fine tuning the system. All changes suggested in this section can be made in the *Installation Settings* step during the installation. See *Book "Start-Up", Chapter 3 "Installation Steps", Section 3.11 "Installation Settings"* for details.

8.1.1 Partitioning

Depending on the server's range of applications and the hardware layout, the partitioning scheme can influence the machine's performance (although to a lesser extent only). It is beyond the scope of this manual to suggest different partitioning schemes for particular workloads. However, the following rules will positively affect performance. They do not apply when using an external storage system.

- Make sure there always is some free space available on the disk, since a full disk delivers inferior performance
- Disperse simultaneous read and write access onto different disks by, for example:
 - using separate disks for the operating system, data, and log files
 - placing a mail server's spool directory on a separate disk
 - distributing the user directories of a home server between different disks

8.1.2 Installation Scope

The installation scope has no direct influence on the machine's performance, but a carefully chosen scope of packages has advantages. It is recommended to install the minimum of packages needed to run the server. A system with a minimum set of packages is easier to maintain and has fewer potential security issues. Furthermore, a tailor made installation scope also ensures that no unnecessary services are started by default.

openSUSE Leap lets you customize the installation scope on the Installation Summary screen. By default, you can select or remove preconfigured patterns for specific tasks, but it is also possible to start the YaST Software Manager for a fine-grained package-based selection.

One or more of the following default patterns may not be needed in all cases:

GNOME Desktop Environment

Servers rarely need a full desktop environment. In case a graphical environment is needed, a more economical solution such as IceWM can be sufficient.

X Window System

When solely administrating the server and its applications via command line, consider not installing this pattern. However, keep in mind that it is needed to run GUI applications from a remote machine. If your application is managed by a GUI or if you prefer the GUI version of YaST, keep this pattern.

Print Server

This pattern is only needed if you want to print from the machine.

8.1.3 Default Target

A running X Window System consumes many resources and is rarely needed on a server. It is strongly recommended to start the system in target `multi-user.target`. You will still be able to remotely start graphical applications.

8.2 Disabling Unnecessary Services

The default installation starts several services (the number varies with the installation scope). Since each service consumes resources, it is recommended to disable the ones not needed. Run `YaST > System > Services Manager` to start the services management module.

If you are using the graphical version of YaST, you can click the column headlines to sort the list of services. Use this to get an overview of which services are currently running. Use the *Start/Stop* button to disable the service for the running session. To permanently disable it, use the *Enable/Disable* button.

The following list shows services that are started by default after the installation of openSUSE Leap. Check which of the components you need, and disable the others:

alsasound

Loads the Advanced Linux Sound System.

auditd

A daemon for the Audit system (see *Book "Security Guide"* for details). Disable this if you do not use Audit.

bluez-coldplug

Handles cold plugging of Bluetooth dongles.

cups

A printer daemon.

java.binfmt_misc

Enables the execution of **.class* or **.jar* Java programs.

nfs

Services needed to mount NFS.

smbfs

Services needed to mount SMB/CIFS file systems from a Windows* server.

splash / splash_early

Shows the splash screen on start-up.

8.3 File Systems and Disk Access

Hard disks are the slowest components in a computer system and therefore often the cause for a bottleneck. Using the file system that best suits your workload helps to improve performance. Using special mount options or prioritizing a process's I/O priority are further means to speed up the system.

8.3.1 File Systems

openSUSE Leap ships with several file systems, including Btrfs, Ext4, Ext3, Ext2, ReiserFS, and XFS. Each file system has its own advantages and disadvantages.

8.3.1.1 NFS

NFS (Version 3) tuning is covered in detail in the NFS Howto at <http://nfs.sourceforge.net/nfs-howto/>. The first thing to experiment with when mounting NFS shares is increasing the read write blocksize to 32768 by using the mount options wsize and rsize.

8.3.2 Time Stamp Update Policy

Each file and directory in a file system has three time stamps associated with it: a time when the file was last read called *access time*, a time when the file data was last modified called *modification time*, and a time when the file metadata was last modified called *change time*. Keeping access time always up to date has significant performance overhead since every read-only access will incur a write operation. Thus by default every file system updates access time only if current file access time is older than a day or if it is older than file modification or change time. This feature is called *relative access time* and the corresponding mount option is relatime. Updates of access time can be completely disabled using the noatime mount option, however you need to verify your applications do not use it. This can be true for file and Web servers or for network storage. If the default relative access time update policy is not suitable for your applications, use the strictatime mount option.

Some file systems (for example Ext4) also support lazy time stamp updates. When this feature is enabled using the lazytime mount option, updates of all time stamps happen in memory but they are not written to disk. That happens only in response to fsync or sync system calls, when the file information is written due to another reason such as file size update, when time stamps are older than 24 hours, or when cached file information needs to be evicted from memory.

To update mount options used for a file system, either edit /etc/fstab directly, or use the *Fstab Options* dialog when editing or adding a partition with the YaST Partitioner.

8.3.3 Prioritizing Disk Access with **ionice**

The **ionice** command lets you prioritize disk access for single processes. This enables you to give less I/O priority to background processes with heavy disk access that are not time-critical, such as backup jobs. **ionice** also lets you raise the I/O priority for a specific process to make sure this process always has immediate access to the disk. The caveat of this feature is that standard writes are cached in the page cache and are written back to persistent storage only later by an independent kernel process. Thus the I/O priority setting generally does not apply for these writes. Also be aware that I/O class and priority setting are obeyed only by *BFQ* I/O scheduler for blk-mq I/O path (refer to [Section 12.2, “Available I/O Elevators with blk-mq I/O path”](#)). You can set the following three scheduling classes:

Idle

A process from the idle scheduling class is only granted disk access when no other process has asked for disk I/O.

Best effort

The default scheduling class used for any process that has not asked for a specific I/O priority. Priority within this class can be adjusted to a level from 0 to 7 (with 0 being the highest priority). Programs running at the same best-effort priority are served in a round-robin fashion. Some kernel versions treat priority within the best-effort class differently—for details, refer to the [ionice\(1\)](#) man page.

Real-time

Processes in this class are always granted disk access first. Fine-tune the priority level from 0 to 7 (with 0 being the highest priority). Use with care, since it can starve other processes.

For more details and the exact command syntax refer to the [ionice\(1\)](#) man page. If you need more reliable control over bandwidth available to each application, use Kernel Control Groups as described in [Chapter 9, Kernel Control Groups](#).

9 Kernel Control Groups

Kernel Control Groups (“cgroups”) are a kernel feature that allows assigning and limiting hardware and system resources for processes. Processes can also be organized in a hierarchical tree structure.

9.1 Overview

Every process is assigned exactly one administrative cgroup. cgroups are ordered in a hierarchical tree structure. You can set resource limitations, such as CPU, memory, disk I/O, or network bandwidth usage, for single processes or for whole branches of the hierarchy tree.

On openSUSE Leap, `systemd` uses cgroups to organize all processes in groups, which `systemd` calls slices. `systemd` also provides an interface for setting cgroup properties.

The command `systemd-cgls` displays the hierarchy tree.

This chapter is an overview. For more details, refer to the listed references.

9.2 Setting Resource Limits



Note: Implicit Resource Consumption

Be aware that resource consumption implicitly depends on the environment where your workload executes (e.g. size of data structures in libraries/kernel, forking behavior of utilities, computational efficiency), hence it is recommended to (re)calibrate your limits should the environment change.

Limitations to `cgroups` can be set with the `systemctl set-property` command. The syntax is:

```
root # systemctl set-property [--runtime] NAME PROPERTY1=VALUE [PROPERTY2=VALUE]
```

Optionally, use the `--runtime` option. With this option, set limits do not persist after the next reboot.

Replace `NAME` with a `systemd` service slice, scope, socket, mount, or swap name. Replace properties with one or more of the following:

```
CPUAccounting= [yes|no]
```


Turns on CPU usage accounting. This property takes yes and no as arguments.

Example:

```
root # systemctl set-property user.slice CPUAccounting=yes
```

CPUQuota= PERCENTAGE

Assigns a CPU time to processes. The value is a percentage followed by a % as suffix. This implies CPUAccounting=yes.

Example:

```
root # systemctl set-property user.slice CPUQuota=50%
```

MemoryAccounting= [yes|no]

Turns on memory usage accounting. This property takes yes and no as arguments.

Example:

```
root # systemctl set-property user.slice MemoryAccounting=yes
```

MemoryLow= BYTES

Unused memory from processes below this limit will not be reclaimed for other use. Use suffixes K, M, G or T for BYTES. This implies MemoryAccounting=yes.

Example:

```
root # systemctl set-property nginx.service MemoryLow=512M
```



Note: Unified Control Group Hierarchy

This setting is available only if the unified control group hierarchy is used, and disables MemoryLimit=. To enable the unified control group hierarchy, append systemd.unified_cgroup_hierarchy=1 as a kernel command line parameter to the GRUB 2 boot loader. Refer to *Book "Reference", Chapter 12 "The Boot Loader GRUB 2"* for more details about configuring GRUB 2.

MemoryHigh= BYTES

If more memory above this limit is used, memory is aggressively taken away from the processes. Use suffixes K, M, G or T for BYTES. This implies MemoryAccounting=yes. For example:

```
root # systemctl set-property nginx.service MemoryHigh=2G
```



Note: Unified Control Group Hierarchy

This setting is available only if the unified control group hierarchy is used, and disables `MemoryLimit=`. To enable the unified control group hierarchy, append `systemd.unified_cgroup_hierarchy=1` as a kernel command line parameter to the GRUB 2 boot loader. For more details about configuring GRUB 2, see *Book "Reference", Chapter 12 "The Boot Loader GRUB 2"*.

`MemoryMax= BYTES`

Sets a maximum limit for used memory. Processes will be killed if they use more memory than allowed. Use suffixes K, M, G or T for `BYTES`. This implies `MemoryAccounting=yes`.
Example:

```
root # systemctl set-property nginx.service MemoryMax=4G
```

`DeviceAllow=`

Allows read (`r`), write (`w`) and mknod (`m`) access. The command takes a device node specifier and a list of `r`, `w` or `m`, separated by a white space.

Example:

```
root # systemctl set-property system.slice DeviceAllow="/dev/sdb1 r"
```

`DevicePolicy= [auto|closed|strict]`

When set to `strict`, only access to devices that are listed in `DeviceAllow` is allowed. `closed` additionally allows access to standard pseudo devices including `/dev/null`, `/dev/zero`, `/dev/full`, `/dev/random`, and `/dev/urandom`. `auto` allows access to all devices if no specific rule is defined in `DeviceAllow`. `auto` is the default setting.

For more details and a complete list of properties, see `man systemd.resource-control`.

9.3 Preventing Fork Bombs with TasksMax

`systemd` 228 shipped with a `DefaultTasksMax` limit of 512. This limited the number of processes any system unit can create at one time to 512. Previous versions had no default limit. The goal was to improve security by preventing runaway processes from creating excessive forks, or spawning enough threads to exhaust system resources.

However, it soon became apparent that there is not a single default that applies to all use cases. 512 is not low enough to prevent a runaway process from crashing a system, especially when other resources such as CPU and RAM are not restricted, and not high enough for processes that create a lot of threads, such as databases. In `systemd` 234 the default was changed to 15%, which is 4915 tasks (15% of the kernel limit of 32768; see `cat /proc/sys/kernel/pid_max`). This default is compiled, and can be changed in configuration files. The compiled defaults are documented in `/etc/systemd/system.conf`. You can edit this file to override the defaults, though there are other methods we will show in the following sections.

9.3.1 Finding the Current Default TasksMax Values

openSUSE Leap ships with two custom configurations that override the upstream defaults for system units and for user slices, and sets them both to `infinity`. `/usr/lib/systemd/system.conf.d/20-suse-defaults.conf` contains these lines:

```
[Manager]
DefaultTasksMax=infinity
```

`/usr/lib/systemd/system/user-.slice.d/20-suse-defaults.conf` contains these lines:

```
[Slice]
TasksMax=infinity
```

`infinity` means having no limit. It is not a requirement to change the default, but setting some limits may help to prevent system crashes from runaway processes.

9.3.2 Overriding the DefaultTasksMax Value

Change the global `DefaultTasksMax` value by creating a new override file, `/etc/systemd/system.conf.d/10-system-tasksmax.conf`, and write the following lines to set new default limit of 256 tasks per system unit:

```
[Manager]
DefaultTasksMax=256
```

Load the new setting, then verify that it changed:

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

```
tux > systemctl show --property DefaultTasksMax
DefaultTasksMax=256
```

Adjust this default value to suit your needs. You can set higher limits on individual services as needed. This example is for MariaDB. First check the current active value:

```
tux > systemctl status mariadb.service
● mariadb.service - MariaDB database server
  Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled; vendor preset>
  Active: active (running) since Tue 2020-05-26 14:15:03 PDT; 27min ago
    Docs: man:mysql(8)
          https://mariadb.com/kb/en/library/systemd/
 Main PID: 11845 (mysqld)
  Status: "Taking your SQL requests now..."
   Tasks: 30 (limit: 256)
  CGroup: /system.slice/mariadb.service
          └─11845 /usr/sbin/mysqld --defaults-file=/etc/my.cnf --user=mysql
```

The Tasks line shows that MariaDB currently has 30 tasks running, and has an upper limit of the default 256, which is inadequate for a database. The following example demonstrates how to raise MariaDB's limit to 8192. Create a new override file with **systemctl edit**, and enter the new value:

```
tux > sudo systemctl edit mariadb.service

[Service]
TasksMax=8192

tux > systemctl status mariadb.service
● mariadb.service - MariaDB database server
  Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled; vendor preset:
  Disab>
 Drop-In: /etc/systemd/system/mariadb.service.d
          └─override.conf
  Active: active (running) since Tue 2020-06-02 17:57:48 PDT; 7min ago
    Docs: man:mysql(8)
          https://mariadb.com/kb/en/library/systemd/
 Process: 3446 ExecStartPre=/usr/lib/mysql/mysql-systemd-helper upgrade (code=exited,
  sta>
 Process: 3440 ExecStartPre=/usr/lib/mysql/mysql-systemd-helper install (code=exited,
  sta>
 Main PID: 3452 (mysqld)
  Status: "Taking your SQL requests now..."
   Tasks: 30 (limit: 8192)
  CGroup: /system.slice/mariadb.service
          └─3452 /usr/sbin/mysqld --defaults-file=/etc/my.cnf --user=mysql
```

`systemctl edit` creates an override file, `/etc/systemd/system/mariadb.service.d/override.conf`, that contains only the changes you want to apply to the existing unit file. The value does not have to be 8192, but should be whatever limit is appropriate for your workloads.

9.3.3 Default TasksMax Limit on Users

The default limit on users should be fairly high, because user sessions need more resources. Set your own default for users by creating a new file, for example `/etc/systemd/system/user-.slice.d/user-taskmask.conf`. The following example sets a default of 16284:

```
[Slice]
TasksMax=16284
```

Then reload `systemd` to load the new value, and verify the change:

```
tux > sudo systemctl daemon-reload

tux > systemctl show --property TasksMax user-.slice
TasksMax=16284



tux > systemctl show --property TasksMax user-1000.slice
TasksMax=16284
```

How do you know what values to use? This varies according to your workloads, system resources, and other resource configurations. When your `TasksMax` value is too low, you will see error messages such as *Failed to fork (Resources temporarily unavailable)*, *Can't create thread to handle new connection*, and *Error: Function call 'fork' failed with error code 11, 'Resource temporarily unavailable'*.

For more information on configuring system resources in `systemd`, see [systemd.resource-control \(5\)](#).

9.4 For More Information

- Kernel documentation (package `kernel-source`): files in `/usr/src/linux/Documentation/cgroups`.
- <http://lwn.net/Articles/604609/> —Brown, Neil: Control Groups Series (2014, 7 parts).

- <http://lwn.net/Articles/243795/> —Corbet, Jonathan: Controlling memory use in containers (2007).
- <http://lwn.net/Articles/236038/> —Corbet, Jonathan: Process containers (2007).

10 Automatic Non-Uniform Memory Access (NUMA) Balancing

There are physical limitations to hardware that are encountered when many CPUs and lots of memory are required. In this chapter, the important limitation is that there is limited communication bandwidth between the CPUs and the memory. One architecture modification that was introduced to address this is Non-Uniform Memory Access (NUMA).

In this configuration, there are multiple nodes. Each of the nodes contains a subset of all CPUs and memory. The access speed to main memory is determined by the location of the memory relative to the CPU. The performance of a workload depends on the application threads accessing data that is local to the CPU the thread is executing on. Automatic NUMA Balancing migrates data on demand to memory nodes that are local to the CPU accessing that data. Depending on the workload, this can dramatically boost performance when using NUMA hardware.

10.1 Implementation

Automatic NUMA balancing happens in three basic steps:

1. A task scanner periodically scans a portion of a task's address space and marks the memory to force a page fault when the data is next accessed.
2. The next access to the data will result in a NUMA Hinting Fault. Based on this fault, the data can be migrated to a memory node associated with the task accessing the memory.
3. To keep a task, the CPU it is using and the memory it is accessing together, the scheduler groups tasks that share data.

The unmapping of data and page fault handling incurs overhead. However, commonly the overhead will be offset by threads accessing data associated with the CPU.

10.2 Configuration

Static configuration has been the recommended way of tuning workloads on NUMA hardware for some time. To do this, memory policies can be set with `numactl`, `taskset` or `cpusets`. NUMA-aware applications can use special APIs. In cases where the static policies have already been created, automatic NUMA balancing should be disabled as the data access should already be local.

`numactl --hardware` will show the memory configuration of the machine and whether it supports NUMA or not. This is example output from a 4-node machine.

```
tux > numactl --hardware
available: 4 nodes (0-3)
node 0 cpus: 0 4 8 12 16 20 24 28 32 36 40 44
node 0 size: 16068 MB
node 0 free: 15909 MB
node 1 cpus: 1 5 9 13 17 21 25 29 33 37 41 45
node 1 size: 16157 MB
node 1 free: 15948 MB
node 2 cpus: 2 6 10 14 18 22 26 30 34 38 42 46
node 2 size: 16157 MB
node 2 free: 15981 MB
node 3 cpus: 3 7 11 15 19 23 27 31 35 39 43 47
node 3 size: 16157 MB
node 3 free: 16028 MB
node distances:
node   0   1   2   3
  0:  10  20  20  20
  1:  20  10  20  20
  2:  20  20  10  20
  3:  20  20  20  10
```

Automatic NUMA balancing can be enabled or disabled for the current session by writing `1` or `0` to `/proc/sys/kernel/numa_balancing` which will enable or disable the feature respectively. To permanently enable or disable it, use the kernel command line option `numa_balancing=[enable|disable]`.

If Automatic NUMA Balancing is enabled, the task scanner behavior can be configured. The task scanner balances the overhead of Automatic NUMA Balancing with the amount of time it takes to identify the best placement of data.

`numa_balancing_scan_delay_ms`

The amount of CPU time a thread must consume before its data is scanned. This prevents creating overhead because of short-lived processes.

numa_balancing_scan_period_min_ms and numa_balancing_scan_period_max_ms

Controls how frequently a task's data is scanned. Depending on the locality of the faults the scan rate will increase or decrease. These settings control the min and max scan rates.

numa_balancing_scan_size_mb

Controls how much address space is scanned when the task scanner is active.

10.3 Monitoring

The most important task is to assign metrics to your workload and measure the performance with Automatic NUMA Balancing enabled and disabled to measure the impact. Profiling tools can be used to monitor local and remote memory accesses if the CPU supports such monitoring. Automatic NUMA Balancing activity can be monitored via the following parameters in /proc/vmstat:

numa_pte_updates

The amount of base pages that were marked for NUMA hinting faults.

numa_huge_pte_updates

The amount of transparent huge pages that were marked for NUMA hinting faults. In combination with numa_pte_updates the total address space that was marked can be calculated.

numa_hint_faults

Records how many NUMA hinting faults were trapped.

numa_hint_faults_local

Shows how many of the hinting faults were to local nodes. In combination with numa_hint_faults, the percentage of local versus remote faults can be calculated. A high percentage of local hinting faults indicates that the workload is closer to being converged.

numa_pages_migrated

Records how many pages were migrated because they were misplaced. As migration is a copying operation, it contributes the largest part of the overhead created by NUMA balancing.

10.4 Impact

The following illustrates a simple test case of a 4-node NUMA machine running the SpecJBB 2005 using a single instance of the JVM with no static tuning around memory policies. Note, however, that the impact for each workload will vary and that this example is based on a pre-release version of openSUSE Leap 12.

	Balancing disabled	Balancing enabled
TPut 1	26629.00 (0.00%)	26507.00 (-0.46%)
TPut 2	55841.00 (0.00%)	53592.00 (-4.03%)
TPut 3	86078.00 (0.00%)	86443.00 (0.42%)
TPut 4	116764.00 (0.00%)	113272.00 (-2.99%)
TPut 5	143916.00 (0.00%)	141581.00 (-1.62%)
TPut 6	166854.00 (0.00%)	166706.00 (-0.09%)
TPut 7	195992.00 (0.00%)	192481.00 (-1.79%)
TPut 8	222045.00 (0.00%)	227143.00 (2.30%)
TPut 9	248872.00 (0.00%)	250123.00 (0.50%)
TPut 10	270934.00 (0.00%)	279314.00 (3.09%)
TPut 11	297217.00 (0.00%)	301878.00 (1.57%)
TPut 12	311021.00 (0.00%)	326048.00 (4.83%)
TPut 13	324145.00 (0.00%)	346855.00 (7.01%)
TPut 14	345973.00 (0.00%)	378741.00 (9.47%)
TPut 15	354199.00 (0.00%)	394268.00 (11.31%)
TPut 16	378016.00 (0.00%)	426782.00 (12.90%)
TPut 17	392553.00 (0.00%)	437772.00 (11.52%)
TPut 18	396630.00 (0.00%)	456715.00 (15.15%)
TPut 19	399114.00 (0.00%)	484020.00 (21.27%)
TPut 20	413907.00 (0.00%)	493618.00 (19.26%)
TPut 21	413173.00 (0.00%)	510386.00 (23.53%)
TPut 22	420256.00 (0.00%)	521016.00 (23.98%)
TPut 23	425581.00 (0.00%)	536214.00 (26.00%)
TPut 24	429052.00 (0.00%)	532469.00 (24.10%)
TPut 25	426127.00 (0.00%)	526548.00 (23.57%)
TPut 26	422428.00 (0.00%)	531994.00 (25.94%)
TPut 27	424378.00 (0.00%)	488340.00 (15.07%)
TPut 28	419338.00 (0.00%)	543016.00 (29.49%)
TPut 29	403347.00 (0.00%)	529178.00 (31.20%)
TPut 30	408681.00 (0.00%)	510621.00 (24.94%)
TPut 31	406496.00 (0.00%)	499781.00 (22.95%)
TPut 32	404931.00 (0.00%)	502313.00 (24.05%)
TPut 33	397353.00 (0.00%)	522418.00 (31.47%)
TPut 34	382271.00 (0.00%)	491989.00 (28.70%)
TPut 35	388965.00 (0.00%)	493012.00 (26.75%)
TPut 36	374702.00 (0.00%)	502677.00 (34.15%)
TPut 37	367578.00 (0.00%)	500588.00 (36.19%)
TPut 38	367121.00 (0.00%)	496977.00 (35.37%)

TPut 39	355956.00 (0.00%)	489430.00 (37.50%)
TPut 40	350855.00 (0.00%)	487802.00 (39.03%)
TPut 41	345001.00 (0.00%)	468021.00 (35.66%)
TPut 42	336177.00 (0.00%)	462260.00 (37.50%)
TPut 43	329169.00 (0.00%)	467906.00 (42.15%)
TPut 44	329475.00 (0.00%)	470784.00 (42.89%)
TPut 45	323845.00 (0.00%)	450739.00 (39.18%)
TPut 46	323878.00 (0.00%)	435457.00 (34.45%)
TPut 47	310524.00 (0.00%)	403914.00 (30.07%)
TPut 48	311843.00 (0.00%)	459017.00 (47.19%)
	Balancing Disabled	Balancing Enabled
Expctd Warehouse	48.00 (0.00%)	48.00 (0.00%)
Expctd Peak Bops	310524.00 (0.00%)	403914.00 (30.07%)
Actual Warehouse	25.00 (0.00%)	29.00 (16.00%)
Actual Peak Bops	429052.00 (0.00%)	543016.00 (26.56%)
SpecJBB Bops	6364.00 (0.00%)	9368.00 (47.20%)
SpecJBB Bops/JVM	6364.00 (0.00%)	9368.00 (47.20%)

Automatic NUMA Balancing simplifies tuning workloads for high performance on NUMA machines. Where possible, it is still recommended to statically tune the workload to partition it within each node. However, in all other cases, automatic NUMA balancing should boost performance.

11 Power Management

Power management aims at reducing operating costs for energy and cooling systems while at the same time keeping the performance of a system at a level that matches the current requirements. Thus, power management is always a matter of balancing the actual performance needs and power saving options for a system. Power management can be implemented and used at different levels of the system. A set of specifications for power management functions of devices and the operating system interface to them has been defined in the Advanced Configuration and Power Interface (ACPI). As power savings in server environments can primarily be achieved at the processor level, this chapter introduces some main concepts and highlights some tools for analyzing and influencing relevant parameters.

11.1 Power Management at CPU Level

At the CPU level, you can control power usage in various ways. For example by using idling power states (C-states), changing CPU frequency (P-states), and throttling the CPU (T-states). The following sections give a short introduction to each approach and its significance for power savings. Detailed specifications can be found at <http://www.acpi.info/spec.htm>.

11.1.1 C-States (Processor Operating States)

Modern processors have several power saving modes called C-states. They reflect the capability of an idle processor to turn off unused components to save power.

When a processor is in the C0 state, it is executing instructions. A processor running in any other C-state is idle. The higher the C number, the deeper the CPU sleep mode: more components are shut down to save power. Deeper sleep states can save large amounts of energy. Their downside is that they introduce latency. This means, it takes more time for the CPU to go back to C0. Depending on workload (threads waking up, triggering CPU usage and then going back to sleep again for a short period of time) and hardware (for example, interrupt activity of a network device), disabling the deepest sleep states can significantly increase overall performance. For details on how to do so, refer to *Section 11.3.2, "Viewing Kernel Idle Statistics with **cpupower**"*.

Some states also have submodes with different power saving latency levels. Which C-states and submodes are supported depends on the respective processor. However, C1 is always available.

Table 11.1, "C-States" gives an overview of the most common C-states.

TABLE 11.1: C-STATES

Mode	Definition
C0	Operational state. CPU fully turned on.
C1	First idle state. Stops CPU main internal clocks via software. Bus interface unit and APIC are kept running at full speed.
C2	Stops CPU main internal clocks via hardware. State in which the processor maintains all software-visible states, but may take longer to wake up through interrupts.
C3	Stops all CPU internal clocks. The processor does not need to keep its cache coherent, but maintains other states. Some processors have variations of the C3 state that differ in how long it takes to wake the processor through interrupts.

To avoid needless power consumption, it is recommended to test your workloads with deep sleep states enabled versus deep sleep states disabled. For more information, refer to *Section 11.3.2, "Viewing Kernel Idle Statistics with cpupower"* or the `cpupower-idle-set(1)` man page.

11.1.2 P-States (Processor Performance States)

While a processor operates (in C0 state), it can be in one of several CPU performance states (P-states). Whereas C-states are idle states (all but C0), P-states are operational states that relate to CPU frequency and voltage.

The higher the P-state, the lower the frequency and voltage at which the processor runs. The number of P-states is processor-specific and the implementation differs across the various types. However, P0 is always the highest-performance state (except for *Section 11.1.3, "Turbo Features"*).

Higher P-state numbers represent slower processor speeds and lower power consumption. For example, a processor in P3 state runs more slowly and uses less power than a processor running in the P1 state. To operate at any P-state, the processor must be in the C0 state, which means that it is working and not idling. The CPU P-states are also defined in the ACPI specification, see <http://www.acpi.info/spec.htm> ↗.

C-states and P-states can vary independently of one another.

11.1.3 Turbo Features

Turbo features allow to dynamically overtick active CPU cores while other cores are in deep sleep states. This increases the performance of active threads while still complying with Thermal Design Power (TDP) limits.

However, the conditions under which a CPU core can use turbo frequencies are architecture-specific. Learn how to evaluate the efficiency of those new features in *Section 11.3, “The cpupower Tools”*.

11.2 In-Kernel Governors

The in-kernel governors belong to the Linux kernel CPUfreq infrastructure and can be used to dynamically scale processor frequencies at runtime. You can think of the governors as a sort of preconfigured power scheme for the CPU. The CPUfreq governors use P-states to change frequencies and lower power consumption. The dynamic governors can switch between CPU frequencies, based on CPU usage, to allow for power savings while not sacrificing performance.

The following governors are available with the CPUfreq subsystem:

Performance Governor

The CPU frequency is statically set to the highest possible for maximum performance. Consequently, saving power is not the focus of this governor.

See also *Section 11.4.1, “Tuning Options for P-States”*.

Powersave Governor

The CPU frequency is statically set to the lowest possible. This can have severe impact on the performance, as the system will never rise above this frequency no matter how busy the processors are. An important exception is the intel_pstate which defaults to the powersave mode. This is due to a hardware-specific decision but functionally it operates similarly to the on-demand governor.

However, using this governor often does not lead to the expected power savings as the highest savings can usually be achieved at idle through entering C-states. With the powersave governor, processes run at the lowest frequency and thus take longer to finish. This means it takes longer until the system can go into an idle C-state.

Tuning options: The range of minimum frequencies available to the governor can be adjusted (for example, with the `cpupower` command line tool).

On-demand Governor

The kernel implementation of a dynamic CPU frequency policy: The governor monitors the processor usage. When it exceeds a certain threshold, the governor will set the frequency to the highest available. If the usage is less than the threshold, the next lowest frequency is used. If the system continues to be underemployed, the frequency is again reduced until the lowest available frequency is set.



Important: Drivers and In-kernel Governors

Not all drivers use the in-kernel governors to dynamically scale power frequency at runtime. For example, the `intel_pstate` driver adjusts power frequency itself. Use the `cpupower frequency-info` command to find out which driver your system uses.

11.3 The cpupower Tools

The `cpupower` tools are designed to give an overview of *all* CPU power-related parameters that are supported on a given machine, including turbo (or boost) states. Use the tool set to view and modify settings of the kernel-related CPUfreq and cpuidle systems and other settings not related to frequency scaling or idle states. The integrated monitoring framework can access both kernel-related parameters and hardware statistics. Therefore, it is ideally suited for performance benchmarks. It also helps you to identify the dependencies between turbo and idle states.

After installing the `cpupower` package, view the available `cpupower` subcommands with `cpupower --help`. Access the general man page with `man cpupower`, and the man pages of the subcommands with `man cpupower-SUBCOMMAND`.

11.3.1 Viewing Current Settings with **cpupower**

The **cpupower frequency-info** command shows the statistics of the cpufreq driver used in the kernel. Additionally, it shows if turbo (boost) states are supported and enabled in the BIOS. Run without any options, it shows an output similar to the following:

EXAMPLE 11.1: EXAMPLE OUTPUT OF **cpupower frequency-info**

```
root # cpupower frequency-info
analyzing CPU 0:
  driver: intel_pstate
  CPUs which run at the same hardware frequency: 0
  CPUs which need to have their frequency coordinated by software: 0
  maximum transition latency: 0.97 ms.
  hardware limits: 1.20 GHz - 3.80 GHz
  available cpufreq governors: performance, powersave
  current policy: frequency should be within 1.20 GHz and 3.80 GHz.
                   The governor "powersave" may decide which speed to use
                   within this range.
  current CPU frequency is 3.40 GHz (asserted by call to hardware).
  boost state support:
    Supported: yes
    Active: yes
    3500 MHz max turbo 4 active cores
    3600 MHz max turbo 3 active cores
    3600 MHz max turbo 2 active cores
    3800 MHz max turbo 1 active cores
```

To get the current values for all CPUs, use **cpupower -c all frequency-info**.

11.3.2 Viewing Kernel Idle Statistics with **cpupower**

The **idle-info** subcommand shows the statistics of the cpuidle driver used in the kernel. It works on all architectures that use the cpuidle kernel framework.

EXAMPLE 11.2: EXAMPLE OUTPUT OF **cpupower idle-info**

```
root # cpupower idle-info
CPUidle driver: intel_idle
CPUidle governor: menu

Analyzing CPU 0:
Number of idle states: 6
Available idle states: POLL C1-SNB C1E-SNB C3-SNB C6-SNB C7-SNB
POLL:
Flags/Description: CPUIDLE CORE POLL IDLE
```



```
Latency: 0
Usage: 163128
Duration: 17585669
C1-SNB:
Flags/Description: MWAIT 0x00
Latency: 2
Usage: 16170005
Duration: 697658910
C1E-SNB:
Flags/Description: MWAIT 0x01
Latency: 10
Usage: 4421617
Duration: 757797385
C3-SNB:
Flags/Description: MWAIT 0x10
Latency: 80
Usage: 2135929
Duration: 735042875
C6-SNB:
Flags/Description: MWAIT 0x20
Latency: 104
Usage: 53268
Duration: 229366052
C7-SNB:
Flags/Description: MWAIT 0x30
Latency: 109
Usage: 62593595
Duration: 324631233978
```

After finding out which processor idle states are supported with `cpupower idle-info`, individual states can be disabled using the `cpupower idle-set` command. Typically one wants to disable the deepest sleep state, for example:

```
root # cpupower idle-set -d 5
```

Or, for disabling all CPUs with latencies equal to or higher than 80:

```
root # cpupower idle-set -D 80
```

11.3.3 Monitoring Kernel and Hardware Statistics with **cpupower**

Use the `monitor` subcommand to report processor topology, and monitor frequency and idle power state statistics over a certain period of time. The default interval is 1 second, but it can be changed with the `-i`. Independent processor sleep states and frequency counters are

implemented in the tool—some retrieved from kernel statistics, others reading out hardware registers. The available monitors depend on the underlying hardware and the system. List them with `cpupower monitor -l`. For a description of the individual monitors, refer to the `cpupower-monitor` man page.

The `monitor` subcommand allows you to execute performance benchmarks. To compare kernel statistics with hardware statistics for specific workloads, concatenate the respective command, for example:

```
cpupower monitor db_test.sh
```

EXAMPLE 11.3: EXAMPLE `cpupower monitor` OUTPUT

```
root # cpupower monitor
|Mperf                || Idle_Stats
 ①                    ②
CPU | C0  | Cx  | Freq || POLL | C1  | C2  | C3
 0 | 3.71| 96.29| 2833|| 0.00| 0.00| 0.02| 96.32
 1 | 100.0| -0.00| 2833|| 0.00| 0.00| 0.00| 0.00
 2 | 9.06| 90.94| 1983|| 0.00| 7.69| 6.98| 76.45
 3 | 7.43| 92.57| 2039|| 0.00| 2.60| 12.62| 77.52
```

- ① Mperf shows the average frequency of a CPU, including boost frequencies, over time. Additionally, it shows the percentage of time the CPU has been active (`C0`) or in any sleep state (`Cx`). As the turbo states are managed by the BIOS, it is impossible to get the frequency values at a given instant. On modern processors with turbo features the Mperf monitor is the only way to find out about the frequency a certain CPU has been running in.
- ② Idle_Stats shows the statistics of the cpuidle kernel subsystem. The kernel updates these values every time an idle state is entered or left. Therefore there can be some inaccuracy when cores are in an idle state for some time when the measure starts or ends.

Apart from the (general) monitors in the example above, other architecture-specific monitors are available. For detailed information, refer to the `cpupower-monitor` man page.

By comparing the values of the individual monitors, you can find correlations and dependencies and evaluate how well the power saving mechanism works for a certain workload. In *Example 11.3* you can see that CPU `0` is idle (the value of `Cx` is near 100%), but runs at a very high frequency. This is because the CPUs `0` and `1` have the same frequency values which means that there is a dependency between them.

11.3.4 Modifying Current Settings with **cpupower**

You can use **cpupower frequency-set** command as root to modify current settings. It allows you to set values for the minimum or maximum CPU frequency the governor may select or to create a new governor. With the -c option, you can also specify for which of the processors the settings should be modified. That makes it easy to use a consistent policy across all processors without adjusting the settings for each processor individually. For more details and the available options, see the man page cpupower-frequency-set or run **cpupower frequency-set --help**.

11.4 Special Tuning Options

The following sections highlight important settings.

11.4.1 Tuning Options for P-States

The CPUfreq subsystem offers several tuning options for P-states: You can switch between the different governors, influence minimum or maximum CPU frequency to be used or change individual governor parameters.

To switch to another governor at runtime, use **cpupower frequency-set** with the -g option. For example, running the following command (as root) will activate the performance governor:

```
root # cpupower frequency-set -g performance
```

To set values for the minimum or maximum CPU frequency the governor may select, use the -d or -u option, respectively.

11.5 Troubleshooting

BIOS options enabled?

To use C-states or P-states, check your BIOS options:

- To use C-states, make sure to enable CPU C State or similar options to benefit from power savings at idle.
- To use P-states and the CPUfreq governors, make sure to enable Processor Performance States options or similar.
- Even if P-states and C-states are available, it is possible that the platform firmware is managing CPU frequencies which may be sub-optimal. For example, if pcc-cpufreq is loaded then the OS is only giving hints to the firmware, which is free to ignore the hints. This can be addressed by selecting "OS Management" or similar for CPU frequency managed in the BIOS. After reboot, an alternative driver will be used but the performance impact should be carefully measured.

In case of a CPU upgrade, make sure to upgrade your BIOS, too. The BIOS needs to know the new CPU and its frequency stepping to pass this information on to the operating system.

Log file information?

Check the systemd journal (see *Book "Reference", Chapter 11 "journalctl: Query the systemd Journal"*) for any output regarding the CPUfreq subsystem. Only severe errors are reported there.

If you suspect problems with the CPUfreq subsystem on your machine, you can also enable additional debug output. To do so, either use cpufreq.debug=7 as boot parameter or execute the following command as root:

```
root # echo 7 > /sys/module/cpufreq/parameters/debug
```

This will cause CPUfreq to log more information to dmesg on state transitions, which is useful for diagnosis. But as this additional output of kernel messages can be rather comprehensive, use it only if you are fairly sure that a problem exists.

11.6 For More Information

Platforms with a Baseboard Management Controller (BMC) may have additional power management configuration options accessible via the service processor. These configurations are vendor specific and therefore not subject of this guide. For more information, refer to the manuals provided by your vendor.

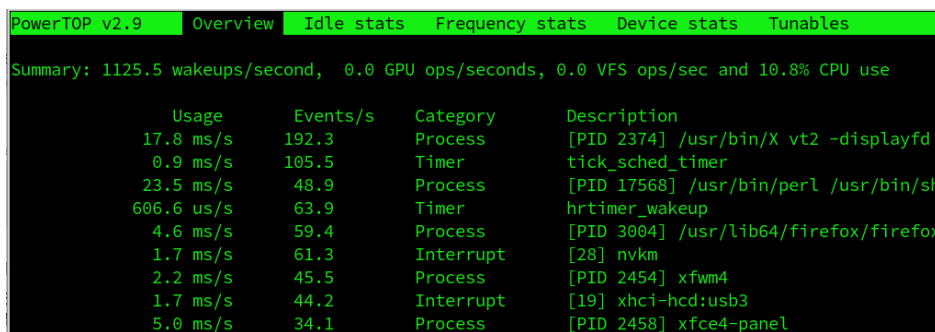
11.7 Monitoring Power Consumption with powerTOP

powerTOP helps to identify the causes of unnecessary high power consumption. This is especially useful for laptops, where minimizing power consumption is more important. It supports both Intel and AMD processors. Install it in the usual way:

```
tux > sudo zypper in powertop
```

powerTOP combines various sources of information (analysis of programs, device drivers, kernel options, number and sources of interrupts waking up processors from sleep states) and provides several ways of viewing them. You can launch it in interactive mode, which runs in an ncurses session (see *Figure 11.1, "powerTOP in Interactive Mode"*):

```
tux > sudo powertop
```



```
PowerTOP v2.9  Overview  Idle stats  Frequency stats  Device stats  Tunables
Summary: 1125.5 wakeups/second, 0.0 GPU ops/seconds, 0.0 VFS ops/sec and 10.8% CPU use
Usage      Events/s  Category  Description
17.8 ms/s  192.3     Process   [PID 2374] /usr/bin/X vt2 -displayfd
0.9 ms/s   105.5     Timer     tick_sched_timer
23.5 ms/s  48.9      Process   [PID 17568] /usr/bin/perl /usr/bin/sh
606.6 us/s 63.9      Timer     hrtimer_wakeup
4.6 ms/s   59.4      Process   [PID 3004] /usr/lib64/firefox/firefox
1.7 ms/s   61.3      Interrupt [28] nvkm
2.2 ms/s   45.5      Process   [PID 2454] xfwm4
1.7 ms/s   44.2      Interrupt [19] xhci-hcd:usb3
5.0 ms/s   34.1      Process   [PID 2458] xfce4-panel
```

FIGURE 11.1: POWERTOP IN INTERACTIVE MODE

powerTOP supports exporting reports to HTML and CSV. The following example generates a single report of a 240-second run:

```
tux > sudo powertop --iteration=1 --time=240 --html=POWERREPORT.HTML
```

It can be useful to run separate reports over time. The following example runs powerTOP 10 times for 20 seconds each time, and creates a separate HTML report for each run:

```
tux > sudo powertop --iteration=10 --time=20 --html=POWERREPORT.HTML
```

This creates 10 time-stamped reports:

```
powerreport-20200108-104512.html
powerreport-20200108-104451.html
powerreport-20200108-104431.html
[...]
```

An HTML report looks like *Figure 11.2, “HTML powerTOP Report”*:

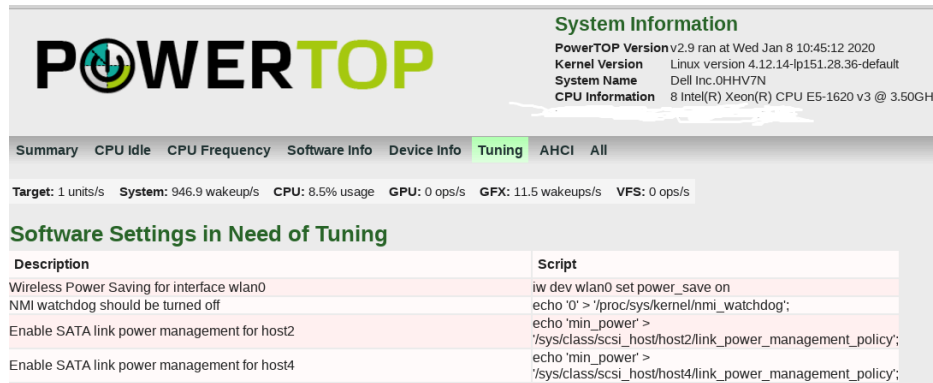


FIGURE 11.2: HTML POWERTOP REPORT

The Tuning tab of the HTML reports, and the Tunables tab in the interactive mode, both provide commands for testing the various power settings. The HTML report prints the commands, which you can copy to a root command line for testing, for example `echo '0' > '/proc/sys/kernel/nmi_watchdog'`. The ncurses mode provides a simple toggle between Good and Bad. Good runs a command to enable power saving, and Bad turns off power saving. Enable all powerTOP settings with one command:

```
tux > sudo powertop --auto-tune
```

None of these changes survive a reboot. To make any changes permanent, use `sysctl`, `udev`, or `systemd` to run your selected commands at boot. powerTOP includes a `systemd` service file, `/usr/lib/systemd/system/powertop.service`. This starts powerTOP with the `--auto-tune` option:

```
ExecStart=/usr/sbin/powertop --auto-tune
```

Test this carefully before launching the `systemd` service, to see if it gives the results that you want. You probably do not want USB keyboards and mice to go into powersave modes because it is a nuisance to continually wake them up, and there may other devices you want left alone. For easier testing and configuration editing, extract the commands from an HTML report with `awk`:

```
tux > awk -F '</?td ?>' '/tune/ { print $4 }' POWERREPORT.HTML
```

In calibrate mode, powerTOP sets up several runs that use different idle settings for backlight, CPU, wifi, USB devices, and disks, and helps to identify optimal brightness settings on battery power:

```
tux > sudo powertop --calibrate
```

You may call a file that creates a workload for more accurate calibration:

```
tux > sudo powertop --calibrate --workload=FILENAME --html=POWERREPORT.HTML
```

For more information, see:

- The powerTOP project page at <https://01.org/powertop> ↗
- *Section 2.6.2, "System Control Parameters: /proc/sys/"*
- *Book "Reference", Chapter 10 "The systemd Daemon"*
- *Book "Reference", Chapter 16 "Dynamic Kernel Device Management with udev"*

V Kernel Tuning

- 12 Tuning I/O Performance **122**
- 13 Tuning the Task Scheduler **127**
- 14 Tuning the Memory Management Subsystem **139**
- 15 Tuning the Network **150**

12 Tuning I/O Performance

I/O scheduling controls how input/output operations will be submitted to storage. openSUSE Leap offers various I/O algorithms—called elevators—suited to different workloads. Elevators can help to reduce seek operations and can prioritize I/O requests.

Choosing the best suited I/O elevator not only depends on the workload, but on the hardware, too. Single ATA disk systems, SSDs, RAID arrays, or network storage systems, for example, each require different tuning strategies.

12.1 Switching I/O Scheduling

openSUSE Leap picks a default I/O scheduler at boot-time, which can be changed on the fly per block device. This makes it possible to set different algorithms, for example, for the device hosting the system partition and the device hosting a database.

The default I/O scheduler is chosen for each device based on whether the device reports to be rotational disk or not. For rotational disks BFQ I/O scheduler is picked. Other devices default to MQ-DEADLINE or NONE.

To change the elevator for a specific device in the running system, run the following command:

```
tux > sudo echo SCHEDULER > /sys/block/DEVICE/queue/scheduler
```

Here, *SCHEDULER* is one of bfq, none, kyber, or mq-deadline. *DEVICE* is the block device (sda for example). Note that this change will not persist during reboot. For permanent I/O scheduler change for a particular device either place the command switching the I/O scheduler into init scripts or add appropriate udev rule into /lib/udev/rules.d/. See /lib/udev/rules.d/60-io-scheduler.rules for an example of such tuning.

12.2 Available I/O Elevators with blk-mq I/O path

Below is a list of elevators available on openSUSE Leap for devices that use the blk-mq I/O path. If an elevator has tunable parameters, they can be set with the command:

```
echo VALUE > /sys/block/DEVICE/queue/iosched/TUNABLE
```

In the command above, *VALUE* is the desired value for the *TUNABLE* and *DEVICE* is the block device.

To find out what elevators are available for a device (sda for example), run the following command (the currently selected scheduler is listed in brackets):

```
tux > cat /sys/block/sda/queue/scheduler
[mq-deadline] kyber bfq none
```

12.2.1 MQ-DEADLINE

MQ-DEADLINE is a latency-oriented I/O scheduler. MQ-DEADLINE has the the following tunable parameters:

TABLE 12.1: MQ-DEADLINE TUNABLE PARAMETERS

File	Description
<u>writes_starved</u>	Controls how many times reads are preferred over writes. A value of <u>3</u> means that three read operations can be done before writes and reads are dispatched on the same selection criteria. Default is <u>3</u> .
<u>read_expire</u>	Sets the deadline (current time plus the <u>read_expire</u> value) for read operations in milliseconds. Default is <u>500</u> .
<u>write_expire</u>	Sets the deadline (current time plus the <u>write_expire</u> value) for write operations in milliseconds. Default is <u>5000</u> .
<u>front_merges</u>	Enables (1) or disables (0) attempts to front merge requests. Default is <u>1</u> .
<u>fifo_batch</u>	Sets the maximum number of requests per batch (deadline expiration is only checked for batches). This parameter allows to balance between latency and throughput. When set to <u>1</u> (that is, one request per batch), it results in "first come, first served" behaviour and usually lowest latency. Higher values usually increase throughput.

File	Description
	Default is <u>16</u> .

12.2.2 NONE

When NONE is selected as I/O elevator option for blk-mq, no I/O scheduler is used, and I/O requests are passed down to the device without further I/O scheduling interaction.

NONE is the default for NVM Express devices. With no overhead compared to other I/O elevator options, it is considered the fastest way of passing down I/O requests on multiple queues to such devices.

There are no tunable parameters for NONE.

12.2.3 BFQ (Budget Fair Queueing)

BFQ is a fairness-oriented scheduler. It is described as "a proportional-share storage-I/O scheduling algorithm based on the slice-by-slice service scheme of CFQ. But BFQ assigns budgets, measured in number of sectors, to processes instead of time slices." (Source: [linux-4.12/block/bfq-iosched.c \(https://github.com/torvalds/linux/blob/6f7da290413ba713f0cdd9f-f1a2a9bb129ef4f6c/block/bfq-iosched.c#L31\)](https://github.com/torvalds/linux/blob/6f7da290413ba713f0cdd9f-f1a2a9bb129ef4f6c/block/bfq-iosched.c#L31))

BFQ allows to assign I/O priorities to tasks which are taken into account during scheduling decisions (see *Section 8.3.3, "Prioritizing Disk Access with ionice"*).

BFQ scheduler has following tunable parameters:

TABLE 12.2: BFQ TUNABLE PARAMETERS

File	Description
<u>slice_idle</u>	Value in milliseconds specifies how long to idle, waiting for next request on an empty queue. Default is <u>8</u> .
<u>slice_idle_us</u>	Same as <u>slice_idle</u> but in microseconds. Default is <u>8000</u> .

File	Description
<u>low_latency</u>	Enables (1) or disables (0) <u>BFQ</u> 's low latency mode. This mode prioritizes certain applications (for example, if interactive) such that they observe lower latency. Default is <u>1</u> .
<u>back_seek_max</u>	Maximum value (in Kbytes) for backward seeking. Default is <u>16384</u> .
<u>back_seek_penalty</u>	Used to compute the cost of backward seeking. Default is <u>2</u> .
<u>fifo_expire_async</u>	Value (in milliseconds) is used to set the timeout of asynchronous requests. Default is <u>250</u> .
<u>fifo_expire_sync</u>	Value in milliseconds specifies the timeout of synchronous requests. Default is <u>125</u> .
<u>timeout_sync</u>	Maximum time in milliseconds that a task (queue) is serviced after it has been selected. Default is <u>124</u> .
<u>max_budget</u>	Limit for number of sectors that are served at maximum within <u>timeout_sync</u> . If set to <u>0</u> <u>BFQ</u> internally calculates a value based on <u>timeout_sync</u> and an estimated peak rate. Default is <u>0</u> (set to auto-tuning).
<u>strict_guarantees</u>	Enables (1) or disables (0) <u>BFQ</u> specific queue handling required to give stricter bandwidth sharing guarantees under certain conditions. Default is <u>0</u> .

12.2.4 KYBER

KYBER is a latency-oriented I/O scheduler. It makes it possible to set target latencies for reads and synchronous writes and throttles I/O requests in order to try to meet these target latencies.

TABLE 12.3: KYBER TUNABLE PARAMETERS

File	Description
<u>read_lat_nsec</u>	Sets the target latency for read operations in nanoseconds. Default is <u>20000000</u> .
<u>write_lat_nsec</u>	Sets the target latency for write operations in nanoseconds. Default is <u>100000000</u> .

12.3 I/O Barrier Tuning

Most file systems (such as XFS, Ext3, or Ext4) send write barriers to disk after fsync or during transaction commits. Write barriers enforce proper ordering of writes, making volatile disk write caches safe to use (at some performance penalty). If your disks are battery-backed in one way or another, disabling barriers can safely improve performance.

Sending write barriers can be disabled using the nobarrier mount option.



Warning: Disabling Barriers Can Lead to Data Loss

Disabling barriers when disks cannot guarantee caches are properly written in case of power failure can lead to severe file system corruption and data loss.

13 Tuning the Task Scheduler

Modern operating systems, such as openSUSE® Leap, normally run many tasks at the same time. For example, you can be searching in a text file while receiving an e-mail and copying a big file to an external hard disk. These simple tasks require many additional processes to be run by the system. To provide each task with its required system resources, the Linux kernel needs a tool to distribute available system resources to individual tasks. And this is exactly what the *task scheduler* does.

The following sections explain the most important terms related to a process scheduling. They also introduce information about the task scheduler policy, scheduling algorithm, description of the task scheduler used by openSUSE Leap, and references to other sources of relevant information.

13.1 Introduction

The Linux kernel controls the way that tasks (or processes) are managed on the system. The task scheduler, sometimes called *process scheduler*, is the part of the kernel that decides which task to run next. It is responsible for best using system resources to guarantee that multiple tasks are being executed simultaneously. This makes it a core component of any multitasking operating system.

13.1.1 Preemption

The theory behind task scheduling is very simple. If there are runnable processes in a system, at least one process must always be running. If there are more runnable processes than processors in a system, not all the processes can be running all the time.

Therefore, some processes need to be stopped temporarily, or *suspended*, so that others can be running again. The scheduler decides what process in the queue will run next.

As already mentioned, Linux, like all other Unix variants, is a *multitasking* operating system. That means that several tasks can be running at the same time. Linux provides a so called *preemptive* multitasking, where the scheduler decides when a process is suspended. This forced suspension is called *preemption*. All Unix flavors have been providing preemptive multitasking since the beginning.

13.1.2 Timeslice

The time period for which a process will be running before it is *preempted* is defined in advance. It is called a *timeslice* of a process and represents the amount of processor time that is provided to each process. By assigning timeslices, the scheduler makes global decisions for the running system, and prevents individual processes from dominating over the processor resources.

13.1.3 Process Priority

The scheduler evaluates processes based on their priority. To calculate the current priority of a process, the task scheduler uses complex algorithms. As a result, each process is given a value according to which it is “allowed” to run on a processor.

13.2 Process Classification

Processes are usually classified according to their purpose and behavior. Although the borderline is not always clearly distinct, generally two criteria are used to sort them. These criteria are independent and do not exclude each other.

One approach is to classify a process either *I/O-bound* or *processor-bound*.

I/O-bound

I/O stands for Input/Output devices, such as keyboards, mice, or optical and hard disks. *I/O-bound processes* spend the majority of time submitting and waiting for requests. They are run very frequently, but for short time intervals, not to block other processes waiting for I/O requests.

processor-bound

On the other hand, *processor-bound* tasks use their time to execute a code, and usually run until they are preempted by the scheduler. They do not block processes waiting for I/O requests, and, therefore, can be run less frequently but for longer time intervals.

Another approach is to divide processes by type into *interactive*, *batch*, and *real-time* processes.

- *Interactive* processes spend a lot of time waiting for I/O requests, such as keyboard or mouse operations. The scheduler must wake up such processes quickly on user request, or the user will find the environment unresponsive. The typical delay is approximately 100 ms. Office applications, text editors or image manipulation programs represent typical interactive processes.
- *Batch* processes often run in the background and do not need to be responsive. They usually receive lower priority from the scheduler. Multimedia converters, database search engines, or log files analyzers are typical examples of batch processes.
- *Real-time* processes must never be blocked by low-priority processes, and the scheduler guarantees a short response time to them. Applications for editing multimedia content are a good example here.

13.3 Completely Fair Scheduler

Since the Linux kernel version 2.6.23, a new approach has been taken to the scheduling of runnable processes. Completely Fair Scheduler (CFS) became the default Linux kernel scheduler. Since then, important changes and improvements have been made. The information in this chapter applies to openSUSE Leap with kernel version 2.6.32 and higher (including 3.x kernels). The scheduler environment was divided into several parts, and three main new features were introduced:

Modular Scheduler Core

The core of the scheduler was enhanced with *scheduling classes*. These classes are modular and represent scheduling policies.

Completely Fair Scheduler

Introduced in kernel 2.6.23 and extended in 2.6.24, CFS tries to assure that each process obtains its “fair” share of the processor time.

Group Scheduling

For example, if you split processes into groups according to which user is running them, CFS tries to provide each of these groups with the same amount of processor time.

As a result, CFS brings optimized scheduling for both servers and desktops.

13.3.1 How CFS Works

CFS tries to guarantee a fair approach to each runnable task. To find the most balanced way of task scheduling, it uses the concept of *red-black tree*. A red-black tree is a type of self-balancing data search tree which provides inserting and removing entries in a reasonable way so that it remains well balanced. For more information, see the wiki pages of [Red-black tree \(http://en.wikipedia.org/wiki/Red_black_tree\)](http://en.wikipedia.org/wiki/Red_black_tree).

When CFS schedules a task it accumulates “virtual runtime” or *vruntime*. The next task picked to run is always the task with the minimum accumulated vruntime so far. By balancing the red-black tree when tasks are inserted into the *run queue* (a planned time line of processes to be executed next), the task with the minimum vruntime is always the first entry in the red-black tree.

The amount of vruntime a task accrues is related to its priority. High priority tasks gain vruntime at a slower rate than low priority tasks, which results in high priority tasks being picked to run on the processor more often.

13.3.2 Grouping Processes

Since the Linux kernel version 2.6.24, CFS can be tuned to be fair to groups rather than to tasks only. Runnable tasks are then grouped to form entities, and CFS tries to be fair to these entities instead of individual runnable tasks. The scheduler also tries to be fair to individual tasks within these entities.

The kernel scheduler lets you group runnable tasks using control groups. For more information, see [Chapter 9, Kernel Control Groups](#).

13.3.3 Kernel Configuration Options

Basic aspects of the task scheduler behavior can be set through the kernel configuration options. Setting these options is part of the kernel compilation process. Because kernel compilation process is a complex task and out of this document's scope, refer to relevant source of information.



Warning: Kernel Compilation

If you run openSUSE Leap on a kernel that was not shipped with it, for example on a self-compiled kernel, you lose the entire support entitlement.

13.3.4 Terminology

Documents regarding task scheduling policy often use several technical terms which you need to know to understand the information correctly. Here are some:

Latency

Delay between the time a process is scheduled to run and the actual process execution.

Granularity

The relation between granularity and latency can be expressed by the following equation:

$$\text{gran} = (\text{lat} / \text{rtasks}) - (\text{lat} / \text{rtasks} / \text{rtasks})$$

where *gran* stands for granularity, *lat* stand for latency, and *rtasks* is the number of running tasks.

13.3.4.1 Scheduling Policies

The Linux kernel supports the following scheduling policies:

SCHED_FIFO

Scheduling policy designed for special time-critical applications. It uses the First In-First Out scheduling algorithm.

SCHED_BATCH

Scheduling policy designed for CPU-intensive tasks.

SCHED_IDLE

Scheduling policy intended for *very* low prioritized tasks.

SCHED_OTHER

Default Linux time-sharing scheduling policy used by the majority of processes.

SCHED_RR

Similar to `SCHED_FIFO`, but uses the Round Robin scheduling algorithm.

13.3.5 Changing Real-time Attributes of Processes with `chrt`

The `chrt` command sets or retrieves the real-time scheduling attributes of a running process, or runs a command with the specified attributes. You can get or retrieve both the scheduling policy and priority of a process.

In the following examples, a process whose PID is 16244 is used.

To *retrieve* the real-time attributes of an existing task:

```
root # chrt -p 16244
pid 16244's current scheduling policy: SCHED_OTHER
pid 16244's current scheduling priority: 0
```

Before setting a new scheduling policy on the process, you need to find out the minimum and maximum valid priorities for each scheduling algorithm:

```
root # chrt -m
SCHED_SCHED_OTHER min/max priority : 0/0
SCHED_SCHED_FIFO min/max priority : 1/99
SCHED_SCHED_RR min/max priority : 1/99
SCHED_SCHED_BATCH min/max priority : 0/0
SCHED_SCHED_IDLE min/max priority : 0/0
```

In the above example, `SCHED_OTHER`, `SCHED_BATCH`, `SCHED_IDLE` policies only allow for priority 0, while that of `SCHED_FIFO` and `SCHED_RR` can range from 1 to 99.

To set `SCHED_BATCH` scheduling policy:

```
root # chrt -b -p 0 16244
pid 16244's current scheduling policy: SCHED_BATCH
pid 16244's current scheduling priority: 0
```

For more information on `chrt`, see its man page (`man 1 chrt`).

13.3.6 Runtime Tuning with `sysctl`

The `sysctl` interface for examining and changing kernel parameters at runtime introduces important variables by means of which you can change the default behavior of the task scheduler. The syntax of the `sysctl` is simple, and all the following commands must be entered on the command line as `root`.

To read a value from a kernel variable, enter

```
root # sysctl VARIABLE
```

To assign a value, enter

```
root # sysctl VARIABLE=VALUE
```

To get a list of all scheduler related **sysctl** variables, enter

```
root # sysctl -A | grep "sched" | grep -v "domain"
```

```
root # sysctl -A | grep "sched" | grep -v "domain"
kernel.sched_cfs_bandwidth_slice_us = 5000
kernel.sched_child_runs_first = 0
kernel.sched_compat_yield = 0
kernel.sched_latency_ns = 24000000
kernel.sched_migration_cost_ns = 500000
kernel.sched_min_granularity_ns = 8000000
kernel.sched_nr_migrate = 32
kernel.sched_rr_timeslice_ms = 25
kernel.sched_rt_period_us = 1000000
kernel.sched_rt_runtime_us = 950000
kernel.sched_schedstats = 0
kernel.sched_shares_window_ns = 10000000
kernel.sched_time_avg_ms = 1000
kernel.sched_tunable_scaling = 1
kernel.sched_wakeup_granularity_ns = 10000000
```

Note that variables ending with “_ns” and “_us” accept values in nanoseconds and microseconds, respectively.

A list of the most important task scheduler **sysctl** tuning variables (located at [/proc/sys/kernel/](#)) with a short description follows:

sched_cfs_bandwidth_slice_us

When CFS bandwidth control is in use, this parameter controls the amount of run-time (bandwidth) transferred to a run queue from the task's control group bandwidth pool. Small values allow the global bandwidth to be shared in a fine-grained manner among tasks, larger values reduce transfer overhead. See <https://www.kernel.org/doc/Documentation/scheduler/sched-bwc.txt>.

sched_child_runs_first

A freshly forked child runs before the parent continues execution. Setting this parameter to 1 is beneficial for an application in which the child performs an execution after fork. For example `make -j<NO_CPUS>` performs better when `sched_child_runs_first` is turned off. The default value is 0.

sched_compat_yield

Enables the aggressive CPU yielding behavior of the old $O(1)$ scheduler by moving the relinquishing task to the end of the runnable queue (right-most position in the red-black tree). Applications that depend on the `sched_yield(2)` syscall behavior may see performance improvements by giving other processes a chance to run when there are highly contended resources (such as locks). On the other hand, given that this call occurs in context switching, misusing the call can hurt the workload. Only use it when you see a drop in performance. The default value is 0.

sched_migration_cost_ns

Amount of time after the last execution that a task is considered to be “cache hot” in migration decisions. A “hot” task is less likely to be migrated to another CPU, so increasing this variable reduces task migrations. The default value is 500000 (ns).

If the CPU idle time is higher than expected when there are runnable processes, try reducing this value. If tasks bounce between CPUs or nodes too often, try increasing it.

sched_latency_ns

Targeted preemption latency for CPU bound tasks. Increasing this variable increases a CPU bound task's timeslice. A task's timeslice is its weighted fair share of the scheduling period: $\text{timeslice} = \text{scheduling period} * (\text{task's weight} / \text{total weight of tasks in the run queue})$

The task's weight depends on the task's nice level and the scheduling policy. Minimum task weight for a SCHED_OTHER task is 15, corresponding to nice 19. The maximum task weight is 88761, corresponding to nice -20.

Timeslices become smaller as the load increases. When the number of runnable tasks exceeds $\text{sched_latency_ns} / \text{sched_min_granularity_ns}$, the slice becomes $\text{number_of_running_tasks} * \text{sched_min_granularity_ns}$. Prior to that, the slice is equal to sched_latency_ns.

This value also specifies the maximum amount of time during which a sleeping task is considered to be running for entitlement calculations. Increasing this variable increases the amount of time a waking task may consume before being preempted, thus increasing scheduler latency for CPU bound tasks. The default value is 6000000 (ns).

sched_min_granularity_ns

Minimal preemption granularity for CPU bound tasks. See [sched_latency_ns](#) for details. The default value is [4000000](#) (ns).

[sched_wakeup_granularity_ns](#)

The wake-up preemption granularity. Increasing this variable reduces wake-up preemption, reducing disturbance of compute bound tasks. Lowering it improves wake-up latency and throughput for latency critical tasks, particularly when a short duty cycle load component must compete with CPU bound components. The default value is [2500000](#) (ns).



Warning: Setting the Right Wake-up Granularity Value

Settings larger than half of [sched_latency_ns](#) will result in no wake-up preemption. Short duty cycle tasks will be unable to compete with CPU hogs effectively.

[sched_rr_timeslice_ms](#)

Quantum that SCHED_RR tasks are allowed to run before they are preempted and put to the end of the task list.

[sched_rt_period_us](#)

Period over which real-time task bandwidth enforcement is measured. The default value is [1000000](#) (μ s).

[sched_rt_runtime_us](#)

Quantum allocated to real-time tasks during [sched_rt_period_us](#). Setting to -1 disables RT bandwidth enforcement. By default, RT tasks may consume 95%CPU/sec, thus leaving 5%CPU/sec or 0.05s to be used by SCHED_OTHER tasks. The default value is [950000](#) (μ s).

[sched_nr_migrate](#)

Controls how many tasks can be migrated across processors for load-balancing purposes. Because balancing iterates the runqueue with interrupts disabled (softirq), it can incur in irq-latency penalties for real-time tasks. Therefore increasing this value may give a performance boost to large SCHED_OTHER threads at the expense of increased irq-latencies for real-time tasks. The default value is [32](#).

[sched_time_avg_ms](#)

This parameter sets the period over which the time spent running real-time tasks is averaged. That average assists CFS in making load-balancing decisions and gives an indication of how busy a CPU is with high-priority real-time tasks.

The optimal setting for this parameter is highly workload dependent and depends, among other things, on how frequently real-time tasks are running and for how long.

13.3.7 Debugging Interface and Scheduler Statistics

CFS comes with a new improved debugging interface, and provides runtime statistics information. Relevant files were added to the `/proc` file system, which can be examined simply with the `cat` or `less` command. A list of the related `/proc` files follows with their short description:

`/proc/sched_debug`

Contains the current values of all tunable variables (see [Section 13.3.6, "Runtime Tuning with sysctl"](#)) that affect the task scheduler behavior, CFS statistics, and information about the run queues (CFS, RT and deadline) on all available processors. A summary of the task running on each processor is also shown, with the task name and PID, along with scheduler specific statistics. The first being `tree-key` column, it indicates the task's virtual runtime, and its name comes from the kernel sorting all runnable tasks by this key in a red-black tree. The `switches` column indicates the total number of switches (involuntary or not), and naturally the `prio` refers to the process priority. The `wait-time` value indicates the amount of time the task waited to be scheduled. Finally both `sum-exec` and `sum-sleep` account for the total amount of time (in nanoseconds) the task was running on the processor or asleep, respectively.

```
root # cat /proc/sched_debug
Sched Debug Version: v0.11, 4.4.21-64-default #1
ktime                               : 23533900.395978
sched_clk                            : 23543587.726648
cpu_clk                              : 23533900.396165
jiffies                              : 4300775771
sched_clock_stable                   : 0

sysctl_sched
  .sysctl_sched_latency               : 6.000000
  .sysctl_sched_min_granularity      : 2.000000
  .sysctl_sched_wakeup_granularity   : 2.500000
  .sysctl_sched_child_runs_first     : 0
  .sysctl_sched_features             : 154871
  .sysctl_sched_tunable_scaling      : 1 (logarithmic)

cpu#0, 2666.762 MHz
  .nr_running                        : 1
  .load                              : 1024
```

```

.nr_switches          : 1918946
[...]

cfs_rq[0]:/
 .exec_clock          : 170176.383770
 .MIN_vruntime        : 0.000001
 .min_vruntime        : 347375.854324
 .max_vruntime        : 0.000001
[...]

rt_rq[0]:/
 .rt_nr_running       : 0
 .rt_throttled        : 0
 .rt_time             : 0.000000
 .rt_runtime          : 950.000000

dl_rq[0]:
 .dl_nr_running       : 0

task  PID          tree-key  switches  prio    wait-time    [...]
-----
R  cc1 63477      98876.717832    197    120    0.000000    ...

```

/proc/schedstat

Displays statistics relevant to the current run queue. Also domain-specific statistics for SMP systems are displayed for all connected processors. Because the output format is not user-friendly, read the contents of </usr/src/linux/Documentation/scheduler/sched-stats.txt> for more information.

/proc/PID/sched

Displays scheduling information on the process with id PID.

```

root # cat /proc/$(pidof gdm)/sched
gdm (744, #threads: 3)
-----
se.exec_start          :          8888.758381
se.vruntime            :          6062.853815
se.sum_exec_runtime    :           7.836043
se.statistics.wait_start :          0.000000
se.statistics.sleep_start :          8888.758381
se.statistics.block_start :          0.000000
se.statistics.sleep_max :          1965.987638
[...]
se.avg.decay_count     :           8477
policy                 :           0
prio                   :           120

```



```
clock-delta : 128
mm->numa_scan_seq : 0
numa_migrations, 0
numa_faults_memory, 0, 0, 1, 0, -1
numa_faults_memory, 1, 0, 0, 0, -1
```

13.4 For More Information

To get a compact knowledge about Linux kernel task scheduling, you need to explore several information sources. Here are some:

- For task scheduler System Calls description, see the relevant manual page (for example [man 2 sched_setaffinity](#)).
- General information on scheduling is described in [Scheduling \(http://en.wikipedia.org/wiki/Scheduling_\(computing\)\)](http://en.wikipedia.org/wiki/Scheduling_(computing)) [wiki page](#).
- A useful lecture on Linux scheduler policy and algorithm is available in <http://www.inf.fu-berlin.de/lehre/SS01/OS/Lectures/Lecture08.pdf> [wiki](#).
- A good overview of Linux process scheduling is given in *Linux Kernel Development* by Robert Love (ISBN-10: 0-672-32512-8). See <http://www.informit.com/articles/article.aspx?p=101760> [wiki](#).
- A very comprehensive overview of the Linux kernel internals is given in *Understanding the Linux Kernel* by Daniel P. Bovet and Marco Cesati (ISBN 978-0-596-00565-8).
- Technical information about task scheduler is covered in files under [/usr/src/linux/Documentation/scheduler](#).

14 Tuning the Memory Management Subsystem

To understand and tune the memory management behavior of the kernel, it is important to first have an overview of how it works and cooperates with other subsystems.

The memory management subsystem, also called the virtual memory manager, will subsequently be called “VM”. The role of the VM is to manage the allocation of physical memory (RAM) for the entire kernel and user programs. It is also responsible for providing a virtual memory environment for user processes (managed via POSIX APIs with Linux extensions). Finally, the VM is responsible for freeing up RAM when there is a shortage, either by trimming caches or swapping out “anonymous” memory.

The most important thing to understand when examining and tuning VM is how its caches are managed. The basic goal of the VM's caches is to minimize the cost of I/O as generated by swapping and file system operations (including network file systems). This is achieved by avoiding I/O completely, or by submitting I/O in better patterns.

Free memory will be used and filled up by these caches as required. The more memory is available for caches and anonymous memory, the more effectively caches and swapping will operate. However, if a memory shortage is encountered, caches will be trimmed or memory will be swapped out.

For a particular workload, the first thing that can be done to improve performance is to increase memory and reduce the frequency that memory must be trimmed or swapped. The second thing is to change the way caches are managed by changing kernel parameters.

Finally, the workload itself should be examined and tuned as well. If an application is allowed to run more processes or threads, effectiveness of VM caches can be reduced, if each process is operating in its own area of the file system. Memory overheads are also increased. If applications allocate their own buffers or caches, larger caches will mean that less memory is available for VM caches. However, more processes and threads can mean more opportunity to overlap and pipeline I/O, and may take better advantage of multiple cores. Experimentation will be required for the best results.

14.1 Memory Usage

Memory allocations in general can be characterized as “pinned” (also known as “unreclaimable”), “reclaimable” or “swappable”.

14.1.1 Anonymous Memory

Anonymous memory tends to be program heap and stack memory (for example, `>malloc()`). It is reclaimable, except in special cases such as `mlock` or if there is no available swap space. Anonymous memory must be written to swap before it can be reclaimed. Swap I/O (both swapping in and swapping out pages) tends to be less efficient than pagecache I/O, because of allocation and access patterns.

14.1.2 Pagecache

A cache of file data. When a file is read from disk or network, the contents are stored in pagecache. No disk or network access is required, if the contents are up-to-date in pagecache. `tmpfs` and shared memory segments count toward pagecache.

When a file is written to, the new data is stored in pagecache before being written back to a disk or the network (making it a write-back cache). When a page has new data not written back yet, it is called “dirty”. Pages not classified as dirty are “clean”. Clean pagecache pages can be reclaimed if there is a memory shortage by simply freeing them. Dirty pages must first be made clean before being reclaimed.

14.1.3 Buffercache

This is a type of pagecache for block devices (for example, `/dev/sda`). A file system typically uses the buffercache when accessing its on-disk metadata structures such as inode tables, allocation bitmaps, and so forth. Buffercache can be reclaimed similarly to pagecache.

14.1.4 Buffer Heads

Buffer heads are small auxiliary structures that tend to be allocated upon pagecache access. They can generally be reclaimed easily when the pagecache or buffercache pages are clean.

14.1.5 Writeback

As applications write to files, the pagecache becomes dirty and the buffercache may become dirty. When the amount of dirty memory reaches a specified number of pages in bytes (`vm.dirty_background_bytes`), or when the amount of dirty memory reaches a specific ratio to total

memory (*vm.dirty_background_ratio*), or when the pages have been dirty for longer than a specified amount of time (*vm.dirty_expire_centisecs*), the kernel begins writeback of pages starting with files that had the pages dirtied first. The background bytes and ratios are mutually exclusive and setting one will overwrite the other. Flusher threads perform writeback in the background and allow applications to continue running. If the I/O cannot keep up with applications dirtying pagecache, and dirty data reaches a critical setting (*vm.dirty_bytes* or *vm.dirty_ratio*), then applications begin to be throttled to prevent dirty data exceeding this threshold.

14.1.6 Readahead

The VM monitors file access patterns and may attempt to perform readahead. Readahead reads pages into the pagecache from the file system that have not been requested yet. It is done to allow fewer, larger I/O requests to be submitted (more efficient). And for I/O to be pipelined (I/O performed at the same time as the application is running).

14.1.7 VFS caches

14.1.7.1 Inode Cache

This is an in-memory cache of the inode structures for each file system. These contain attributes such as the file size, permissions and ownership, and pointers to the file data.

14.1.7.2 Directory Entry Cache

This is an in-memory cache of the directory entries in the system. These contain a name (the name of a file), the inode which it refers to, and children entries. This cache is used when traversing the directory structure and accessing a file by name.

14.2 Reducing Memory Usage

14.2.1 Reducing malloc (Anonymous) Usage

Applications running on openSUSE Leap 15.2 can allocate more memory compared to openSUSE Leap 10. This is because of `glibc` changing its default behavior while allocating user space memory. See http://www.gnu.org/s/libc/manual/html_node/Malloc-Tunable-Parameters.html for explanation of these parameters.

To restore a openSUSE Leap 10-like behavior, `M_MMAP_THRESHOLD` should be set to `128*1024`. This can be done with `mallopt()` call from the application, or via setting `MALLOCM_MMAP_THRESHOLD` environment variable before running the application.

14.2.2 Reducing Kernel Memory Overheads

Kernel memory that is reclaimable (caches, described above) will be trimmed automatically during memory shortages. Most other kernel memory cannot be easily reduced but is a property of the workload given to the kernel.

Reducing the requirements of the user space workload will reduce the kernel memory usage (fewer processes, fewer open files and sockets, etc.)

14.2.3 Memory Controller (Memory Cgroups)

If the memory cgroups feature is not needed, it can be switched off by passing `cgroup_disable=memory` on the kernel command line, reducing memory consumption of the kernel a bit. There is also a slight performance benefit as there is a small amount of accounting overhead when memory cgroups are available even if none are configured.

14.3 Virtual Memory Manager (VM) Tunable Parameters

When tuning the VM it should be understood that some changes will take time to affect the workload and take full effect. If the workload changes throughout the day, it may behave very differently at different times. A change that increases throughput under some conditions may decrease it under other conditions.

14.3.1 Reclaim Ratios

/proc/sys/vm/swappiness

This control is used to define how aggressively the kernel swaps out anonymous memory relative to pagecache and other caches. Increasing the value increases the amount of swapping. The default value is 60.

Swap I/O tends to be much less efficient than other I/O. However, some pagecache pages will be accessed much more frequently than less used anonymous memory. The right balance should be found here.

If swap activity is observed during slowdowns, it may be worth reducing this parameter. If there is a lot of I/O activity and the amount of pagecache in the system is rather small, or if there are large dormant applications running, increasing this value might improve performance.

Note that the more data is swapped out, the longer the system will take to swap data back in when it is needed.

/proc/sys/vm/vfs_cache_pressure

This variable controls the tendency of the kernel to reclaim the memory which is used for caching of VFS caches, versus pagecache and swap. Increasing this value increases the rate at which VFS caches are reclaimed.

It is difficult to know when this should be changed, other than by experimentation. The **slabtop** command (part of the package procps) shows top memory objects used by the kernel. The vfs caches are the "dentry" and the "*_inode_cache" objects. If these are consuming a large amount of memory in relation to pagecache, it may be worth trying to increase pressure. Could also help to reduce swapping. The default value is 100.

/proc/sys/vm/min_free_kbytes

This controls the amount of memory that is kept free for use by special reserves including “atomic” allocations (those which cannot wait for reclaim). This should not normally be lowered unless the system is being very carefully tuned for memory usage (normally useful for embedded rather than server applications). If “page allocation failure” messages and stack traces are frequently seen in logs, `min_free_kbytes` could be increased until the errors disappear. There is no need for concern, if these messages are very infrequent. The default value depends on the amount of RAM.

/proc/sys/vm/watermark_scale_factor

Broadly speaking, free memory has high, low and min watermarks. When the low watermark is reached then **kswapd** wakes to reclaim memory in the background. It stays awake until free memory reaches the high watermark. Applications will stall and reclaim memory when the low watermark is reached.

The `watermark_scale_factor` defines the amount of memory left in a node/system before `kswapd` is woken up and how much memory needs to be free before `kswapd` goes back to sleep. The unit is in fractions of 10,000. The default value of 10 means the distances between watermarks are 0.1% of the available memory in the node/system. The maximum value is 1000, or 10% of memory.

Workloads that frequently stall in direct reclaim, accounted by `allocstall` in `/proc/vmstat`, may benefit from altering this parameter. Similarly, if **kswapd** is sleeping prematurely, as accounted for by `kswapd_low_wmark_hit_quickly`, then it may indicate that the number of pages kept free to avoid stalls is too low.

14.3.2 Writeback Parameters

One important change in writeback behavior since openSUSE Leap 10 is that modification to file-backed `mmap()` memory is accounted immediately as dirty memory (and subject to writeback). Whereas previously it would only be subject to writeback after it was unmapped, upon an `msync()` system call, or under heavy memory pressure.

Some applications do not expect `mmap` modifications to be subject to such writeback behavior, and performance can be reduced. Berkeley DB (and applications using it) is one known example that can cause problems. Increasing writeback ratios and times can improve this type of slowdown.

/proc/sys/vm/dirty_background_ratio

This is the percentage of the total amount of free and reclaimable memory. When the amount of dirty pagecache exceeds this percentage, writeback threads start writing back dirty memory. The default value is 10 (%).

/proc/sys/vm/dirty_background_bytes

This contains the amount of dirty memory at which the background kernel flusher threads will start writeback. dirty_background_bytes is the counterpart of dirty_background_ratio. If one of them is set, the other one will automatically be read as 0.

/proc/sys/vm/dirty_ratio

Similar percentage value as for dirty_background_ratio. When this is exceeded, applications that want to write to the pagecache are blocked and wait for kernel background flusher threads to reduce the amount of dirty memory. The default value is 20 (%).

/proc/sys/vm/dirty_bytes

This file controls the same tunable as dirty_ratio however the amount of dirty memory is in bytes as opposed to a percentage of reclaimable memory. Since both dirty_ratio and dirty_bytes control the same tunable, if one of them is set, the other one will automatically be read as 0. The minimum value allowed for dirty_bytes is two pages (in bytes); any value lower than this limit will be ignored and the old configuration will be retained.

/proc/sys/vm/dirty_expire_centisecs

Data which has been dirty in-memory for longer than this interval will be written out next time a flusher thread wakes up. Expiration is measured based on the modification time of a file's inode. Therefore, multiple dirtied pages from the same file will all be written when the interval is exceeded.

dirty_background_ratio and dirty_ratio together determine the pagecache writeback behavior. If these values are increased, more dirty memory is kept in the system for a longer time. With more dirty memory allowed in the system, the chance to improve throughput by avoiding writeback I/O and to submitting more optimal I/O patterns increases. However, more dirty memory can either harm latency when memory needs to be reclaimed or at points of data integrity (“synchronization points”) when it needs to be written back to disk.

14.3.3 Readahead Parameters

/sys/block/<bdev>/queue/read_ahead_kb

If one or more processes are sequentially reading a file, the kernel reads some data in advance (ahead) to reduce the amount of time that processes need to wait for data to be available. The actual amount of data being read in advance is computed dynamically, based on how much "sequential" the I/O seems to be. This parameter sets the maximum amount of data that the kernel reads ahead for a single file. If you observe that large sequential reads from a file are not fast enough, you can try increasing this value. Increasing it too far may result in readahead thrashing where pagecache used for readahead is reclaimed before it can be used, or slowdowns because of a large amount of useless I/O. The default value is 512 (KB).

14.3.4 Transparent Huge Page Parameters

Transparent Huge Pages (THP) provide a way to dynamically allocate huge pages either on-demand by the process or deferring the allocation until later via the **khugepaged** kernel thread. This method is distinct from the use of hugetlbfs to manually manage their allocation and use. Workloads with contiguous memory access patterns can benefit greatly from THP. A 1000-fold decrease in page faults can be observed when running synthetic workloads with contiguous memory access patterns.

There are cases when THP may be undesirable. Workloads with sparse memory access patterns can perform poorly with THP due to excessive memory usage. For example, 2 MB of memory may be used at fault time instead of 4 KB for each fault and ultimately lead to premature page reclaim. On releases older than openSUSE Leap 42.2, it was possible for an application to stall for long periods of time trying to allocate a THP which frequently led to a recommendation of disabling THP. Such recommendations should be re-evaluated for openSUSE Leap 42.3

The behavior of THP may be configured via the transparent_hugepage= kernel parameter or via sysfs. For example, it may be disabled by adding the kernel parameter transparent_hugepage=never, rebuilding your grub2 configuration, and rebooting. Verify if THP is disabled with:

```
root # cat /sys/kernel/mm/transparent_hugepage/enabled
always madvise [never]
```

If disabled, the value never is shown in square brackets like in the example above. A value of always will always try and use THP at fault time but defer to **khugepaged** if the allocation fails. A value of madvise will only allocate THP for address spaces explicitly specified by an application.

/sys/kernel/mm/transparent_hugepage/defrag

This parameter controls how much effort an application commits when allocating a THP. A value of always is the default for openSUSE 42.1 and earlier releases that supported THP. If a THP is not available, the application will try to defragment memory. It potentially incurs large stalls in an application if the memory is fragmented and a THP is not available. A value of madvise means that THP allocation requests will only defragment if the application explicitly requests it. This is the default for openSUSE 42.2 and later releases.

defer is only available on openSUSE 42.2 and later releases. If a THP is not available, the application will fall back to using small pages if a THP is not available. It will wake the kswapd and kcompactd kernel threads to defragment memory in the background and a THP will be allocated later by khugepaged.

The final option never will use small pages if a THP is unavailable but no other action will take place.

14.3.5 khugepaged Parameters

khugepaged will be automatically started when transparent_hugepage is set to always or madvise, and it will be automatically shut down if it is set to never. Normally this runs at low frequency but the behavior can be tuned.

/sys/kernel/mm/transparent_hugepage/khugepaged/defrag

A value of 0 will disable khugepaged even though THP may still be used at fault time. This may be important for latency-sensitive applications that benefit from THP but cannot tolerate a stall if khugepaged tries to update an application memory usage.

/sys/kernel/mm/transparent_hugepage/khugepaged/pages_to_scan

This parameter controls how many pages are scanned by khugepaged in a single pass. A scan identifies small pages that can be reallocated as THP. Increasing this value will allocate THP in the background faster at the cost of CPU usage.

/sys/kernel/mm/transparent_hugepage/khugepaged/scan_sleep_millisecs

khugepaged sleeps for a short interval specified by this parameter after each pass to limit how much CPU usage is used. Reducing this value will allocate THP in the background faster at the cost of CPU usage. A value of 0 will force continual scanning.

/sys/kernel/mm/transparent_hugepage/khugepaged/alloc_sleep_millisecs

This parameter controls how long khugepaged will sleep in the event it fails to allocate a THP in the background waiting for kswapd and kcompactd to take action.

The remaining parameters for **khugepaged** are rarely useful for performance tuning but are fully documented in </usr/src/linux/Documentation/vm/transhuge.txt>

14.3.6 Further VM Parameters

For the complete list of the VM tunable parameters, see </usr/src/linux/Documentation/sysctl/vm.txt> (available after having installed the [kernel-source](#) package).

14.4 Monitoring VM Behavior

Some simple tools that can help monitor VM behavior:

1. **vmstat**: This tool gives a good overview of what the VM is doing. See [Section 2.1.1, “vmstat”](#) for details.
2. [/proc/meminfo](#): This file gives a detailed breakdown of where memory is being used. See [Section 2.4.2, “Detailed Memory Usage: /proc/meminfo”](#) for details.
3. **slabtop**: This tool provides detailed information about kernel slab memory usage. `buffer_head`, `dentry`, `inode_cache`, `ext3_inode_cache`, etc. are the major caches. This command is available with the package [procps](#).
4. [/proc/vmstat](#): This file gives a detailed breakdown of internal VM behavior. The information contained within is implementation specific and may not always be available. Some information is duplicated in [/proc/meminfo](#) and other information can be presented in a friendly fashion by utilities. For maximum utility, this file needs to be monitored over time to observe rates of change. The most important pieces of information that are hard to derive from other sources are as follows:

[pgscan_kswapd_*](#), [pgsteal_kswapd_*](#)

These report respectively the number of pages scanned and reclaimed by **kswapd** since the system started. The ratio between these values can be interpreted as the reclaim efficiency with a low efficiency implying that the system is struggling to reclaim memory and may be thrashing. Light activity here is generally not something to be concerned with.

[pgscan_direct_*](#), [pgsteal_direct_*](#)

These report respectively the number of pages scanned and reclaimed by an application directly. This is correlated with increases in the `allocstall` counter. This is more serious than `kswapt` activity as these events indicate that processes are stalling. Heavy activity here combined with `kswapt` and high rates of `pgpgin`, `pgpout` and/or high rates of `pswapin` or `pswpout` are signs that a system is thrashing heavily. More detailed information can be obtained using tracepoints.

thp_fault_alloc, thp_fault_fallback

These counters correspond to how many THPs were allocated directly by an application and how many times a THP was not available and small pages were used. Generally a high fallback rate is harmless unless the application is very sensitive to TLB pressure.

thp_collapse_alloc, thp_collapse_alloc_failed

These counters correspond to how many THPs were allocated by `khugepaged` and how many times a THP was not available and small pages were used. A high fallback rate implies that the system is fragmented and THPs are not being used even when the memory usage by applications would allow them. It is only a problem for applications that are sensitive to TLB pressure.

compact*_scanned, compact_stall, compact_fail, compact_success

These counters may increase when THP is enabled and the system is fragmented. `compact_stall` is incremented when an application stalls allocating THP. The remaining counters account for pages scanned, the number of defragmentation events that succeeded or failed.

15 Tuning the Network

The network subsystem is complex and its tuning highly depends on the system use scenario and on external factors such as software clients or hardware components (switches, routers, or gateways) in your network. The Linux kernel aims more at reliability and low latency than low overhead and high throughput. Other settings can mean less security, but better performance.

15.1 Configurable Kernel Socket Buffers

Networking is largely based on the TCP/IP protocol and a socket interface for communication; for more information about TCP/IP, see *Book "Reference", Chapter 13 "Basic Networking"*. The Linux kernel handles data it receives or sends via the socket interface in socket buffers. These kernel socket buffers are tunable.

Important: TCP Autotuning

Since kernel version 2.6.17 full autotuning with 4 MB maximum buffer size exists. This means that manual tuning usually will not improve networking performance considerably. It is often the best not to touch the following variables, or, at least, to check the outcome of tuning efforts carefully.

If you update from an older kernel, it is recommended to remove manual TCP tunings in favor of the autotuning feature.

The special files in the `/proc` file system can modify the size and behavior of kernel socket buffers; for general information about the `/proc` file system, see [Section 2.6, "The /proc File System"](#). Find networking related files in:

```
/proc/sys/net/core
/proc/sys/net/ipv4
/proc/sys/net/ipv6
```

General `net` variables are explained in the kernel documentation ([linux/Documentation/sysctl/net.txt](#)). Special `ipv4` variables are explained in [linux/Documentation/networking/ip-sysctl.txt](#) and [linux/Documentation/networking/ipvs-sysctl.txt](#).

In the `/proc` file system, for example, it is possible to either set the Maximum Socket Receive Buffer and Maximum Socket Send Buffer for all protocols, or both these options for the TCP protocol only (in `ipv4`) and thus overriding the setting for all protocols (in `core`).

/proc/sys/net/ipv4/tcp_moderate_rcvbuf

If /proc/sys/net/ipv4/tcp_moderate_rcvbuf is set to 1, autotuning is active and buffer size is adjusted dynamically.

/proc/sys/net/ipv4/tcp_rmem

The three values setting the minimum, initial, and maximum size of the Memory Receive Buffer per connection. They define the actual memory usage, not only TCP window size.

/proc/sys/net/ipv4/tcp_wmem

The same as tcp_rmem, but for Memory Send Buffer per connection.

/proc/sys/net/core/rmem_max

Set to limit the maximum receive buffer size that applications can request.

/proc/sys/net/core/wmem_max

Set to limit the maximum send buffer size that applications can request.

Via /proc it is possible to disable TCP features that you do not need (all TCP features are switched on by default). For example, check the following files:

/proc/sys/net/ipv4/tcp_timestamps

TCP time stamps are defined in RFC1323.

/proc/sys/net/ipv4/tcp_window_scaling

TCP window scaling is also defined in RFC1323.

/proc/sys/net/ipv4/tcp_sack

Select acknowledgments (SACKS).

Use **sysctl** to read or write variables of the /proc file system. **sysctl** is preferable to **cat** (for reading) and **echo** (for writing), because it also reads settings from /etc/sysctl.conf and, thus, those settings survive reboots reliably. With **sysctl** you can read all variables and their values easily; as root use the following command to list TCP related settings:

```
tux > sudo sysctl -a | grep tcp
```



Note: Side-Effects of Tuning Network Variables

Tuning network variables can affect other system resources such as CPU or memory use.

15.2 Detecting Network Bottlenecks and Analyzing Network Traffic

Before starting with network tuning, it is important to isolate network bottlenecks and network traffic patterns. There are some tools that can help you with detecting those bottlenecks.

The following tools can help analyzing your network traffic: netstat, tcpdump, and wireshark. Wireshark is a network traffic analyzer.

15.3 Netfilter

The Linux firewall and masquerading features are provided by the Netfilter kernel modules. This is a highly configurable rule based framework. If a rule matches a packet, Netfilter accepts or denies it or takes special action (“target”) as defined by rules such as address translation.

There are quite a lot of properties Netfilter can take into account. Thus, the more rules are defined, the longer packet processing may last. Also advanced connection tracking could be rather expensive and, thus, slowing down overall networking.

When the kernel queue becomes full, all new packets are dropped, causing existing connections to fail. The 'fail-open' feature allows a user to temporarily disable the packet inspection and maintain the connectivity under heavy network traffic. For reference, see https://home.regit.org/netfilter-en/using-nfqueue-and-libnetfilter_queue/.

For more information, see the home page of the Netfilter and iptables project, <http://www.netfilter.org>

15.4 Improving the Network Performance with Receive Packet Steering (RPS)

Modern network interface devices can move so many packets that the host can become the limiting factor for achieving maximum performance. To keep up, the system must be able to distribute the work across multiple CPU cores.

Some modern network interfaces can help distribute the work to multiple CPU cores through the implementation of multiple transmission and multiple receive queues in hardware. However, others are only equipped with a single queue and the driver must deal with all incoming packets

in a single, serialized stream. To work around this issue, the operating system must "parallelize" the stream to distribute the work across multiple CPUs. On openSUSE Leap this is done via Receive Packet Steering (RPS). RPS can also be used in virtual environments.

RPS creates a unique hash for each data stream using IP addresses and port numbers. The use of this hash ensures that packets for the same data stream are sent to the same CPU, which helps to increase performance.

RPS is configured per network device receive queue and interface. The configuration file names match the following scheme:

```
/sys/class/net/<device>/queues/<rx-queue>/rps_cpus
```

<device> stands for the network device, such as eth0, eth1. <rx-queue> stands for the receive queue, such as rx-0, rx-1.

If the network interface hardware only supports a single receive queue, only rx-0 will exist. If it supports multiple receive queues, there will be an rx-N directory for each receive queue.

These configuration files contain a comma-delimited list of CPU bitmaps. By default, all bits are set to 0. With this setting RPS is disabled and therefore the CPU that handles the interrupt will also process the packet queue.

To enable RPS and enable specific CPUs to process packets for the receive queue of the interface, set the value of their positions in the bitmap to 1. For example, to enable CPUs 0-3 to process packets for the first receive queue for eth0, set the bit positions 0-3 to 1 in binary: 00001111. This representation then needs to be converted to hex—which results in F in this case. Set this hex value with the following command:

```
tux > sudo echo "f" > /sys/class/net/eth0/queues/rx-0/rps_cpus
```

If you wanted to enable CPUs 8-15:

```
1111 1111 0000 0000 (binary)
15    15    0    0 (decimal)
F     F     0    0 (hex)
```

The command to set the hex value of ff00 would be:

```
tux > sudo echo "ff00" > /sys/class/net/eth0/queues/rx-0/rps_cpus
```

On NUMA machines, best performance can be achieved by configuring RPS to use the CPUs on the same NUMA node as the interrupt for the interface's receive queue.

On non-NUMA machines, all CPUs can be used. If the interrupt rate is very high, excluding the CPU handling the network interface can boost performance. The CPU being used for the network interface can be determined from `/proc/interrupts`. For example:

```
tux > sudo cat /proc/interrupts
          CPU0      CPU1      CPU2      CPU3
...
51: 113915241          0          0          0    Phys-fasteoi  eth0
...
```

In this case, `CPU 0` is the only CPU processing interrupts for `eth0`, since only `CPU0` contains a non-zero value.

On x86 and AMD64/Intel 64 platforms, `irqbalance` can be used to distribute hardware interrupts across CPUs. See `man 1 irqbalance` for more details.

VI Handling System Dumps

- 16 Tracing Tools **156**
- 17 Kexec and Kdump **166**
- 18 Using systemd-coredump to Debug Application Crashes **185**

16 Tracing Tools

openSUSE Leap comes with several tools that help you obtain useful information about your system. You can use the information for various purposes, for example, to debug and find problems in your program, to discover places causing performance drops, or to trace a running process to find out what system resources it uses.



Note: Tracing and Impact on Performance

While a running process is being monitored for system or library calls, the performance of the process is heavily reduced. You are advised to use tracing tools only for the time you need to collect the data.

16.1 Tracing System Calls with `strace`

The `strace` command traces system calls of a process and signals received by the process. `strace` can either run a new command and trace its system calls, or you can attach `strace` to an already running command. Each line of the command's output contains the system call name, followed by its arguments in parentheses and its return value.

To run a new command and start tracing its system calls, enter the command to be monitored as you normally do, and add `strace` at the beginning of the command line:

```
tux > strace ls
execve("/bin/ls", ["ls"], [/* 52 vars */]) = 0
brk(0)                                = 0x618000
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) \
    = 0x7f9848667000
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) \
    = 0x7f9848666000
access("/etc/ld.so.preload", R_OK)     = -1 ENOENT \
(No such file or directory)
open("/etc/ld.so.cache", O_RDONLY)     = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=200411, ...}) = 0
mmap(NULL, 200411, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f9848635000
close(3)                                = 0
open("/lib64/librt.so.1", O_RDONLY)    = 3
[...]
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) \
    = 0x7fd780f79000
```

```

write(1, "Desktop\nDocuments\nbin\ninst-sys\n", 31Desktop
Documents
bin
inst-sys
) = 31
close(1) = 0
munmap(0x7fd780f79000, 4096) = 0
close(2) = 0
exit_group(0) = ?

```

To attach **strace** to an already running process, you need to specify the **-p** with the process ID (**PID**) of the process that you want to monitor:

```

tux > strace -p `pidof cron`
Process 1261 attached
restart_syscall(<... resuming interrupted call ...>) = 0
stat("/etc/localtime", {st_mode=S_IFREG|0644, st_size=2309, ...}) = 0
select(5, [4], NULL, NULL, {0, 0}) = 0 (Timeout)
socket(PF_LOCAL, SOCK_STREAM|SOCK_CLOEXEC|SOCK_NONBLOCK, 0) = 5
connect(5, {sa_family=AF_LOCAL, sun_path="/var/run/nscd/socket"}, 110) = 0
sendto(5, "\2\0\0\0\0\0\0\0\5\0\0\0root\0", 17, MSG_NOSIGNAL, NULL, 0) = 17
poll([{fd=5, events=POLLIN|POLLERR|POLLHUP}], 1, 5000) = 1 ({{fd=5, revents=POLLIN|
POLLHUP}})
read(5, "\2\0\0\0\1\0\0\0\5\0\0\0\2\0\0\0\0\0\0\0\0\0\0\0\0\5\0\0\0\6\0\0\0"... , 36) = 36
read(5, "root\0x\0root\0/root\0/bin/bash\0", 28) = 28
close(5) = 0
rt_sigprocmask(SIG_BLOCK, [CHLD], [], 8) = 0
rt_sigaction(SIGCHLD, NULL, {0x7f772b9ea890, [], SA_RESTORER|SA_RESTART,
0x7f772adf7880}, 8) = 0
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
nanosleep({60, 0}, {0x7fff87d8c580}) = 0
stat("/etc/localtime", {st_mode=S_IFREG|0644, st_size=2309, ...}) = 0
select(5, [4], NULL, NULL, {0, 0}) = 0 (Timeout)
socket(PF_LOCAL, SOCK_STREAM|SOCK_CLOEXEC|SOCK_NONBLOCK, 0) = 5
connect(5, {sa_family=AF_LOCAL, sun_path="/var/run/nscd/socket"}, 110) = 0
sendto(5, "\2\0\0\0\0\0\0\0\5\0\0\0root\0", 17, MSG_NOSIGNAL, NULL, 0) = 17
poll([{fd=5, events=POLLIN|POLLERR|POLLHUP}], 1, 5000) = 1 ({{fd=5, revents=POLLIN|
POLLHUP}})
read(5, "\2\0\0\0\1\0\0\0\5\0\0\0\2\0\0\0\0\0\0\0\0\0\0\0\0\5\0\0\0\6\0\0\0"... , 36) = 36
read(5, "root\0x\0root\0/root\0/bin/bash\0", 28) = 28
close(5)
[...]

```

The **-e** option understands several sub-options and arguments. For example, to trace all attempts to open or write to a particular file, use the following:

```
tux > strace -e trace=open,write ls ~
```

```

open("/etc/ld.so.cache", 0_RDONLY) = 3
open("/lib64/librt.so.1", 0_RDONLY) = 3
open("/lib64/libselinux.so.1", 0_RDONLY) = 3
open("/lib64/libacl.so.1", 0_RDONLY) = 3
open("/lib64/libc.so.6", 0_RDONLY) = 3
open("/lib64/libpthread.so.0", 0_RDONLY) = 3
[...]
open("/usr/lib/locale/cs_CZ.utf8/LC_CTYPE", 0_RDONLY) = 3
open(".", 0_RDONLY|0_NONBLOCK|0_DIRECTORY|0_CLOEXEC) = 3
write(1, "addressbook.db.bak\nbin\ncxoffice\n"... , 311) = 311

```

To trace only network related system calls, use `-e trace=network`:

```

tux > strace -e trace=network -p 26520
Process 26520 attached - interrupt to quit
socket(PF_NETLINK, SOCK_RAW, 0) = 50
bind(50, {sa_family=AF_NETLINK, pid=0, groups=00000000}, 12) = 0
getsockname(50, {sa_family=AF_NETLINK, pid=26520, groups=00000000}, \
[12]) = 0
sendto(50, "\24\0\0\0\26\0\1\3~p\315K\0\0\0\0\0\0", 20, 0,
{sa_family=AF_NETLINK, pid=0, groups=00000000}, 12) = 20
[...]

```

The `-c` calculates the time the kernel spent on each system call:

```

tux > strace -c find /etc -name xorg.conf
/etc/X11/xorg.conf
% time      seconds  usecs/call   calls   errors syscall
-----
 32.38    0.000181     181         1       execve
 22.00    0.000123         0       576     getdents64
 19.50    0.000109         0       917     31 open
 19.14    0.000107         0       888     close
  4.11    0.000023         2        10     mprotect
  0.00    0.000000         0         1     write
[...]
  0.00    0.000000         0         1     getrlimit
  0.00    0.000000         0         1     arch_prctl
  0.00    0.000000         0         3     1 futex
  0.00    0.000000         0         1     set_tid_address
  0.00    0.000000         0         4     fadvise64
  0.00    0.000000         0         1     set_robust_list
-----
100.00    0.000559           3633     33 total

```

To trace all child processes of a process, use `-f`:

```

tux > strace -f systemctl status apache2.service

```

```

execve("/usr/bin/systemctl", ["systemctl", "status", "apache2.service"], \
 0x7ffea44a3318 /* 56 vars */) = 0
brk(NULL)                               = 0x5560f664a000
[...]
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f98c58a5000
mmap(NULL, 4420544, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
 0x7f98c524a000
mprotect(0x7f98c53f4000, 2097152, PROT_NONE) = 0
[...]
[pid 9130] read(0, "\342\227\217 apache2.service - The Apache"... , 8192) = 165
[pid 9130] read(0, "", 8027)           = 0
● apache2.service - The Apache Webserver227\217 apache2.service - Th"... , 193
  Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled; vendor preset:
  disabled)
  Active: inactive (dead)
) = 193
[pid 9130] ioctl(3, SNDCTL_TMR_STOP or TCSETSW, {B38400 opost isig icanon echo ...}) = 0
[pid 9130] exit_group(0)                 = ?
[pid 9130] +++ exited with 0 +++
<... waitid resumed>{si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=9130, \
 si_uid=0, si_status=0, si_utime=0, si_stime=0}, WEXITED, NULL) = 0
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=9130, si_uid=0, \
 si_status=0, si_utime=0, si_stime=0} ---
exit_group(3)                            = ?
+++ exited with 3 +++

```

If you need to analyze the output of **strace** and the output messages are too long to be inspected directly in the console window, use **-o**. In that case, unnecessary messages, such as information about attaching and detaching processes, are suppressed. You can also suppress these messages (normally printed on the standard output) with **-q**. To add time stamps at the beginning of each line with a system call, use **-t**:

```

tux > strace -t -o strace_sleep.txt sleep 1; more strace_sleep.txt
08:44:06 execve("/bin/sleep", ["sleep", "1"], [/* 81 vars */]) = 0
08:44:06 brk(0)                               = 0x606000
08:44:06 mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, \
-1, 0) = 0x7f8e78cc5000
[...]
08:44:06 close(3)                             = 0
08:44:06 nanosleep({1, 0}, NULL)             = 0
08:44:07 close(1)                             = 0
08:44:07 close(2)                             = 0
08:44:07 exit_group(0)                       = ?

```

The behavior and output format of **strace** can be largely controlled. For more information, see the relevant manual page (man 1 **strace**).

16.2 Tracing Library Calls with ltrace

ltrace traces dynamic library calls of a process. It is used in a similar way to **strace**, and most of their parameters have a very similar or identical meaning. By default, **ltrace** uses `/etc/ltrace.conf` or `~/.ltrace.conf` configuration files. You can, however, specify an alternative one with the `-F CONFIG_FILE` option.

In addition to library calls, **ltrace** with the `-S` option can trace system calls as well:

```
tux > ltrace -S -o ltrace_find.txt find /etc -name \
xorg.conf; more ltrace_find.txt
SYS_brk(NULL) = 0x00628000
SYS_mmap(0, 4096, 3, 34, 0xffffffff) = 0x7f1327ea1000
SYS_mmap(0, 4096, 3, 34, 0xffffffff) = 0x7f1327ea0000
[...]
fnmatch("xorg.conf", "xorg.conf", 0) = 0
free(0x0062db80) = <void>
__errno_location() = 0x7f1327e5d698
__ctype_get_mb_cur_max(0x7fff25227af0, 8192, 0x62e020, -1, 0) = 6
__ctype_get_mb_cur_max(0x7fff25227af0, 18, 0x7f1327e5d6f0, 0x7fff25227af0,
0x62e031) = 6
__fprintf_chk(0x7f1327821780, 1, 0x420cf7, 0x7fff25227af0, 0x62e031
<unfinished ...>
SYS_fstat(1, 0x7fff25227230) = 0
SYS_mmap(0, 4096, 3, 34, 0xffffffff) = 0x7f1327e72000
SYS_write(1, "/etc/X11/xorg.conf\n", 19) = 19
[...]
```

You can change the type of traced events with the `-e` option. The following example prints library calls related to `fnmatch` and `strlen` functions:

```
tux > ltrace -e fnmatch,strlen find /etc -name xorg.conf
[...]
fnmatch("xorg.conf", "xorg.conf", 0) = 0
strlen("Xresources") = 10
strlen("Xresources") = 10
strlen("Xresources") = 10
fnmatch("xorg.conf", "Xresources", 0) = 1
strlen("xorg.conf.install") = 17
[...]
```

To display only the symbols included in a specific library, use `-l /path/to/library`:

```
tux > ltrace -l /lib64/librt.so.1 sleep 1
clock_gettime(1, 0x7fff4b5c34d0, 0, 0, 0) = 0
```

```
clock_gettime(1, 0x7fff4b5c34c0, 0xffffffff600180, -1, 0) = 0
+++ exited (status 0) +++
```

You can make the output more readable by indenting each nested call by the specified number of space with the `-n NUM_OF_SPACES`.

16.3 Debugging and Profiling with Valgrind

Valgrind is a set of tools to debug and profile your programs so that they can run both faster and with less errors. Valgrind can detect problems related to memory management and threading, or can also serve as a framework for building new debugging tools. It is well known that this tool can incur high overhead, causing, for example, higher runtimes or changing the normal program behavior under concurrent workloads based on timing.

16.3.1 General Information

The main advantage of Valgrind is that it works with existing compiled executables. You do not need to recompile or modify your programs to use it. Run Valgrind like this:

```
valgrind VALGRIND_OPTIONS your-prog YOUR-PROGRAM-OPTIONS
```

Valgrind consists of several tools, and each provides specific functionality. Information in this section is general and valid regardless of the used tool. The most important configuration option is `--tool`. This option tells Valgrind which tool to run. If you omit this option, `memcheck` is selected by default. For example, to run `find ~ -name .bashrc` with Valgrind's `memcheck` tools, enter the following in the command line:

```
valgrind --tool=memcheck find ~ -name .bashrc
```

A list of standard Valgrind tools with a brief description follows:

memcheck

Detects memory errors. It helps you tune your programs to behave correctly.

cachegrind

Profiles cache prediction. It helps you tune your programs to run faster.

callgrind

Works in a similar way to `cachegrind` but also gathers additional cache-profiling information.

exp-drd

Detects thread errors. It helps you tune your multi-threaded programs to behave correctly.

helgrind

Another thread error detector. Similar to exp-drd but uses different techniques for problem analysis.

massif

A heap profiler. Heap is an area of memory used for dynamic memory allocation. This tool helps you tune your program to use less memory.

lackey

An example tool showing instrumentation basics.

16.3.2 Default Options

Valgrind can read options at start-up. There are three places which Valgrind checks:

1. The file .valgrindrc in the home directory of the user who runs Valgrind.
2. The environment variable \$VALGRIND_OPTS
3. The file .valgrindrc in the current directory where Valgrind is run from.

These resources are parsed exactly in this order, while later given options take precedence over earlier processed options. Options specific to a particular Valgrind tool must be prefixed with the tool name and a colon. For example, if you want cachegrind to always write profile data to the /tmp/cachegrind_PID.log, add the following line to the .valgrindrc file in your home directory:

```
--cachegrind:cachegrind-out-file=/tmp/cachegrind_%p.log
```

16.3.3 How Valgrind Works

Valgrind takes control of your executable before it starts. It reads debugging information from the executable and related shared libraries. The executable's code is redirected to the selected Valgrind tool, and the tool adds its own code to handle its debugging. Then the code is handed back to the Valgrind core and the execution continues.

For example, `memcheck` adds its code, which checks every memory access. As a consequence, the program runs much slower than in the native execution environment.

Valgrind simulates every instruction of your program. Therefore, it not only checks the code of your program, but also all related libraries (including the C library), libraries used for graphical environment, and so on. If you try to detect errors with Valgrind, it also detects errors in associated libraries (like C, X11, or Gtk libraries). Because you probably do not need these errors, Valgrind can selectively, suppress these error messages to suppression files. The `--gen-suppressions=yes` tells Valgrind to report these suppressions which you can copy to a file.

You should supply a real executable (machine code) as a Valgrind argument. If your application is run, for example, from a shell or Perl script, you will by mistake get error reports related to `/bin/sh` (or `/usr/bin/perl`). In such cases, you can use `--trace-children=yes` to work around this issue. However, using the executable itself will avoid any confusion over this issue.

16.3.4 Messages

During its runtime, Valgrind reports messages with detailed errors and important events. The following example explains the messages:

```
tux > valgrind --tool=memcheck find ~ -name .bashrc
[...]
==6558== Conditional jump or move depends on uninitialised value(s)
==6558==    at 0x400AE79: _dl_relocate_object (in /lib64/ld-2.11.1.so)
==6558==    by 0x4003868: dl_main (in /lib64/ld-2.11.1.so)
[...]
==6558== Conditional jump or move depends on uninitialised value(s)
==6558==    at 0x400AE82: _dl_relocate_object (in /lib64/ld-2.11.1.so)
==6558==    by 0x4003868: dl_main (in /lib64/ld-2.11.1.so)
[...]
==6558== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 0 from 0)
==6558== malloc/free: in use at exit: 2,228 bytes in 8 blocks.
==6558== malloc/free: 235 allocs, 227 frees, 489,675 bytes allocated.
==6558== For counts of detected errors, rerun with: -v
==6558== searching for pointers to 8 not-freed blocks.
==6558== checked 122,584 bytes.
==6558==
==6558== LEAK SUMMARY:
==6558==    definitely lost: 0 bytes in 0 blocks.
==6558==    possibly lost: 0 bytes in 0 blocks.
==6558==    still reachable: 2,228 bytes in 8 blocks.
==6558==    suppressed: 0 bytes in 0 blocks.
==6558== Rerun with --leak-check=full to see details of leaked memory.
```

The `==6558==` introduces Valgrind's messages and contains the process ID number (PID). You can easily distinguish Valgrind's messages from the output of the program itself, and decide which messages belong to a particular process.

To make Valgrind's messages more detailed, use `-v` or even `-v -v`.

You can make Valgrind send its messages to three different places:

1. By default, Valgrind sends its messages to the file descriptor 2, which is the standard error output. You can tell Valgrind to send its messages to any other file descriptor with the `--log-fd=FILE_DESCRIPTOR_NUMBER` option.
2. The second and probably more useful way is to send Valgrind's messages to a file with `--log-file=FILENAME`. This option accepts several variables, for example, `%p` gets replaced with the PID of the currently profiled process. This way you can send messages to different files based on their PID. `%q{env_var}` is replaced with the value of the related `env_var` environment variable.

The following example checks for possible memory errors during the Apache Web server restart, while following children processes and writing detailed Valgrind's messages to separate files distinguished by the current process PID:

```
tux > valgrind -v --tool=memcheck --trace-children=yes \  
--log-file=valgrind_pid_%p.log systemctl restart apache2.service
```

This process created 52 log files in the testing system, and took 75 seconds instead of the usual 7 seconds needed to run `sudo systemctl restart apache2.service` without Valgrind, which is approximately 10 times more.

```
tux > ls -l valgrind_pid_*log  
valgrind_pid_11780.log  
valgrind_pid_11782.log  
valgrind_pid_11783.log  
[...]  
valgrind_pid_11860.log  
valgrind_pid_11862.log  
valgrind_pid_11863.log
```

3. You may also prefer to send the Valgrind's messages over the network. You need to specify the `aa.bb.cc.dd` IP address and `port_num` port number of the network socket with the `--log-socket=AA.BB.CC.DD:PORT_NUM` option. If you omit the port number, 1500 will be used.

It is useless to send Valgrind's messages to a network socket if no application is capable of receiving them on the remote machine. That is why `valgrind-listener`, a simple listener, is shipped together with Valgrind. It accepts connections on the specified port and copies everything it receives to the standard output.

16.3.5 Error Messages

Valgrind remembers all error messages, and if it detects a new error, the error is compared against old error messages. This way Valgrind checks for duplicate error messages. In case of a duplicate error, it is recorded but no message is shown. This mechanism prevents you from being overwhelmed by millions of duplicate errors.

The `-v` option will add a summary of all reports (sorted by their total count) to the end of the Valgrind's execution output. Moreover, Valgrind stops collecting errors if it detects either 1000 different errors, or 10 000 000 errors in total. If you want to suppress this limit and wish to see all error messages, use `--error-limit=no`.

Some errors usually cause other ones. Therefore, fix errors in the same order as they appear and re-check the program continuously.

16.4 For More Information

- For a complete list of options related to the described tracing tools, see the corresponding man page (`man 1 strace`, `man 1 ltrace`, and `man 1 valgrind`).
- To describe advanced usage of Valgrind is beyond the scope of this document. It is very well documented, see [Valgrind User Manual \(http://valgrind.org/docs/manual/manual.html\)](http://valgrind.org/docs/manual/manual.html). These pages are indispensable if you need more advanced information on Valgrind or the usage and purpose of its standard tools.

17 Kexec and Kdump

Kexec is a tool to boot to another kernel from the currently running one. You can perform faster system reboots without any hardware initialization. You can also prepare the system to boot to another kernel if the system crashes.

17.1 Introduction

With Kexec, you can replace the running kernel with another one without a hard reboot. The tool is useful for several reasons:

- **Faster system rebooting**
If you need to reboot the system frequently, Kexec can save you significant time.
- **Avoiding unreliable firmware and hardware**
Computer hardware is complex and serious problems may occur during the system start-up. You cannot always replace unreliable hardware immediately. Kexec boots the kernel to a controlled environment with the hardware already initialized. The risk of unsuccessful system start is then minimized.
- **Saving the dump of a crashed kernel**
Kexec preserves the contents of the physical memory. After the *production* kernel fails, the *capture* kernel (an additional kernel running in a reserved memory range) saves the state of the failed kernel. The saved image can help you with the subsequent analysis.
- **Booting without GRUB 2 configuration**
When the system boots a kernel with Kexec, it skips the boot loader stage. The normal booting procedure can fail because of an error in the boot loader configuration. With Kexec, you do not depend on a working boot loader configuration.

17.2 Required Packages

To use Kexec on openSUSE® Leap to speed up reboots or avoid potential hardware problems, make sure that the package `kexec-tools` is installed. It contains a script called **`kexec-boot-loader`**, which reads the boot loader configuration and runs Kexec using the same kernel options as the normal boot loader.

To set up an environment that helps you obtain debug information in case of a kernel crash, make sure that the package `makedumpfile` is installed.

The preferred method of using Kdump in openSUSE Leap is through the YaST Kdump module. To use the YaST module, make sure that the package `yast2-kdump` is installed.

17.3 Kexec Internals

The most important component of Kexec is the `/sbin/kexec` command. You can load a kernel with Kexec in two different ways:

- Load the kernel to the address space of a production kernel for a regular reboot:

```
root # kexec -l KERNEL_IMAGE
```

You can later boot to this kernel with `kexec -e`.

- Load the kernel to a reserved area of memory:

```
root # kexec -p KERNEL_IMAGE
```

This kernel will be booted automatically when the system crashes.

If you want to boot another kernel and preserve the data of the production kernel when the system crashes, you need to reserve a dedicated area of the system memory. The production kernel never loads to this area because it must be always available. It is used for the capture kernel so that the memory pages of the production kernel can be preserved.

To reserve the area, append the option `crashkernel` to the boot command line of the production kernel. To determine the necessary values for `crashkernel`, follow the instructions in [Section 17.4, “Calculating crashkernel Allocation Size”](#).

Note that this is not a parameter of the capture kernel. The capture kernel does not use Kexec. The capture kernel is loaded to the reserved area and waits for the kernel to crash. Then, Kdump tries to invoke the capture kernel because the production kernel is no longer reliable at this stage. This means that even Kdump can fail.

To load the capture kernel, you need to include the kernel boot parameters. Usually, the initial RAM file system is used for booting. You can specify it with `--initrd = FILENAME`. With `--append = CMDLINE`, you append options to the command line of the kernel to boot.

It is required to include the command line of the production kernel. You can simply copy the command line with `--append = "$(cat /proc/cmdline)"` or add more options with `--append = "$(cat /proc/cmdline) more_options"`.

You can always unload the previously loaded kernel. To unload a kernel that was loaded with the `-l` option, use the `kexec -u` command. To unload a crash kernel loaded with the `-p` option, use `kexec -p -u` command.

17.4 Calculating crashkernel Allocation Size

To use Kexec with a capture kernel and to use Kdump in any way, RAM needs to be allocated for the capture kernel. The allocation size depends on the expected hardware configuration of the computer, therefore you need to specify it.

The allocation size also depends on the hardware architecture of your computer. Make sure to follow the procedure intended for your system architecture.

PROCEDURE 17.1: ALLOCATION SIZE ON AMD64/INTEL 64

1. To find out the base value for the computer, run the following command:

```
root # kdumptool calibrate
Total: 49074
Low: 72
High: 180
MinLow: 72
MaxLow: 3085
MinHigh: 0
MaxHigh: 45824
```

All values are given in megabytes.

2. Take note of the values of `Low` and `High`.



Note: Significance of Low and High Values

On AMD64/Intel 64 computers, the High value stands for the memory reservation for all available memory. The Low value stands for the memory reservation in the DMA32 zone, that is, all the memory up to the 4 GB mark.

SIZE_LOW is the amount of memory required by 32-bit-only devices. The kernel will allocate 64M for DMA32 bounce buffers. If your server does not have any 32-bit-only devices, everything should work with the default allocation of 72M for SIZE_LOW. A possible exception to this is on NUMA machines, which may make it appear that more Low memory is needed. The Kdump kernel may be booted with numa=off to make sure normal kernel allocations do not use Low memory.

3. Adapt the High value from the previous step for the number of LUN kernel paths (paths to storage devices) attached to the computer. A sensible value in megabytes can be calculated using this formula:

$$\text{SIZE_HIGH} = \text{RECOMMENDATION} + (\text{LUNs} / 2)$$

The following parameters are used in this formula:

- **SIZE_HIGH.** The resulting value for High.
 - **RECOMMENDATION.** The value recommended by **kdumptool calibrate** for High.
 - **LUNs.** The maximum number of LUN kernel paths that you expect to ever create on the computer. Exclude multipath devices from this number, as these are ignored.
4. If the drivers for your device make many reservations in the DMA32 zone, the Low value also needs to be adjusted. However, there is no simple formula to calculate these. Finding the right size can therefore be a process of trial and error.
For the beginning, use the Low value recommended by **kdumptool calibrate**.
 5. The values now need to be set in the correct location.

If you are changing the kernel command line directly

Append the following kernel option to your boot loader configuration:

```
crashkernel=SIZE_HIGH,high crashkernel=SIZE_LOW,low
```


Replace the placeholders SIZE_HIGH and SIZE_LOW with the appropriate value from the previous steps and append the letter M (for megabytes).

As an example, the following is valid:

```
crashkernel=36M,high crashkernel=72M,low
```

If you are using the YaST GUI:

Set *Kdump Low Memory* to the determined Low value.

Set *Kdump High Memory* to the determined High value.

If you are using the YaST command line interface:

Use the following command:

```
root # yast kdump startup enable alloc_mem=LOW,HIGH
```

Replace LOW with the determined Low value. Replace HIGH with the determined HIGH value.



Tip: Excluding Unused and Inactive CCW Devices on IBM Z

Depending on the number of available devices the calculated amount of memory specified by the crashkernel kernel parameter may not be sufficient. Instead of increasing the value, you may alternatively limit the amount of devices visible to the kernel. This will lower the required amount of memory for the "crashkernel" setting.

1. To ignore devices you can run the cio_ignore tool to generate an appropriate stanza to ignore all devices, except the ones currently active or in use.

```
tux > sudo cio_ignore -u -k  
cio_ignore=all,!da5d,!f500-f502
```

When you run cio_ignore -u -k, the blacklist will become active and replace any existing blacklist immediately. Unused devices are not being purged, so they still appear in the channel subsystem. But adding new channel devices (via CP ATTACH under z/VM or dynamic I/O configuration change in LPAR) will treat them as

blacklisted. To prevent this, preserve the original setting by running `sudo cio_ignore -l` first and reverting to that state after running `cio_ignore -u -k`. As an alternative, add the generated stanza to the regular kernel boot parameters.

2. Now add the `cio_ignore` kernel parameter with the stanza from above to `KDUMP_CMDLINE_APPEND` in `/etc/sysconfig/kdump`, for example:

```
KDUMP_COMMANDLINE_APPEND="cio_ignore=all,!da5d,!f500-f502"
```

3. Activate the setting by restarting `kdump`:

```
systemctl restart kdump.service
```

17.5 Basic Kexec Usage

To use Kexec, ensure the respective service is enabled and running:

- Make sure the Kexec service is loaded at system start:

```
tux > sudo systemctl enable kexec-load.service
```

- Make sure the Kexec service is running:

```
tux > sudo systemctl start kexec-load.service
```

To verify if your Kexec environment works properly, try rebooting into a new Kernel with Kexec. Make sure no users are currently logged in and no important services are running on the system. Then run the following command:

```
systemctl kexec
```

The new kernel previously loaded to the address space of the older kernel rewrites it and takes control immediately. It displays the usual start-up messages. When the new kernel boots, it skips all hardware and firmware checks. Make sure no warning messages appear.



Tip: Using Kexec with the reboot Command

To make **reboot** use Kexec rather than performing a regular reboot, run the following command:

```
ln -s /usr/lib/systemd/system/kexec.target /etc/systemd/system/reboot.target
```

You can revert this at any time by deleting `etc/systemd/system/reboot.target`.

17.6 How to Configure Kexec for Routine Reboots

Kexec is often used for frequent reboots. For example, if it takes a long time to run through the hardware detection routines or if the start-up is not reliable.

Note that firmware and the boot loader are not used when the system reboots with Kexec. Any changes you make to the boot loader configuration will be ignored until the computer performs a hard reboot.

17.7 Basic Kdump Configuration

You can use Kdump to save kernel dumps. If the kernel crashes, it is useful to copy the memory image of the crashed environment to the file system. You can then debug the dump file to find the cause of the kernel crash. This is called “core dump”.

Kdump works similarly to Kexec (see *Chapter 17, Kexec and Kdump*). The capture kernel is executed after the running production kernel crashes. The difference is that Kexec replaces the production kernel with the capture kernel. With Kdump, you still have access to the memory space of the crashed production kernel. You can save the memory snapshot of the crashed kernel in the environment of the Kdump kernel.



Tip: Dumps over Network

In environments with limited local storage, you need to set up kernel dumps over the network. Kdump supports configuring the specified network interface and bringing it up via `initrd`. Both LAN and VLAN interfaces are supported. Specify the network interface and the mode (DHCP or static) either with YaST, or using the `KDUMP_NETCONFIG` option in the `/etc/sysconfig/kdump` file.

! Important: Target File System for Kdump Must Be Mounted During Configuration

When configuring Kdump, you can specify a location to which the dumped images will be saved (default: `/var/crash`). This location must be mounted when configuring Kdump, otherwise the configuration will fail.

17.7.1 Manual Kdump Configuration

Kdump reads its configuration from the `/etc/sysconfig/kdump` file. To make sure that Kdump works on your system, its default configuration is sufficient. To use Kdump with the default settings, follow these steps:

1. Determine the amount of memory needed for Kdump by following the instructions in [Section 17.4, "Calculating crashkernel Allocation Size"](#). Make sure to set the kernel parameter `crashkernel`.
2. Reboot the computer.
3. Enable the Kdump service:

```
root # systemctl enable kdump
```

4. You can edit the options in `/etc/sysconfig/kdump`. Reading the comments will help you understand the meaning of individual options.
5. Execute the init script once with `sudo systemctl start kdump`, or reboot the system.

After configuring Kdump with the default values, check if it works as expected. Make sure that no users are currently logged in and no important services are running on your system. Then follow these steps:

1. Switch to the rescue target with `systemctl isolate rescue.target`
2. Restart the Kdump service:

```
root # systemctl start kdump
```

3. Unmount all the disk file systems except the root file system with:

```
root # umount -a
```

4. Remount the root file system in read-only mode:

```
root # mount -o remount,ro /
```

5. Invoke a “kernel panic” with the `procfs` interface to Magic SysRq keys:

```
root # echo c > /proc/sysrq-trigger
```

! Important: Size of Kernel Dumps

The `KDUMP_KEEP_OLD_DUMPS` option controls the number of preserved kernel dumps (default is 5). Without compression, the size of the dump can take up to the size of the physical RAM memory. Make sure you have sufficient space on the `/var` partition.

The capture kernel boots and the crashed kernel memory snapshot is saved to the file system. The save path is given by the `KDUMP_SAVEDIR` option and it defaults to `/var/crash`. If `KDUMP_IMMEDIATE_REBOOT` is set to `yes`, the system automatically reboots the production kernel. Log in and check that the dump has been created under `/var/crash`.

17.7.1.1 Static IP Configuration for Kdump

In case Kdump is configured to use a static IP configuration from a network device, you need to add the network configuration to the `KDUMP_COMMANDLINE_APPEND` variable in `/etc/sysconfig/kdump`.

EXAMPLE 17.1: KDUMP: EXAMPLE CONFIGURATION USING A STATIC IP SETUP

The following setup has been configured:

- `eth0` has been configured with the static IP address `192.168.1.1/24`
- `eth1` has been configured with the static IP address `10.50.50.100/20`
- The Kdump configuration in `/etc/sysconfig/kdump` looks like:

```
KDUMP_CPUS=1
KDUMP_IMMEDIATE_REBOOT=yes
KDUMP_SAVEDIR=ftp://anonymous@10.50.50.140/crashdump/
KDUMP_KEEP_OLD_DUMPS=5
KDUMP_FREE_DISK_SIZE=64
```

```
KDUMP_VERBOSE=3
KDUMP_DUMPLEVEL=31
KDUMP_DUMPFORMAT=lzo
KDUMP_CONTINUE_ON_ERROR=yes
KDUMP_NETCONFIG=eth1:static
KDUMP_NET_TIMEOUT=30
```

Using this configuration, Kdump fails to reach the network when trying to write the dump to the FTP server. To solve this issue, add the network configuration to `KDUMP_COMMANDLINE_APPEND` in `/etc/sysconfig/kdump`. The general pattern for this looks like the following:

```
KDUMP_COMMANDLINE_APPEND='ip=CLIENT IP:SERVER IP:GATEWAY IP:NETMASK:CLIENT
HOSTNAME:DEVICE:PROTOCOL'
```

For the example configuration this would result in:

```
KDUMP_COMMANDLINE_APPEND='ip=10.50.50.100:10.50.50.140:10.60.48.1:255.255.240.0:dump-
client:eth1:none'
```

17.7.2 YaST Configuration

To configure Kdump with YaST, you need to install the `yast2-kdump` package. Then either start the *Kernel Kdump* module in the *System* category of *YaST Control Center*, or enter `yast2 kdump` in the command line as `root`.

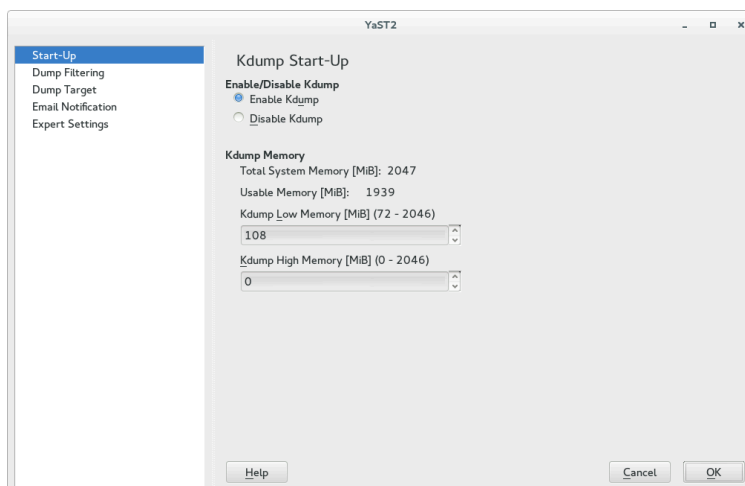


FIGURE 17.1: YAST KDUMP MODULE: START-UP PAGE

In the *Start-Up* window, select *Enable Kdump*.

The values for *Kdump Memory* are automatically generated the first time you open the window. However, that does not mean that they are always sufficient. To set the right values, follow the instructions in [Section 17.4, "Calculating crashkernel Allocation Size"](#).

Important: After Hardware Changes, Set Kdump Memory Values Again

If you have set up Kdump on a computer and later decide to change the amount of RAM or hard disks available to it, YaST will continue to display and use outdated memory values.

To work around this, determine the necessary memory again, as described in [Section 17.4, "Calculating crashkernel Allocation Size"](#). Then set it manually in YaST.

Click *Dump Filtering* in the left pane, and check what pages to include in the dump. You do not need to include the following memory content to be able to debug kernel problems:

- Pages filled with zero
- Cache pages
- User data pages
- Free pages

In the *Dump Target* window, select the type of the dump target and the URL where you want to save the dump. If you selected a network protocol, such as FTP or SSH, you need to enter relevant access information as well.

Tip: Sharing the Dump Directory with Other Applications

It is possible to specify a path for saving Kdump dumps where other applications also save their dumps. When cleaning its old dump files, Kdump will safely ignore other applications' dump files.

Fill the *Email Notification* window information if you want Kdump to inform you about its events via e-mail and confirm your changes with *OK* after fine tuning Kdump in the *Expert Settings* window. Kdump is now configured.

17.7.3 Kdump over SSH

Dump files usually contain sensitive data which should be protected from unauthorized disclosure. To allow transmission of such data over an insecure network, Kdump can save dump files to a remote machine using the SSH protocol.

1. The target host identity must be known to Kdump. This is needed to ensure that sensitive data is never sent to an imposter. When Kdump generates a new `initrd`, it runs `ssh-keygen -F TARGET_HOST` to query the target host's identity. This works only if `TARGET_HOST` public key is already known. An easy way to achieve that is to make an SSH connection to `TARGET_HOST` as `root` on the Kdump host.
2. Kdump must be able to authenticate to the target machine. Only public key authentication is currently available. By default, Kdump will use `root`'s private key, but it is advisable to make a separate key for Kdump. This can be done with `ssh-keygen`:

a.

```
root # ssh-keygen -f ~/.ssh/kdump_key
```

b. Press `Enter` when prompted for passphrase (that is, do not use any passphrase).

c. Open `/etc/sysconfig/kdump` and set `KDUMP_SSH_IDENTITY` to `kdump_key`. You can use full path to the file if it is not placed under `~/.ssh`.

3. Set up the Kdump SSH key to authorize logins to the remote host.

```
root # ssh-copy-id -i ~/.ssh/kdump_key TARGET_HOST
```

4. Set up `KDUMP_SAVEDIR`. There are two options:

Secure File Transfer Protocol (sftp)

SFTP is the preferred method for transmitting files over SSH. The target host must enable the sftp subsystem (SLE default). Example:

```
KDUMP_SAVEDIR=sftp://TARGET_HOST/path/to/dumps
```

Secure Shell Protocol (ssh)

Some other distributions use SSH to run some commands on the target host. openSUSE Leap can also use this method. The Kdump user on the target host must have a login shell that can execute these commands: `mkdir`, `dd` and `mv`. Example:

```
KDUMP_SAVEDIR=ssh://TARGET_HOST/path/to/dumps
```


5. Restart the Kdump service to use the new configuration.

17.8 Analyzing the Crash Dump

After you obtain the dump, it is time to analyze it. There are several options.

The original tool to analyze the dumps is GDB. You can even use it in the latest environments, although it has several disadvantages and limitations:

- GDB was not specifically designed to debug kernel dumps.
- GDB does not support ELF64 binaries on 32-bit platforms.
- GDB does not understand other formats than ELF dumps (it cannot debug compressed dumps).

That is why the **crash** utility was implemented. It analyzes crash dumps and debugs the running system as well. It provides functionality specific to debugging the Linux kernel and is much more suitable for advanced debugging.

If you want to debug the Linux kernel, you need to install its debugging information package in addition. Check if the package is installed on your system with:

```
tux > zypper se kernel | grep debug
```

! Important: Repository for Packages with Debugging Information

If you subscribed your system for online updates, you can find “debuginfo” packages in the [*-Debuginfo-Updates](#) online installation repository relevant for openSUSE Leap 15.2. Use YaST to enable the repository.

To open the captured dump in **crash** on the machine that produced the dump, use a command like this:

```
crash /boot/vmlinuz-2.6.32.8-0.1-default.gz \  
/var/crash/2010-04-23-11\ :17/vmcore
```

The first parameter represents the kernel image. The second parameter is the dump file captured by Kdump. You can find this file under [/var/crash](#) by default.



Tip: Getting Basic Information from a Kernel Crash Dump

openSUSE Leap ships with the utility `kdumpid` (included in a package with the same name) for identifying unknown kernel dumps. It can be used to extract basic information such as architecture and kernel release. It supports `lkcd`, `diskdump`, `Kdump` files and ELF dumps. When called with the `-v` switch it tries to extract additional information such as machine type, kernel banner string and kernel configuration flavor.

17.8.1 Kernel Binary Formats

The Linux kernel comes in Executable and Linkable Format (ELF). This file is usually called `vmlinux` and is directly generated in the compilation process. Not all boot loaders support ELF binaries, especially on the AMD64/Intel 64 architecture. The following solutions exist on different architectures supported by openSUSE® Leap.

17.8.1.1 AMD64/Intel 64

Kernel packages for AMD64/Intel 64 from SUSE contain two kernel files: `vmlinuz` and `vmlinux.gz`.

- `vmlinuz`. This is the file executed by the boot loader. The Linux kernel consists of two parts: the kernel itself (`vmlinux`) and the setup code run by the boot loader. These two parts are linked together to create `vmlinuz` (note the distinction: `z` compared to `x`). In the kernel source tree, the file is called `bzImage`.
- `vmlinux.gz`. This is a compressed ELF image that can be used by `crash` and GDB. The ELF image is never used by the boot loader itself on AMD64/Intel 64. Therefore, only a compressed version is shipped.

17.8.1.2 POWER

The `yaboot` boot loader on POWER also supports loading ELF images, but not compressed ones. In the POWER kernel package, there is an ELF Linux kernel file `vmlinux`. Considering `crash`, this is the easiest architecture.

If you decide to analyze the dump on another machine, you must check both the architecture of the computer and the files necessary for debugging.

You can analyze the dump on another computer only if it runs a Linux system of the same architecture. To check the compatibility, use the command `uname -i` on both computers and compare the outputs.

If you are going to analyze the dump on another computer, you also need the appropriate files from the `kernel` and `kernel debug` packages.

1. Put the kernel dump, the kernel image from `/boot`, and its associated debugging info file from `/usr/lib/debug/boot` into a single empty directory.
2. Additionally, copy the kernel modules from `/lib/modules/$(uname -r)/kernel/` and the associated debug info files from `/usr/lib/debug/lib/modules/$(uname -r)/kernel/` into a subdirectory named `modules`.
3. In the directory with the dump, the kernel image, its debug info file, and the `modules` subdirectory, start the `crash` utility:

```
tux > crash VMLINUX-VERSION vmcore
```

Regardless of the computer on which you analyze the dump, the crash utility will produce output similar to this:

```
tux > crash /boot/vmlinux-5.3.18-8-default.gz \  
/var/crash/2020-04-23-11\ :17/vmcore  
crash 7.2.1  
Copyright (C) 2002-2017 Red Hat, Inc.  
Copyright (C) 2004, 2005, 2006, 2010 IBM Corporation  
Copyright (C) 1999-2006 Hewlett-Packard Co  
Copyright (C) 2005, 2006, 2011, 2012 Fujitsu Limited  
Copyright (C) 2006, 2007 VA Linux Systems Japan K.K.  
Copyright (C) 2005, 2011 NEC Corporation  
Copyright (C) 1999, 2002, 2007 Silicon Graphics, Inc.  
Copyright (C) 1999, 2000, 2001, 2002 Mission Critical Linux, Inc.  
This program is free software, covered by the GNU General Public License,  
and you are welcome to change it and/or distribute copies of it under  
certain conditions. Enter "help copying" to see the conditions.  
This program has absolutely no warranty. Enter "help warranty" for details.  
  
GNU gdb (GDB) 7.6  
Copyright (C) 2013 Free Software Foundation, Inc.  
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
```

```
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-unknown-linux-gnu".
```

```

    KERNEL: /boot/vmlinuz-5.3.18-8-default.gz
DEBUGINFO: /usr/lib/debug/boot/vmlinuz-5.3.18-8-default.debug
DUMPFILE: /var/crash/2020-04-23-11:17/vmcore
    CPUS: 2
    DATE: Thu Apr 23 13:17:01 2020
    UPTIME: 00:10:41
LOAD AVERAGE: 0.01, 0.09, 0.09
    TASKS: 42
NODENAME: eros
RELEASE: 5.3.18-8-default
VERSION: #1 SMP 2020-03-31 14:50:44 +0200
MACHINE: x86_64 (2999 Mhz)
MEMORY: 16 GB
PANIC: "SysRq : Trigger a crashdump"
    PID: 9446
COMMAND: "bash"
    TASK: ffff88003a57c3c0 [THREAD_INFO: ffff880037168000]
    CPU: 1
    STATE: TASK_RUNNING (SYSRQ)
crash>
```

The command output prints first useful data: There were 42 tasks running at the moment of the kernel crash. The cause of the crash was a SysRq trigger invoked by the task with PID 9446. It was a Bash process because the echo that has been used is an internal command of the Bash shell.

The crash utility builds upon GDB and provides many additional commands. If you enter bt without any parameters, the backtrace of the task running at the moment of the crash is printed:

```
crash> bt
PID: 9446 TASK: ffff88003a57c3c0 CPU: 1 COMMAND: "bash"
#0 [ffff880037169db0] crash_kexec at ffffffff80268fd6
#1 [ffff880037169e80] __handle_sysrq at ffffffff803d50ed
#2 [ffff880037169ec0] write_sysrq_trigger at ffffffff802f6fc5
#3 [ffff880037169ed0] proc_reg_write at ffffffff802f068b
#4 [ffff880037169f10] vfs_write at ffffffff802b1aba
#5 [ffff880037169f40] sys_write at ffffffff802b1c1f
#6 [ffff880037169f80] system_call_fastpath at ffffffff8020bfb5
RIP: 00007fa958991f60 RSP: 00007fff61330390 RFLAGS: 00010246
RAX: 0000000000000001 RBX: ffffffff8020bfb5 RCX: 0000000000000001
RDX: 0000000000000002 RSI: 00007fa959284000 RDI: 0000000000000001
RBP: 0000000000000002 R8: 00007fa9592516f0 R9: 00007fa958c209c0
```

```
R10: 00007fa958c209c0 R11: 0000000000000246 R12: 00007fa958c1f780
R13: 00007fa959284000 R14: 0000000000000002 R15: 00000000595569d0
ORIG_RAX: 0000000000000001 CS: 0033 SS: 002b
crash>
```

Now it is clear what happened: The internal `echo` command of Bash shell sent a character to `/proc/sysrq-trigger`. After the corresponding handler recognized this character, it invoked the `crash_kexec()` function. This function called `panic()` and Kdump saved a dump.

In addition to the basic GDB commands and the extended version of `bt`, the crash utility defines other commands related to the structure of the Linux kernel. These commands understand the internal data structures of the Linux kernel and present their contents in a human readable format. For example, you can list the tasks running at the moment of the crash with `ps`. With `sym`, you can list all the kernel symbols with the corresponding addresses, or inquire an individual symbol for its value. With `files`, you can display all the open file descriptors of a process. With `kmem`, you can display details about the kernel memory usage. With `vm`, you can inspect the virtual memory of a process, even at the level of individual page mappings. The list of useful commands is very long and many of these accept a wide range of options.

The commands that we mentioned reflect the functionality of the common Linux commands, such as `ps` and `lsof`. To find out the exact sequence of events with the debugger, you need to know how to use GDB and to have strong debugging skills. Both of these are out of the scope of this document. In addition, you need to understand the Linux kernel. Several useful reference information sources are given at the end of this document.

17.9 Advanced Kdump Configuration

The configuration for Kdump is stored in `/etc/sysconfig/kdump`. You can also use YaST to configure it. Kdump configuration options are available under *System > Kernel Kdump* in *YaST Control Center*. The following Kdump options may be useful for you.

You can change the directory for the kernel dumps with the `KDUMP_SAVEDIR` option. Keep in mind that the size of kernel dumps can be very large. Kdump will refuse to save the dump if the free disk space, subtracted by the estimated dump size, drops below the value specified by the `KDUMP_FREE_DISK_SIZE` option. Note that `KDUMP_SAVEDIR` understands the URL format `PROTOCOL://SPECIFICATION`, where `PROTOCOL` is one of `file`, `ftp`, `sftp`, `nfs` or `cifs`, and `specification` varies for each protocol. For example, to save kernel dump on an FTP server, use the following URL as a template: `ftp://username:password@ftp.example.com:123/var/crash`.

Kernel dumps are usually huge and contain many pages that are not necessary for analysis. With `KDUMP_DUMPLEVEL` option, you can omit such pages. The option understands numeric value between 0 and 31. If you specify `0`, the dump size will be largest. If you specify `31`, it will produce the smallest dump. For a complete table of possible values, see the manual page of `kdump` (`man 7 kdump`).

Sometimes it is very useful to make the size of the kernel dump smaller. For example, if you want to transfer the dump over the network, or if you need to save some disk space in the dump directory. This can be done with `KDUMP_DUMPFORMAT` set to `compressed`. The `crash` utility supports dynamic decompression of the compressed dumps.



Important: Changes to the Kdump Configuration File

You always need to execute `systemctl restart kdump` after you make manual changes to `/etc/sysconfig/kdump`. Otherwise, these changes will take effect next time you reboot the system.



17.10 For More Information

There is no single comprehensive reference to Kexec and Kdump usage. However, there are helpful resources that deal with certain aspects:

- For the Kexec utility usage, see the manual page of `kexec` (`man 8 kexec`).
- You can find general information about Kexec at <http://www.ibm.com/developerworks/linux/library/l-kexec.html> . Might be slightly outdated.
- For more details on Kdump specific to openSUSE Leap, see <http://ftp.suse.com/pub/people/tiwai/kdump-training/kdump-training.pdf> .
- An in-depth description of Kdump internals can be found at <http://lse.sourceforge.net/kdump/documentation/ols2oo5-kdump-paper.pdf> .

For more details on `crash` dump analysis and debugging tools, use the following resources:

- In addition to the info page of GDB (`info gdb`), there are printable guides at <http://sourceware.org/gdb/documentation/> .
- The crash utility also features a comprehensive online help. Use `help COMMAND` to display the online help for `command`.

- If you have the necessary Perl skills, you can use Alicia to make the debugging easier. This Perl-based front-end to the crash utility can be found at <http://alicia.sourceforge.net/>  .
- If you prefer to use Python instead, you should install Pykdump. This package helps you control GDB through Python scripts and can be downloaded from <http://sf.net/projects/pykdump>  .
- A very comprehensive overview of the Linux kernel internals is given in *Understanding the Linux Kernel* by Daniel P. Bovet and Marco Cesati (ISBN 978-0-596-00565-8).

18 Using systemd-coredump to Debug Application Crashes

`systemd-coredump` collects and displays kernel core dumps, for analyzing application crashes. When a process crashes (or all processes belonging to an application), its default is to log the core dump to the `systemd` journal, including a backtrace if possible, and to store the core dump in a file in `/var/lib/systemd/coredump`. You also have the option to examine the dump file with other tools such as `gdb` or `crash` (see [Section 17.8, “Analyzing the Crash Dump”](#)). There is an option to not store core dumps, but to log only to the journal, which may be useful to minimize the collection and storage of sensitive information.

18.1 Use and Configuration

`systemd-coredump` is enabled and ready to run by default. The default configuration is in `/etc/systemd/coredump.conf`:

```
[Coredump]
#Storage=external
#Compress=yes
#ProcessSizeMax=2G
#ExternalSizeMax=2G
#JournalSizeMax=767M
#MaxUse=
#KeepFree=
```

The following example shows how to use Vim for simple testing, by creating a segfault to generate journal entries and a core dump.

PROCEDURE 18.1: CREATING A CORE DUMP WITH VIM

1. Enable the `debuginfo-pool` and `debuginfo-update` repositories
2. Install `vim-debuginfo`
3. Launch `vim testfile` and type a few characters
4. Get the PID and generate a segfault:

```
tux > ps ax | grep vim
```



```
2345 pts/3    S+      0:00 vim testfile
```

```
root # kill -s SIGSEGV 2345
```

Vim will emit error messages:

```
Vim: Caught deadly signal SEGV
Vim: Finished.
Segmentation fault (core dumped)
```

5. List your core dumps, then examine them:

```
root # coredumpctl
TIME                               PID  UID  GID SIG PRESENT EXE
Wed 2019-11-12 11:56:47 PST 2345 1000 100 11 *      /bin/vim

root # coredumpctl info
PID: 2345 (vim)
UID: 0 (root)
GID: 0 (root)
Signal: 11 (SEGV)
Timestamp: Wed 2019-11-12 11:58:05 PST
Command Line: vim testfile
Executable: /bin/vim
Control Group: /user.slice/user-1000.slice/session-1.scope
  Unit: session-1.scope
  Slice: user-1000.slice
  Session: 1
  Owner UID: 1000 (tux)
  Boot ID: b5c251b86ab34674a2222cef102c0c88
  Machine ID: b43c44a64696799b985cafd95dc1b698
  Hostname: linux-uoch
  Coredump: /var/lib/systemd/coredump/core.vim.0.b5c251b86ab34674a2222cef102
  Message: Process 2345 (vim) of user 0 dumped core.

Stack trace of thread 2345:
#0  0x00007f21dd87e2a7 kill (libc.so.6)
#1  0x0000000000050cb35 may_core_dump (vim)
#2  0x00007f21ddbfc70 __restore_rt (libpthread.so.0)
#3  0x00007f21dd92ea33 __select (libc.so.6)
#4  0x0000000000050b4e3 RealWaitForChar (vim)
#5  0x0000000000050b86b mch_inchar (vim)

[...]
```

When you have multiple core dumps, `coredumpctl info` displays all of them. Filter them by `PID`, `COMM` (command), or `EXE` (full path to the executable), for example, all core dumps for Vim:

```
root # coredumpctl info /bin/vim
```

See a single core dump by `PID`:

```
root # coredumpctl info 2345
```

Output the selected core to `gdb`:

```
root # coredumpctl gdb 2345
```

The asterisk in the `PRESENT` column indicates that a stored core dump is present. If the field is empty there is no stored core dump, and `coredumpctl` retrieves crash information from the journal. You may control this behavior in `/etc/systemd/coredump.conf` with the `Storage` option:

- `Storage=none`, core dumps are logged in the journal, but not stored. This is useful to minimize collecting and storing sensitive information, for example for General Data Protection Regulation (GDPR) compliance.
- `Storage=external`, cores are stored in `/var/lib/systemd/coredump`
- `Storage=journal`, cores are stored in the `systemd` journal

A new instance of `systemd-coredump` is invoked for every core dump, so configuration changes are applied with the next core dump, and there is no need to restart any services.

Core dumps are not preserved after a system restart. You may save them permanently with `coredumpctl`. The following example filters by the `PID` and stores the core in `vim.dump`:

```
root # coredumpctl -o vim.dump dump 2345
```

See `man systemd-coredump`, `man coredumpctl`, and `man coredump.conf` for complete command and option listings.

VII Synchronized Clocks with Precision Time Protocol

19 Precision Time Protocol **189**

19 Precision Time Protocol

For network environments, it is vital to keep the computer and other devices' clocks synchronized and accurate. There are several solutions to achieve this, for example the widely used Network Time Protocol (NTP) described in *Book "Reference", Chapter 18 "Time Synchronization with NTP"*.

The Precision Time Protocol (PTP) is a protocol capable of sub-microsecond accuracy, which is better than what NTP achieves. PTP support is divided between the kernel and user space. The kernel in openSUSE Leap includes support for PTP clocks, which are provided by network drivers.

19.1 Introduction to PTP

The clocks managed by PTP follow a master-slave hierarchy. The slaves are synchronized to their masters. The hierarchy is updated by the *best master clock* (BMC) algorithm, which runs on every clock. The clock with only one port can be either master or slave. Such a clock is called an *ordinary clock* (OC). A clock with multiple ports can be master on one port and slave on another. Such a clock is called a *boundary clock* (BC). The top-level master is called the *grandmaster clock*. The grandmaster clock can be synchronized with a Global Positioning System (GPS). This way disparate networks can be synchronized with a high degree of accuracy.

The hardware support is the main advantage of PTP. It is supported by various network switches and network interface controllers (NIC). While it is possible to use non-PTP enabled hardware within the network, having network components between all PTP clocks PTP hardware enabled achieves the best possible accuracy.

19.1.1 PTP Linux Implementation

On openSUSE Leap, the implementation of PTP is provided by the `linuxptp` package. Install it with `zypper install linuxptp`. It includes the `ptp4l` and `phc2sys` programs for clock synchronization. `ptp4l` implements the PTP boundary clock and ordinary clock. When hardware time stamping is enabled, `ptp4l` synchronizes the PTP hardware clock to the master clock. With software time stamping, it synchronizes the system clock to the master clock. `phc2sys` is needed only with hardware time stamping to synchronize the system clock to the PTP hardware clock on the network interface card (NIC).

19.2 Using PTP

19.2.1 Network Driver and Hardware Support

PTP requires that the used kernel network driver supports either software or hardware time stamping. Moreover, the NIC must support time stamping in the physical hardware. You can verify the driver and NIC time stamping capabilities with **ethtool**:

```
tux > sudo ethtool -T eth0
Time stamping parameters for eth0:
Capabilities:
hardware-transmit      (SOF_TIMESTAMPING_TX_HARDWARE)
software-transmit      (SOF_TIMESTAMPING_TX_SOFTWARE)
hardware-receive       (SOF_TIMESTAMPING_RX_HARDWARE)
software-receive       (SOF_TIMESTAMPING_RX_SOFTWARE)
software-system-clock  (SOF_TIMESTAMPING_SOFTWARE)
hardware-raw-clock     (SOF_TIMESTAMPING_RAW_HARDWARE)
PTP Hardware Clock: 0
Hardware Transmit Timestamp Modes:
off                    (HWTSTAMP_TX_OFF)
on                     (HWTSTAMP_TX_ON)
Hardware Receive Filter Modes:
none                   (HWTSTAMP_FILTER_NONE)
all                    (HWTSTAMP_FILTER_ALL)
```

Software time stamping requires the following parameters:

```
SOF_TIMESTAMPING_SOFTWARE
SOF_TIMESTAMPING_TX_SOFTWARE
SOF_TIMESTAMPING_RX_SOFTWARE
```

Hardware time stamping requires the following parameters:

```
SOF_TIMESTAMPING_RAW_HARDWARE
SOF_TIMESTAMPING_TX_HARDWARE
SOF_TIMESTAMPING_RX_HARDWARE
```

19.2.2 Using `ptp4l`

`ptp4l` uses hardware time stamping by default. As `root`, you need to specify the network interface capable of hardware time stamping with the `-i` option. The `-m` tells `ptp4l` to print its output to the standard output instead of the system's logging facility:

```
tux > sudo ptp4l -m -i eth0
selected eth0 as PTP clock
port 1: INITIALIZING to LISTENING on INITIALIZE
port 0: INITIALIZING to LISTENING on INITIALIZE
port 1: new foreign master 00a152.ffff.0b334d-1
selected best master clock 00a152.ffff.0b334d
port 1: LISTENING to UNCALIBRATED on RS_SLAVE
master offset -25937 s0 freq +0 path delay      12340
master offset -27887 s0 freq +0 path delay      14232
master offset -38802 s0 freq +0 path delay      13847
master offset -36205 s1 freq +0 path delay      10623
master offset  -6975 s2 freq -30575 path delay   10286
port 1: UNCALIBRATED to SLAVE on MASTER_CLOCK_SELECTED
master offset  -4284 s2 freq -30135 path delay    9892
```

The `master offset` value represents the measured offset from the master (in nanoseconds).

The `s0`, `s1`, `s2` indicators show the different states of the clock servo: `s0` is unlocked, `s1` is clock step, and `s2` is locked. If the servo is in the locked state (`s2`), the clock will not be stepped (only slowly adjusted) if the `pi_offset_const` option is set to a negative value in the configuration file (see [man 8 ptp4l](#) for more information).

The `freq` value represents the frequency adjustment of the clock (in parts per billion, ppb).

The `path delay` value represents the estimated delay of the synchronization messages sent from the master (in nanoseconds).

Port 0 is a Unix domain socket used for local PTP management. Port 1 is the `eth0` interface.

`INITIALIZING`, `LISTENING`, `UNCALIBRATED` and `SLAVE` are examples of port states which change on `INITIALIZE`, `RS_SLAVE`, and `MASTER_CLOCK_SELECTED` events. When the port state changes from `UNCALIBRATED` to `SLAVE`, the computer has successfully synchronized with a PTP master clock.

You can enable software time stamping with the `-S` option.

```
tux > sudo ptp4l -m -S -i eth3
```

You can also run `ptp4l` as a service:

```
tux > sudo systemctl start ptp4l
```

In this case, **ptp4l** reads its options from the `/etc/sysconfig/ptp4l` file. By default, this file tells **ptp4l** to read the configuration options from `/etc/ptp4l.conf`. For more information on **ptp4l** options and the configuration file settings, see [man 8 ptp4l](#).

To enable the **ptp4l** service permanently, run the following:

```
tux > sudo systemctl enable ptp4l
```

To disable it, run

```
tux > sudo systemctl disable ptp4l
```

19.2.3 **ptp4l** Configuration File

ptp4l can read its configuration from an optional configuration file. As no configuration file is used by default, you need to specify it with `-f`.

```
tux > sudo ptp4l -f /etc/ptp4l.conf
```

The configuration file is divided into sections. The global section (indicated as `[global]`) sets the program options, clock options and default port options. Other sections are port specific, and they override the default port options. The name of the section is the name of the configured port—for example, `[eth0]`. An empty port section can be used to replace the command line option.

```
[global]
verbose          1
time_stamping    software
[eth0]
```

The example configuration file is an equivalent of the following command's options:

```
tux > sudo ptp4l -i eth0 -m -S
```

For a complete list of **ptp4l** configuration options, see [man 8 ptp4l](#).

19.2.4 Delay Measurement

ptp4l measures time delay in two different ways: *peer-to-peer* (P2P) or *end-to-end* (E2E).

P2P

This method is specified with `-P`.

It reacts to changes in the network environment faster and is more accurate in measuring the delay. It is only used in networks where each port exchanges PTP messages with one other port. P2P needs to be supported by all hardware on the communication path.

E2E

This method is specified with `-E`. This is the default.

Automatic method selection

This method is specified with `-A`. The automatic option starts `ptp4l` in E2E mode, and changes to P2P mode if a peer delay request is received.



Important: Common Measurement Method

All clocks on a single PTP communication path must use the same method to measure the time delay. A warning will be printed if either a peer delay request is received on a port using the E2E mechanism, or an E2E delay request is received on a port using the P2P mechanism.

19.2.5 PTP Management Client: `pmc`

You can use the `pmc` client to obtain more detailed information about `ptp4l`. It reads from the standard input—or from the command line—actions specified by name and management ID. Then it sends the actions over the selected transport, and prints any received replies. There are three actions supported: `GET` retrieves the specified information, `SET` updates the specified information, and `CMD` (or `COMMAND`) initiates the specified event.

By default, the management commands are addressed to all ports. The `TARGET` command can be used to select a particular clock and port for the subsequent messages. For a complete list of management IDs, run `pmc help`.

```
tux > sudo pmc -u -b 0 'GET TIME_STATUS_NP'
sending: GET TIME_STATUS_NP
90f2ca.ffff.20d7e9-0 seq 0 RESPONSE MANAGMENT TIME_STATUS_NP
  master_offset           283
  ingress_time            1361569379345936841
  cumulativeScaledRateOffset +1.000000000
  scaledLastGmPhaseChange 0
  gmTimeBaseIndicator     0
  lastGmPhaseChange       0x0000'0000000000000000.0000
```


gmPresent	true
gmIdentity	00b058.feef.0b448a

The `-b` option specifies the boundary hops value in sent messages. Setting it to zero limits the boundary to the local `ptp4l` instance. Increasing the value will retrieve the messages also from PTP nodes that are further from the local instance. The returned information may include:

stepsRemoved

The number of communication nodes to the grandmaster clock.

offsetFromMaster, master_offset

The last measured offset of the clock from the master clock (nanoseconds).

meanPathDelay

The estimated delay of the synchronization messages sent from the master clock (nanoseconds).

gmPresent

If `true`, the PTP clock is synchronized to the master clock; the local clock is not the grandmaster clock.

gmIdentity

This is the grandmaster's identity.

For a complete list of `pmc` command line options, see `man 8 pmc`.

19.3 Synchronizing the Clocks with `phc2sys`

Use `phc2sys` to synchronize the system clock to the PTP hardware clock (PHC) on the network card. The system clock is considered a *slave*, while the network card a *master*. PHC itself is synchronized with `ptp4l` (see [Section 19.2, "Using PTP"](#)). Use `-s` to specify the master clock by device or network interface. Use `-w` to wait until `ptp4l` is in a synchronized state.

```
tux > sudo phc2sys -s eth0 -w
```

PTP operates in *International Atomic Time* (TAI), while the system clock uses *Coordinated Universal Time* (UTC). If you do not specify `-w` to wait for `ptp4l` synchronization, you can specify the offset in seconds between TAI and UTC with `-0`:

```
tux > sudo phc2sys -s eth0 -0 -35
```

You can run **phc2sys** as a service as well:

```
tux > sudo systemctl start phc2sys
```

In this case, **phc2sys** reads its options from the `/etc/sysconfig/phc2sys` file. For more information on **phc2sys** options, see `man 8 phc2sys`.

To enable the **phc2sys** service permanently, run the following:

```
tux > sudo systemctl enable phc2sys
```

To disable it, run

```
tux > sudo systemctl disable phc2sys
```

19.3.1 Verifying Time Synchronization

When PTP time synchronization is working properly and hardware time stamping is used, **ptp4l** and **phc2sys** output messages with time offsets and frequency adjustments periodically to the system log.

An example of the **ptp4l** output:

```
ptp4l[351.358]: selected /dev/ptp0 as PTP clock
ptp4l[352.361]: port 1: INITIALIZING to LISTENING on INITIALIZE
ptp4l[352.361]: port 0: INITIALIZING to LISTENING on INITIALIZE
ptp4l[353.210]: port 1: new foreign master 00a069.eefe.0b442d-1
ptp4l[357.214]: selected best master clock 00a069.eefe.0b662d
ptp4l[357.214]: port 1: LISTENING to UNCALIBRATED on RS_SLAVE
ptp4l[359.224]: master offset      3304 s0 freq      +0 path delay      9202
ptp4l[360.224]: master offset      3708 s1 freq     -28492 path delay      9202
ptp4l[361.224]: master offset     -3145 s2 freq     -32637 path delay      9202
ptp4l[361.224]: port 1: UNCALIBRATED to SLAVE on MASTER_CLOCK_SELECTED
ptp4l[362.223]: master offset     -145 s2 freq     -30580 path delay      9202
ptp4l[363.223]: master offset      1043 s2 freq     -28436 path delay      8972
[...]
ptp4l[371.235]: master offset        285 s2 freq     -28511 path delay      9199
ptp4l[372.235]: master offset       -78 s2 freq     -28788 path delay      9204
```

An example of the **phc2sys** output:

```
phc2sys[616.617]: Waiting for ptp4l...
phc2sys[628.628]: phc offset      66341 s0 freq      +0 delay      2729
phc2sys[629.628]: phc offset      64668 s1 freq     -37690 delay      2726
```

```
[...]
phc2sys[646.630]: phc offset      -333 s2 freq  -37426 delay  2747
phc2sys[646.630]: phc offset      194 s2 freq  -36999 delay  2749
```

ptp4l normally writes messages very frequently. You can reduce the frequency with the `summary_interval` directive. Its value is an exponent of the 2^N expression. For example, to reduce the output to every 1024 (which is equal to 2^{10}) seconds, add the following line to the `/etc/ptp4l.conf` file:

```
summary_interval 10
```

You can also reduce the frequency of the `phc2sys` command's updates with the `-u SUMMARY-UPDATES` option.

19.4 Examples of Configurations

This section includes several examples of `ptp4l` configuration. The examples are not full configuration files but rather a minimal list of changes to be made to the specific files. The string `ethX` stands for the actual network interface name in your setup.

EXAMPLE 19.1: SLAVE CLOCK USING SOFTWARE TIME STAMPING

```
/etc/sysconfig/ptp4l:
```

```
OPTIONS="-f /etc/ptp4l.conf -i ethX"
```

No changes made to the distribution `/etc/ptp4l.conf`.

EXAMPLE 19.2: SLAVE CLOCK USING HARDWARE TIME STAMPING

```
/etc/sysconfig/ptp4l:
```

```
OPTIONS="-f /etc/ptp4l.conf -i ethX"
```

```
/etc/sysconfig/phc2sys:
```

```
OPTIONS="-s ethX -w"
```

No changes made to the distribution `/etc/ptp4l.conf`.

EXAMPLE 19.3: MASTER CLOCK USING HARDWARE TIME STAMPING

```
/etc/sysconfig/ptp4l:
```

```
OPTIONS="-f /etc/ptp4l.conf -i ethX"
```

/etc/sysconfig/phc2sys:

```
OPTIONS="-s CLOCK_REALTIME -c ethX -w"
```

/etc/ptp4l.conf:

```
priority1 127
```

EXAMPLE 19.4: MASTER CLOCK USING SOFTWARE TIME STAMPING (NOT GENERALLY RECOMMENDED)

/etc/sysconfig/ptp4l:

```
OPTIONS="-f /etc/ptp4l.conf -i ethX"
```

/etc/ptp4l.conf:

```
priority1 127
```

19.5 PTP and NTP

NTP and PTP time synchronization tools can coexist, synchronizing time from one to another in both directions.

19.5.1 NTP to PTP Synchronization

When chronyd is used to synchronize the local system clock, you can configure the ptp4l to be the grandmaster clock distributing the time from the local system clock via PTP. Include the priority1 option in /etc/ptp4l.conf:

```
[global]
priority1 127
[eth0]
```

Then run ptp4l:

```
tux > sudo ptp4l -f /etc/ptp4l.conf
```

When hardware time stamping is used, you need to synchronize the PTP hardware clock to the system clock with phc2sys:

```
tux > sudo phc2sys -c eth0 -s CLOCK_REALTIME -w
```

19.5.2 Configuring PTP-NTP Bridge

If a highly accurate PTP grandmaster is available in a network without switches or routers with PTP support, a computer may operate as a PTP slave and a stratum-1 NTP server. Such a computer needs to have two or more network interfaces, and be close to the grandmaster or have a direct connection to it. This will ensure highly accurate synchronization in the network.

Configure the `ptp4l` and `phc2sys` programs to use one network interface to synchronize the system clock using PTP. Then configure `chronyd` to provide the system time using the other interface:

```
bindaddress 192.0.131.47
hwtimestamp eth1
local stratum 1
```



Note: NTP and DHCP

When the DHCP client command `dhclient` receives a list of NTP servers, it adds them to NTP configuration by default. To prevent this behavior, set

```
NETCONFIG_NTP_POLICY=""
```

in the `/etc/sysconfig/network/config` file.

A GNU Licenses

This appendix contains the GNU Free Documentation License version 1.2.

GNU Free Documentation License

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary

formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.  
Permission is granted to copy, distribute  
and/or modify this document  
under the terms of the GNU Free  
Documentation License, Version 1.2  
or any later version published by the Free  
Software Foundation;  
with no Invariant Sections, no Front-Cover  
Texts, and no Back-Cover Texts.  
A copy of the license is included in the  
section entitled "GNU  
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being LIST  
THEIR TITLES, with the  
Front-Cover Texts being LIST, and with the  
Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.



Virtualization Guide

openSUSE Leap 15.2



Virtualization Guide

openSUSE Leap 15.2

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

Publication Date: July 06, 2020

SUSE LLC

1800 South Novell Place

Provo, UT 84606

USA

<https://documentation.suse.com> ↗

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

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

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

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

Contents

About This Manual xv

I	INTRODUCTION	1
1	Virtualization Technology	2
1.1	Overview	2
1.2	Virtualization Capabilities	3
1.3	Virtualization Benefits	3
1.4	Virtualization Modes	4
1.5	I/O Virtualization	4
2	Introduction to Xen Virtualization	7
2.1	Basic Components	7
2.2	Xen Virtualization Architecture	8
3	Introduction to KVM Virtualization	10
3.1	Basic Components	10
3.2	KVM Virtualization Architecture	10
4	Introduction to Linux Containers	12
5	Virtualization Tools	13
5.1	Virtualization Console Tools	13
5.2	Virtualization GUI Tools	14
6	Installation of Virtualization Components	18
6.1	Installing KVM	18

6.2	Installing Xen	18
6.3	Installing Containers	19
6.4	Patterns	19
6.5	Installing UEFI Support	20
6.6	Enable Support for Nested Virtualization in KVM	21
II	MANAGING VIRTUAL MACHINES WITH <code>libvirt</code>	23
7	Starting and Stopping <code>libvirtd</code>	24
8	Guest Installation	26
8.1	GUI-Based Guest Installation	26
8.2	Installing from the Command Line with <code>virt-install</code>	28
8.3	Advanced Guest Installation Scenarios	31
	Including Add-on Products in the Installation	31
9	Basic VM Guest Management	32
9.1	Listing VM Guests	32
	Listing VM Guests with Virtual Machine Manager	32
	• Listing VM Guests with <code>virsh</code>	33
9.2	Accessing the VM Guest via Console	33
	Opening a Graphical Console	33
	• Opening a Serial Console	35
9.3	Changing a VM Guest's State: Start, Stop, Pause	36
	Changing a VM Guest's State with Virtual Machine Manager	37
	• Changing a VM Guest's State with <code>virsh</code>	38
9.4	Saving and Restoring the State of a VM Guest	39
	Saving/Restoring with Virtual Machine Manager	40
	• Saving and Restoring with <code>virsh</code>	40
9.5	Creating and Managing Snapshots	40
	Terminology	41
	• Creating and Managing Snapshots with Virtual Machine Manager	42
	• Creating and Managing Snapshots with <code>virsh</code>	43

- 9.6 Deleting a VM Guest 46
 - Deleting a VM Guest with Virtual Machine Manager 46 • Deleting a VM Guest with **virsh** 46
- 9.7 Migrating VM Guests 47
 - Migration Requirements 47 • Migrating with Virtual Machine Manager 48 • Migrating with **virsh** 49 • Step-by-Step Example 51
- 9.8 Monitoring 54
 - Monitoring with Virtual Machine Manager 54 • Monitoring with **virt-top** 55 • Monitoring with **kvm_stat** 56
- 10 Connecting and Authorizing 58**
- 10.1 Authentication 58
 - libvirtd Authentication 59 • VNC Authentication 63
- 10.2 Connecting to a VM Host Server 67
 - “system” Access for Non-Privileged Users 68 • Managing Connections with Virtual Machine Manager 69
- 10.3 Configuring Remote Connections 70
 - Remote Tunnel over SSH (qemu+ssh or xen+ssh) 71 • Remote TLS/SSL Connection with x509 Certificate (qemu+tls or xen+tls) 71
- 11 Managing Storage 79**
- 11.1 Managing Storage with Virtual Machine Manager 81
 - Adding a Storage Pool 82 • Managing Storage Pools 85
- 11.2 Managing Storage with **virsh** 87
 - Listing Pools and Volumes 87 • Starting, Stopping and Deleting Pools 89 • Adding Volumes to a Storage Pool 90 • Deleting Volumes from a Storage Pool 91 • Attaching Volumes to a VM Guest 91 • Detaching Volumes from a VM Guest 92
- 11.3 Locking Disk Files and Block Devices with **virtlockd** 93
 - Enable Locking 93 • Configure Locking 94
- 11.4 Online Resizing of Guest Block Devices 95

- 11.5 Sharing Directories between Host and Guests (File System Pass-Through) 96
- 11.6 Using RADOS Block Devices with `libvirt` 97
- 12 Managing Networks 98**
 - 12.1 Network Bridge 98
 - Managing Network Bridges with YaST 98 • Managing Network Bridges from the Command Line 99 • Using VLAN Interfaces 101
 - 12.2 Virtual Networks 102
 - Managing Virtual Networks with Virtual Machine Manager 103 • Managing Virtual Networks with `virsh` 107
- 13 Configuring Virtual Machines with Virtual Machine Manager 113**
 - 13.1 Machine Setup 114
 - Overview 114 • Performance 115 • Processor 116 • Memory 117 • Boot Options 118
 - 13.2 Storage 119
 - 13.3 Controllers 120
 - 13.4 Networking 121
 - 13.5 Input Devices 123
 - 13.6 Video 124
 - 13.7 USB Redirectors 126
 - 13.8 Miscellaneous 127
 - 13.9 Adding a CD/DVD-ROM Device with Virtual Machine Manager 127
 - 13.10 Adding a Floppy Device with Virtual Machine Manager 128
 - 13.11 Ejecting and Changing Floppy or CD/DVD-ROM Media with Virtual Machine Manager 129

- 13.12 Assigning a Host PCI Device to a VM Guest 130
 - Adding a PCI Device with Virtual Machine Manager 130
- 13.13 Assigning a Host USB Device to a VM Guest 131
 - Adding a USB Device with Virtual Machine Manager 132
- 14 Configuring Virtual Machines with `virsh` 133**
 - 14.1 Editing the VM Configuration 133
 - 14.2 Changing the Machine Type 134
 - 14.3 Configuring Hypervisor Features 135
 - 14.4 Configuring CPU Allocation 136
 - 14.5 Changing Boot Options 137
 - Changing Boot Order 137 • Using Direct Kernel Boot 138
 - 14.6 Configuring Memory Allocation 138
 - 14.7 Adding a PCI Device 139
 - 14.8 Adding a USB Device 141
 - 14.9 Adding SR-IOV Devices 142
 - Requirements 143 • Loading and Configuring the SR-IOV Host Drivers 143 • Adding a VF Network Device to a VM Guest 146 • Dynamic Allocation of VFs from a Pool 149
 - 14.10 Configuring Storage Devices 151
 - 14.11 Configuring Controller Devices 152
 - 14.12 Configuring Video Devices 153
 - Changing the Amount of Allocated VRAM 153 • Changing the State of 2D/3D Acceleration 154
 - 14.13 Configuring Network Devices 154
 - Scaling Network Performance with Multiqueue virtio-net 155
 - 14.14 Using Macvtap to Share VM Host Server Network Interfaces 155
 - 14.15 Disabling a Memory Balloon Device 157

14.16	Configuring Multiple Monitors (Dual Head)	157
III	HYPERVISOR-INDEPENDENT FEATURES	159
15	Disk Cache Modes	160
15.1	Disk Interface Cache Modes	160
15.2	Description of Cache Modes	160
15.3	Data Integrity Implications of Cache Modes	162
15.4	Performance Implications of Cache Modes	162
15.5	Effect of Cache Modes on Live Migration	163
16	VM Guest Clock Settings	164
16.1	KVM: Using <code>kvm_clock</code>	164
	Other Timekeeping Methods	165
16.2	Xen Virtual Machine Clock Settings	165
17	<code>libguestfs</code>	166
17.1	VM Guest Manipulation Overview	166
	VM Guest Manipulation Risk	166
	<code>libguestfs</code> Design	167
17.2	Package Installation	167
17.3	Guestfs Tools	168
	Modifying Virtual Machines	168
	Supported File Systems and Disk Images	168
	<code>virt-rescue</code>	169
	<code>virt-resize</code>	169
	Other <code>virt-*</code> Tools	171
	<code>guestfish</code>	173
	Converting a Physical Machine into a KVM Guest	174
17.4	Troubleshooting	176
	Btrfs-related Problems	176
	Environment	177
	<code>libguestfs-test-tool</code>	177
17.5	External References	177

IV	MANAGING VIRTUAL MACHINES WITH XEN	178
18	Setting Up a Virtual Machine Host	179
18.1	Best Practices and Suggestions	179
18.2	Managing Dom0 Memory	180
	Setting Dom0 Memory Allocation	181
18.3	Network Card in Fully Virtualized Guests	181
18.4	Starting the Virtual Machine Host	182
18.5	PCI Pass-Through	183
	Configuring the Hypervisor for PCI Pass-Through	184
	Assigning PCI Devices to VM Guest Systems	185
	VGA Pass-Through	186
	Troubleshooting	186
	For More Information	187
18.6	USB Pass-Through	187
	Identify the USB Device	187
	Emulated USB Device	188
	Paravirtualized PVUSB	188
19	Virtual Networking	190
19.1	Network Devices for Guest Systems	190
19.2	Host-Based Routing in Xen	192
19.3	Creating a Masqueraded Network Setup	195
19.4	Special Configurations	197
	Bandwidth Throttling in Virtual Networks	197
	Monitoring the Network Traffic	198
20	Managing a Virtualization Environment	199
20.1	XL—Xen Management Tool	199
	Guest Domain Configuration File	200
20.2	Automatic Start of Guest Domains	201
20.3	Event Actions	201
20.4	Time Stamp Counter	202

20.5	Saving Virtual Machines	203
20.6	Restoring Virtual Machines	203
20.7	Virtual Machine States	204
21	Block Devices in Xen	205
21.1	Mapping Physical Storage to Virtual Disks	205
21.2	Mapping Network Storage to Virtual Disk	206
21.3	File-Backed Virtual Disks and Loopback Devices	206
21.4	Resizing Block Devices	207
21.5	Scripts for Managing Advanced Storage Scenarios	208
22	Virtualization: Configuration Options and Settings	209
22.1	Virtual CD Readers	209
	Virtual CD Readers on Paravirtual Machines	209
	Virtual CD Readers on Fully Virtual Machines	209
	Adding Virtual CD Readers	210
	Removing Virtual CD Readers	211
22.2	Remote Access Methods	211
22.3	VNC Viewer	211
	Assigning VNC Viewer Port Numbers to Virtual Machines	212
	Using SDL instead of a VNC Viewer	213
22.4	Virtual Keyboards	213
22.5	Dedicating CPU Resources	214
	Dom0	214
	VM Guests	215
22.6	HVM Features	215
	Specify Boot Device on Boot	216
	Changing CPUIDs for Guests	216
	Increasing the Number of PCI-IRQs	217
23	Administrative Tasks	218
23.1	The Boot Loader Program	218

23.2	Sparse Image Files and Disk Space	219
23.3	Migrating Xen VM Guest Systems	220
	Preparing Block Devices for Migrations	221
	Migrating VM Guest Systems	222
23.4	Monitoring Xen	222
	Monitor Xen with xentop	222
	Additional Tools	223
23.5	Providing Host Information for VM Guest Systems	224
24	XenStore: Configuration Database Shared between Domains	226
24.1	Introduction	226
24.2	File System Interface	226
	XenStore Commands	227
	/vm	227
	/local/domain/<domid>	229
25	Xen as a High-Availability Virtualization Host	231
25.1	Xen HA with Remote Storage	231
25.2	Xen HA with Local Storage	232
25.3	Xen HA and Private Bridges	233
V	MANAGING VIRTUAL MACHINES WITH QEMU	234
26	QEMU Overview	235
27	Setting Up a KVM VM Host Server	236
27.1	CPU Support for Virtualization	236
27.2	Required Software	236
27.3	KVM Host-Specific Features	238
	Using the Host Storage with virtio-scsi	238
	Accelerated Networking with vhost-net	239
	Scaling Network Performance with Multiqueue virtio-net	240
	VFIO: Secure Direct Access to Devices	241
	VirtFS: Sharing Directories between Host and Guests	243
	KSM: Sharing Memory Pages between Guests	244

28 Guest Installation 246

- 28.1 Basic Installation with **qemu-system-ARCH** 246
- 28.2 Managing Disk Images with **qemu-img** 247
 - General Information on qemu-img Invocation 248 • Creating, Converting and Checking Disk Images 249 • Managing Snapshots of Virtual Machines with qemu-img 255 • Manipulate Disk Images Effectively 257

29 Running Virtual Machines with **qemu-system-ARCH** 262

- 29.1 Basic **qemu-system-ARCH** Invocation 262
- 29.2 General **qemu-system-ARCH** Options 262
 - Basic Virtual Hardware 263 • Storing and Reading Configuration of Virtual Devices 265 • Guest Real-Time Clock 266
- 29.3 Using Devices in QEMU 266
 - Block Devices 267 • Graphic Devices and Display Options 272 • USB Devices 274 • Character Devices 276
- 29.4 Networking in QEMU 278
 - Defining a Network Interface Card 278 • User-Mode Networking 279 • Bridged Networking 281
- 29.5 Viewing a VM Guest with VNC 284
 - Secure VNC Connections 286

30 Virtual Machine Administration Using QEMU Monitor 289

- 30.1 Accessing Monitor Console 289
- 30.2 Getting Information about the Guest System 290
- 30.3 Changing VNC Password 292
- 30.4 Managing Devices 293
- 30.5 Controlling Keyboard and Mouse 294
- 30.6 Changing Available Memory 294

- 30.7 Dumping Virtual Machine Memory 295
- 30.8 Managing Virtual Machine Snapshots 296
- 30.9 Suspending and Resuming Virtual Machine Execution 297
- 30.10 Live Migration 297
- 30.11 QMP - QEMU Machine Protocol 299
 - Access QMP via Standard Input/Output 299 • Access QMP via Telnet 300 • Access QMP via Unix Socket 301 • Access QMP via libvirt's **virsh** Command 302

VI MANAGING VIRTUAL MACHINES WITH LXC 303

31 Linux Containers 304

- 31.1 Setting Up LXC Distribution Containers 304
- 31.2 Setting Up LXC Application Containers 307
- 31.3 Securing a Container Using AppArmor 308
- 31.4 Differences between the libvirt LXC Driver and LXC 308
- 31.5 Sharing Namespaces across Containers 310
- 31.6 For More Information 310

32 Migration from LXC to libvirt-lxc 311

- 32.1 Host Migration 311
- 32.2 Container Migration 312
- 32.3 Starting the Container 313

Glossary 314

A Appendix 325

B XM, XL Toolstacks and Libvirt framework 326

B.1 Xen Toolstacks 326

Upgrading from xend/xm to xl/libxl 326 • XL design 327 • Checklist before Upgrade 327

B.2 Import Xen Domain Configuration into libvirt 328

B.3 Differences between the **xm** and **xl** Applications 330

Notation Conventions 330 • New Global Options 330 • Unchanged Options 331 • Removed Options 335 • Changed Options 338 • New Options 352

B.4 External links 353

B.5 Saving a Xen Guest Configuration in an **xm** Compatible Format 354

C GNU Licenses 355

C.1 GNU Free Documentation License 355

About This Manual

This manual offers an introduction to setting up and managing virtualization with KVM (Kernel-based Virtual Machine), Xen, and Linux Containers (LXC) on openSUSE Leap. The first part introduces the different virtualization solutions by describing their requirements, their installations and SUSE's support status. The second part deals with managing VM Guests and VM Host Servers with `libvirt`. The following parts describe various administration tasks and practices and the last three parts deal with hypervisor-specific topics.

1 Available Documentation



Note: Online Documentation and Latest Updates

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

The following documentation is available for this product:

Book “Start-Up”

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

Book “Reference”

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

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.

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

2 Giving Feedback


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

Bug Reports

Report issues with the documentation at <https://bugzilla.opensuse.org/> . To simplify this process, you can use the *Report Documentation Bug* links next to headlines in the HTML version of this document. These preselect the right product and category in Bugzilla and add a link to the current section. You can start typing your bug report right away. A Bugzilla account is required.

Contributions

To contribute to this documentation, use the *Edit Source* links next to headlines in the HTML version of this document. They take you to the source code on GitHub, where you can open a pull request. A GitHub account is required.

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

Mail

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

Help

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

3 Documentation Conventions

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

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

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

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

```
tux > command
```

- Notices



Warning: Warning Notice

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



Important: Important Notice

Important information you should be aware of before proceeding.



Note: Note Notice

Additional information, for example about differences in software versions.



Tip: Tip Notice

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

I Introduction

- 1 Virtualization Technology **2**
- 2 Introduction to Xen Virtualization **7**
- 3 Introduction to KVM Virtualization **10**
- 4 Introduction to Linux Containers **12**
- 5 Virtualization Tools **13**
- 6 Installation of Virtualization Components **18**

1 Virtualization Technology

Virtualization is a technology that provides a way for a machine (Host) to run another operating system (guest virtual machines) on top of the host operating system.

1.1 Overview

openSUSE Leap includes the latest open source virtualization technologies, Xen and KVM. With these hypervisors, openSUSE Leap can be used to provision, de-provision, install, monitor and manage multiple virtual machines (VM Guests) on a single physical system (for more information see *Hypervisor*).

Out of the box, openSUSE Leap can create virtual machines running both modified, highly tuned, paravirtualized operating systems and fully virtualized unmodified operating systems. Full virtualization allows the guest OS to run unmodified and requires the presence of AMD64/Intel 64 processors which support either Intel* Virtualization Technology (Intel VT) or AMD* Virtualization (AMD-V).

The primary component of the operating system that enables virtualization is a hypervisor (or virtual machine manager), which is a layer of software that runs directly on server hardware. It controls platform resources, sharing them among multiple VM Guests and their operating systems by presenting virtualized hardware interfaces to each VM Guest.

openSUSE is a Linux server operating system that offers two types of hypervisors: Xen and KVM. Both hypervisors support virtualization on the AMD64/Intel 64 architecture. For the POWER architecture, KVM is supported. Both Xen and KVM support full virtualization mode. In addition, Xen supports paravirtualized mode, and you can run both paravirtualized and fully-virtualized guests on the same host.

openSUSE Leap with Xen or KVM acts as a virtualization host server (*VHS*) that supports VM Guests with its own guest operating systems. The SUSE VM Guest architecture consists of a hypervisor and management components that constitute the VHS, which runs many application-hosting VM Guests.

In Xen, the management components run in a privileged VM Guest often called *Dom0*. In KVM, where the Linux kernel acts as the hypervisor, the management components run directly on the VHS.

1.2 Virtualization Capabilities

Virtualization design provides many capabilities to your organization. Virtualization of operating systems is used in many computing areas:

- **Server consolidation:** Many servers can be replaced by one big physical server, so hardware is consolidated, and Guest Operating Systems are converted to virtual machine. It provides the ability to run legacy software on new hardware.
- **Isolation:** guest operating system can be fully isolated from the Host running it. So if the virtual machine is corrupted, the Host system is not harmed.
- **Migration:** A process to move a running virtual machine to another physical machine. Live migration is an extended feature that allows this move without disconnection of the client or the application.
- **Disaster recovery:** *Virtualized* guests are less dependent on the hardware, and the Host server provides snapshot features to be able to restore a known running system without any corruption.
- **Dynamic load balancing:** A migration feature that brings a simple way to load-balance your service across your infrastructure.

1.3 Virtualization Benefits

Virtualization brings a lot of advantages while providing the same service as a hardware server. First, it reduces the cost of your infrastructure. Servers are mainly used to provide a service to a customer, and a virtualized operating system can provide the same service, with:

- **Less hardware:** You can run several operating system on one host, so all hardware maintenance will be reduced.
- **Less power/cooling:** Less hardware means you do not need to invest more in electric power, backup power, and cooling if you need more service.
- **Save space:** Your data center space will be saved because you do not need more hardware servers (less servers than service running).

- **Less management:** Using a VM Guest simplifies the administration of your infrastructure.
- **Agility and productivity:** Virtualization provides *migration* capabilities, *live migration* and *snapshots*. These features reduce downtime, and bring an easy way to move your service from one place to another without any service interruption.

1.4 Virtualization Modes

Guest operating systems are hosted on virtual machines in either full virtualization (FV) mode or paravirtual (PV) mode. Each virtualization mode has advantages and disadvantages.

- Full virtualization mode lets virtual machines run unmodified operating systems, such as Windows* Server 2003. It can use either Binary Translation or *hardware-assisted* virtualization technology, such as AMD* Virtualization or Intel* Virtualization Technology. Using hardware assistance allows for better performance on processors that support it.
- To be able to run under paravirtual mode, guest operating systems usually need to be modified for the virtualization environment. However, operating systems running in paravirtual mode have better performance than those running under full virtualization. Operating systems currently modified to run in paravirtual mode are called *paravirtualized operating systems* and include openSUSE Leap and NetWare® 6.5 SP8.

1.5 I/O Virtualization

VM Guests not only share CPU and memory resources of the host system, but also the I/O subsystem. Because software I/O virtualization techniques deliver less performance than bare metal, hardware solutions that deliver almost “native” performance have been developed recently. openSUSE Leap supports the following I/O virtualization techniques:

Full Virtualization

Fully Virtualized (FV) drivers emulate widely supported real devices, which can be used with an existing driver in the VM Guest. The guest is also called *Hardware Virtual Machine* (HVM). Since the physical device on the VM Host Server may differ from the emulated one, the hypervisor needs to process all I/O operations before handing them over to the physical device. Therefore all I/O operations need to traverse two software layers, a process that not only significantly impacts I/O performance, but also consumes CPU time.

Paravirtualization

Paravirtualization (PV) allows direct communication between the hypervisor and the VM Guest. With less overhead involved, performance is much better than with full virtualization. However, paravirtualization requires either the guest operating system to be modified to support the paravirtualization API or paravirtualized drivers.

PVHVM

This type of virtualization enhances HVM (see *Full Virtualization*) with paravirtualized (PV) drivers, and PV interrupt and timer handling.

VFIO

VFIO stands for *Virtual Function I/O* and is a new user-level driver framework for Linux. It replaces the traditional KVM PCI Pass-Through device assignment. The VFIO driver exposes direct device access to user space in a secure memory (*IOMMU*) protected environment. With VFIO, a VM Guest can directly access hardware devices on the VM Host Server (pass-through), avoiding performance issues caused by emulation in performance critical paths. This method does not allow to share devices—each device can only be assigned to a single VM Guest. VFIO needs to be supported by the VM Host Server CPU, chipset and the BIOS/EFI.

Compared to the legacy KVM PCI device assignment, VFIO has the following advantages:

- Resource access is compatible with secure boot.
- Device is isolated and its memory access protected.
- Offers a user space device driver with more flexible device ownership model.
- Is independent of KVM technology, and not bound to x86 architecture only.

As of openSUSE 42.2, the USB and PCI Pass-through methods of device assignment are considered deprecated and were superseded by the VFIO model.

SR-IOV

The latest I/O virtualization technique, Single Root I/O Virtualization SR-IOV combines the benefits of the aforementioned techniques—performance and the ability to share a device with several VM Guests. SR-IOV requires special I/O devices, that are capable of replicating resources so they appear as multiple separate devices. Each such “pseudo” device can be directly used by a single guest. However, for network cards for example the number of concurrent queues that can be used is limited, potentially reducing performance for the VM Guest compared to paravirtualized drivers. On the VM Host Server, SR-IOV must be supported by the I/O device, the CPU and chipset, the BIOS/EFI and the hypervisor—for setup instructions see *Section 13.12, “Assigning a Host PCI Device to a VM Guest”*.



Important: Requirements for VFIO and SR-IOV

To be able to use the VFIO and SR-IOV features, the VM Host Server needs to fulfill the following requirements:

- IOMMU needs to be enabled in the BIOS/EFI.
- For Intel CPUs, the kernel parameter `intel_iommu=on` needs to be provided on the kernel command line. For more information, see *Book "Reference", Chapter 12 "The Boot Loader GRUB 2", Section 12.3.3.2 "Kernel Parameters Tab"*.
- The VFIO infrastructure needs to be available. This can be achieved by loading the kernel module `vfio_pci`. For more information, see *Book "Reference", Chapter 10 "The systemd Daemon", Section 10.6.4 "Loading Kernel Modules"*.

2 Introduction to Xen Virtualization

This chapter introduces and explains the components and technologies you need to understand to set up and manage a Xen-based virtualization environment.

2.1 Basic Components

The basic components of a Xen-based virtualization environment are the *Xen hypervisor*, the *Dom0*, any number of other *VM Guests*, and the tools, commands, and configuration files that let you manage virtualization. Collectively, the physical computer running all these components is called a *VM Host Server* because together these components form a platform for hosting virtual machines.

The Xen Hypervisor

The Xen hypervisor, sometimes simply called a virtual machine monitor, is an open source software program that coordinates the low-level interaction between virtual machines and physical hardware.

The Dom0

The virtual machine host environment, also called *Dom0* or controlling domain, is composed of several components, such as:

- openSUSE Leap provides a graphical and a command line environment to manage the virtual machine host components and its virtual machines.



Note

The term “Dom0” refers to a special domain that provides the management environment. This may be run either in graphical or in command line mode.

- The xl tool stack based on the xenlight library (libxl). Use it to manage Xen guest domains.
- QEMU—an open source software that emulates a full computer system, including a processor and various peripherals. It provides the ability to host operating systems in both full virtualization or paravirtualization mode.

Xen-Based Virtual Machines

A Xen-based virtual machine, also called a *VM Guest* or *DomU*, consists of the following components:

- At least one virtual disk that contains a bootable operating system. The virtual disk can be based on a file, partition, volume, or other type of block device.
- A configuration file for each guest domain. It is a text file following the syntax described in the manual page `man 5 xl.conf`.
- Several network devices, connected to the virtual network provided by the controlling domain.

Management Tools, Commands, and Configuration Files

There is a combination of GUI tools, commands, and configuration files to help you manage and customize your virtualization environment.

2.2 Xen Virtualization Architecture

The following graphic depicts a virtual machine host with four virtual machines. The Xen hypervisor is shown as running directly on the physical hardware platform. Note that the controlling domain is also a virtual machine, although it has several additional management tasks compared to all the other virtual machines.

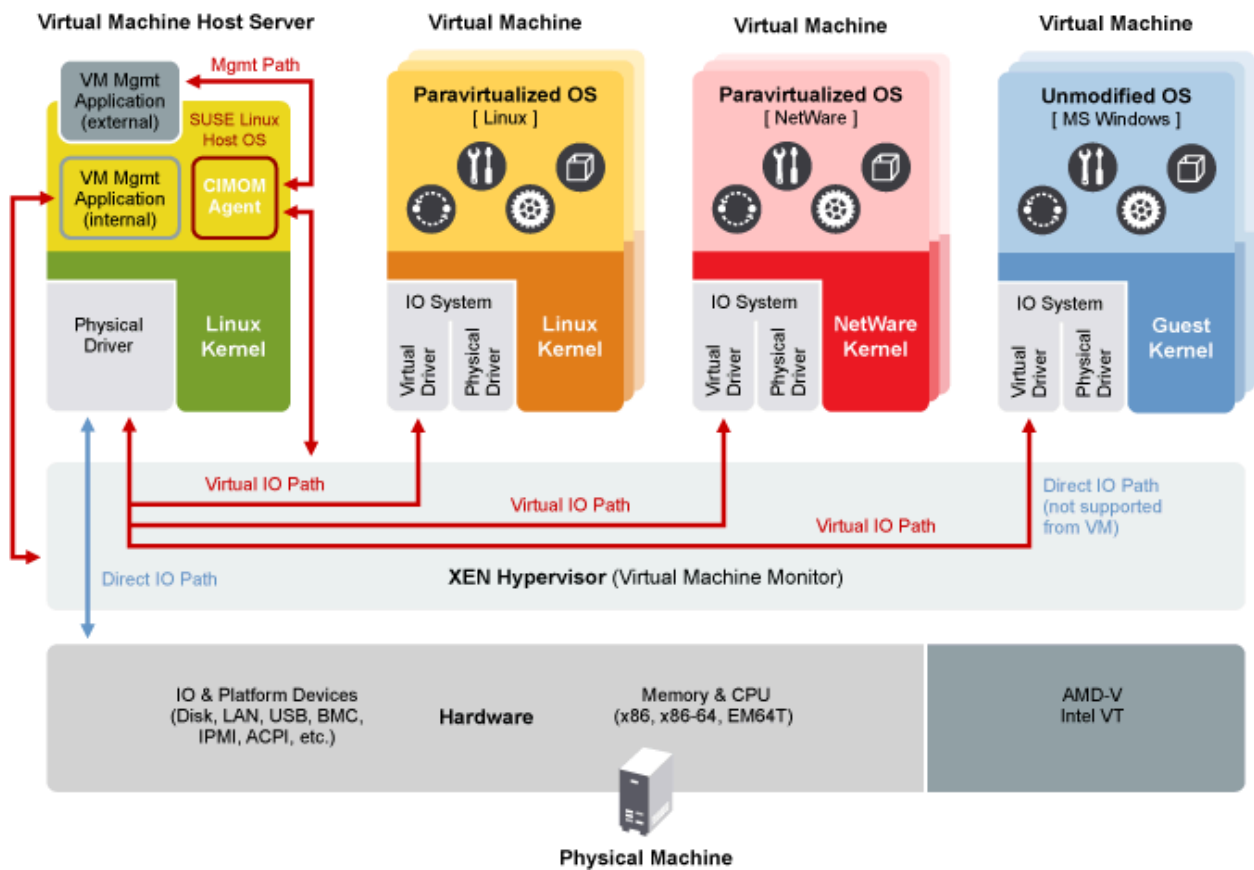


FIGURE 2.1: XEN VIRTUALIZATION ARCHITECTURE

On the left, the virtual machine host's Dom0 is shown running the openSUSE Leap operating system. The two virtual machines shown in the middle are running paravirtualized operating systems. The virtual machine on the right shows a fully virtual machine running an unmodified operating system, such as the latest version of Microsoft Windows/Server.

3 Introduction to KVM Virtualization

3.1 Basic Components

KVM is a full virtualization solution for the AMD64/Intel 64 architecture supporting hardware virtualization.

VM Guests (virtual machines), virtual storage, and virtual networks can be managed with QEMU tools directly, or with the `libvirt`-based stack. The QEMU tools include `qemu-system-ARCH`, the QEMU monitor, `qemu-img`, and `qemu-ndb`. A `libvirt`-based stack includes `libvirt` itself, along with `libvirt`-based applications such as `virsh`, `virt-manager`, `virt-install`, and `virt-viewer`.

3.2 KVM Virtualization Architecture

This full virtualization solution consists of two main components:

- A set of kernel modules (`kvm.ko`, `kvm-intel.ko`, and `kvm-amd.ko`) that provides the core virtualization infrastructure and processor-specific drivers.
- A user space program (`qemu-system-ARCH`) that provides emulation for virtual devices and control mechanisms to manage VM Guests (virtual machines).

The term KVM more properly refers to the kernel level virtualization functionality, but is in practice more commonly used to refer to the user space component.

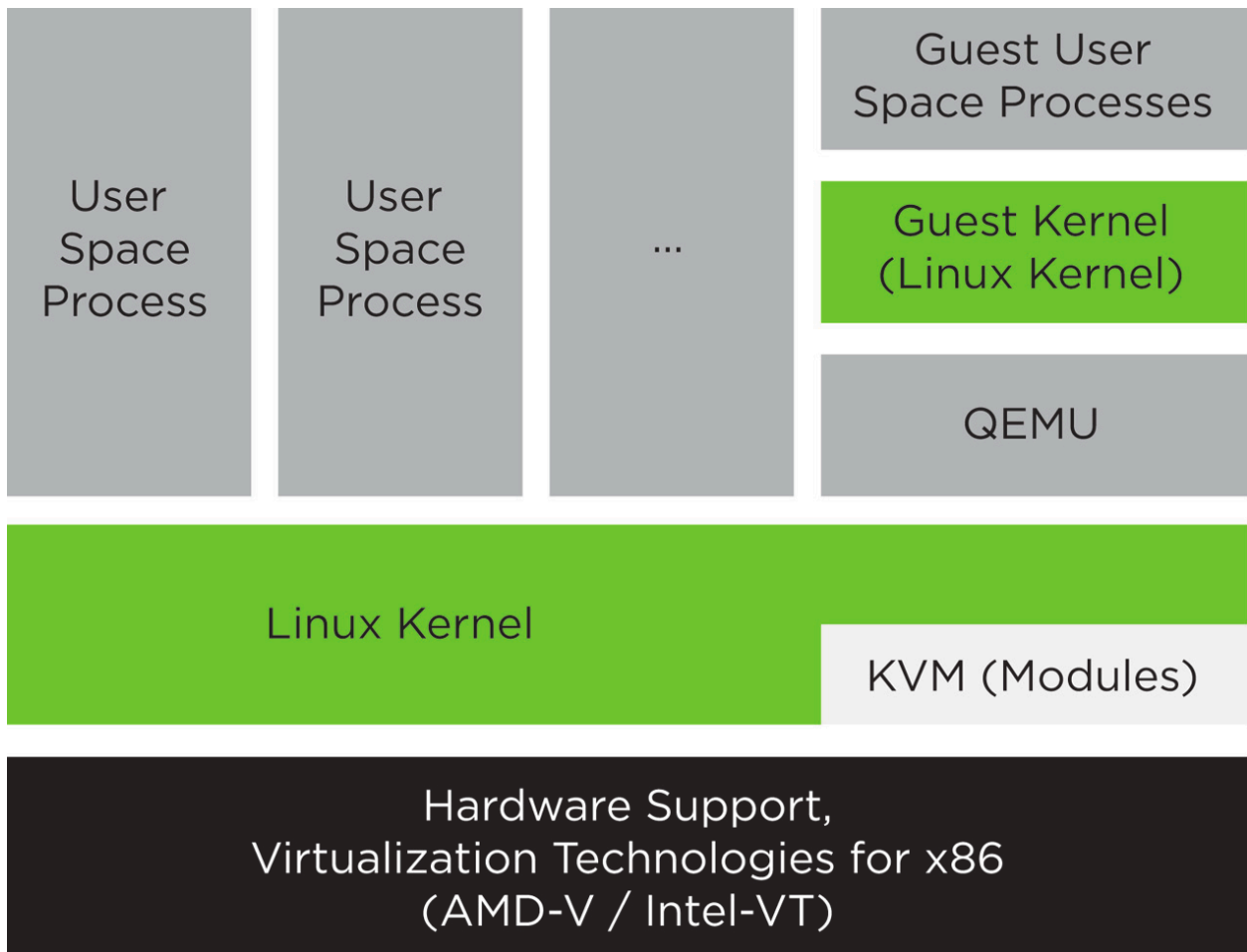


FIGURE 3.1: KVM VIRTUALIZATION ARCHITECTURE



Note: Hyper-V Emulation Support

QEMU can provide certain Hyper-V hypercalls for Windows* guests to partly emulate a Hyper-V environment. This can be used to achieve better behavior for Windows* guests that are Hyper-V enabled.

4 Introduction to Linux Containers

Linux containers are a lightweight virtualization method to run multiple virtual units (“containers”) simultaneously on a single host. This is similar to the *chroot* environment. Containers are isolated with kernel Control Groups (*cgroups*) and kernel Namespaces.

Containers provide virtualization at the operating system level where the *kernel* controls the isolated containers. This is unlike full virtualization solutions like Xen or KVM where the *processor* simulates a complete hardware environment and controls virtual machines.

Conceptually, containers can be seen as an improved *chroot* technique. The difference is that a *chroot* environment separates only the file system, whereas containers go further and provide resource management and control via *cgroups*.

BENEFITS OF CONTAINERS

- Isolating applications and operating systems through containers.
- Providing nearly native performance as container manages allocation of resources in real-time.
- Controlling network interfaces and applying resources inside containers through *cgroups*.

LIMITATIONS OF CONTAINERS

- All containers run inside the host system's kernel and not with a different kernel.
- Only allows Linux “guest” operating systems.
- Security depends on the host system. Container is not secure. If you need a secure system, you can confine it using an AppArmor or SELinux profile.

5 Virtualization Tools

libvirt is a library that provides a common API for managing popular virtualization solutions, among them KVM, LXC, and Xen. The library provides a normalized management API for these virtualization solutions, allowing a stable, cross-hypervisor interface for higher-level management tools. The library also provides APIs for management of virtual networks and storage on the VM Host Server. The configuration of each VM Guest is stored in an XML file.

With libvirt you can also manage your VM Guests remotely. It supports TLS encryption, x509 certificates and authentication with SASL. This enables managing VM Host Servers centrally from a single workstation, alleviating the need to access each VM Host Server individually.

Using the libvirt-based tools is the recommended way of managing VM Guests. Interoperability between libvirt and libvirt-based applications has been tested and is an essential part of SUSE's support stance.

5.1 Virtualization Console Tools

The following libvirt-based tools for the command line are available on openSUSE Leap:

virsh (Package: libvirt-client)

A command line tool to manage VM Guests with similar functionality as the Virtual Machine Manager. Allows you to change a VM Guest's status (start, stop, pause, etc.), to set up new guests and devices, or to edit existing configurations. virsh is also useful to script VM Guest management operations.

virsh takes the first argument as a command and further arguments as options to this command:

```
virsh [-c URI] COMMAND DOMAIN-ID [OPTIONS]
```

Like zypper, virsh can also be called without a command. In this case it starts a shell waiting for your commands. This mode is useful when having to run subsequent commands:

```
~> virsh -c qemu+ssh://wilber@mercury.example.com/system
```

```
Enter passphrase for key '/home/wilber/.ssh/id_rsa':
Welcome to virsh, the virtualization interactive terminal.

Type: 'help' for help with commands
      'quit' to quit

virsh # hostname
mercury.example.com
```

virt-install (Package: virt-install)

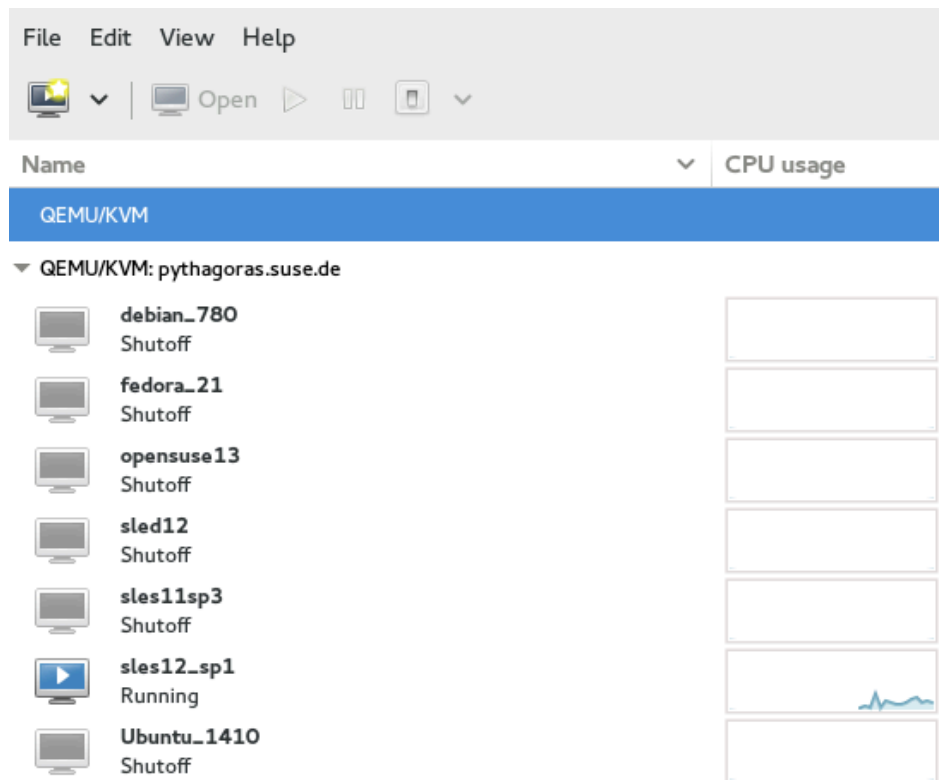
A command line tool for creating new VM Guests using the libvirt library. It supports graphical installations via VNC or *SPICE* protocols. Given suitable command line arguments, **virt-install** can run completely unattended. This allows for easy automation of guest installs. **virt-install** is the default installation tool used by the Virtual Machine Manager.

5.2 Virtualization GUI Tools

The following libvirt-based graphical tools are available on openSUSE Leap. All tools are provided by packages carrying the tool's name.

Virtual Machine Manager (Package: virt-manager)

The Virtual Machine Manager is a desktop tool for managing VM Guests. It provides the ability to control the lifecycle of existing machines (start/shutdown, pause/resume, save/restore) and create new VM Guests. It allows managing various types of storage and virtual networks. It provides access to the graphical console of VM Guests with a built-in VNC viewer and can be used to view performance statistics. **virt-manager** supports connecting to a local libvirtd, managing a local VM Host Server, or a remote libvirtd managing a remote VM Host Server.



To start the Virtual Machine Manager, enter **virt-manager** at the command prompt.



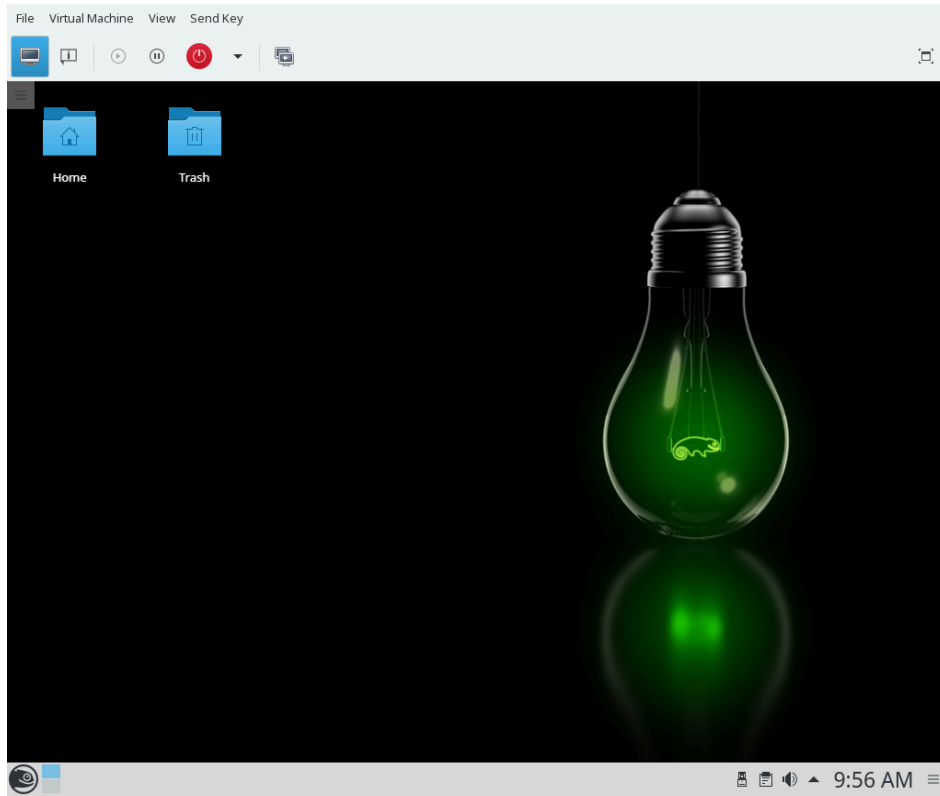
Note

To disable automatic USB device redirection for VM Guest using spice, either launch **virt-manager** with the **--spice-disable-auto-usbredir** parameter or run the following command to persistently change the default behavior:

```
tux > dconf write /org/virt-manager/virt-manager/console/auto-redirect false
```

virt-viewer (Package: **virt-viewer**)

A viewer for the graphical console of a VM Guest. It uses SPICE (configured by default on the VM Guest) or VNC protocols and supports TLS and x509 certificates. VM Guests can be accessed by name, ID, or UUID. If the guest is not already running, the viewer can be told to wait until the guest starts, before attempting to connect to the console. **virt-viewer** is not installed by default and is available after installing the package virt-viewer.

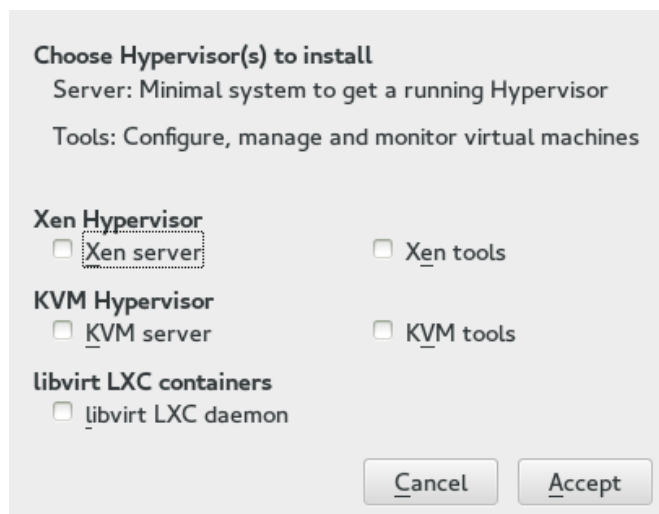


Note

To disable automatic USB device redirection for VM Guest using spice, add an empty filter using the `--spice-usbredir-auto-redirect-filter=''` parameter.

yast2 vm (Package: yast2-vm)

A YaST module that simplifies the installation of virtualization tools and can set up a network bridge:



6 Installation of Virtualization Components

Depending on the scope of the installation, none of the virtualization tools may be installed on your system. They will be automatically installed when configuring the hypervisor with the YaST module *Virtualization > Install Hypervisor and Tools*. In case this module is not available in YaST, install the package `yast2-vm`.

6.1 Installing KVM

To install KVM and KVM tools, proceed as follows:

1. Verify that the `yast2-vm` package is installed. This package is YaST's configuration tool that simplifies the installation of virtualization hypervisors.
2. Start YaST and choose *Virtualization > Install Hypervisor and Tools*.
3. Select *KVM server* for a minimal installation of QEMU tools. Select *KVM tools* if a `libvirt`-based management stack is also desired. Confirm with *Accept*.
4. To enable normal networking for the VM Guest, using a network bridge is recommended. YaST offers to automatically configure a bridge on the VM Host Server. Agree to do so by choosing *Yes*, otherwise choose *No*.
5. After the setup has been finished, you can start setting up VM Guests. Rebooting the VM Host Server is not required.

6.2 Installing Xen

To install Xen and Xen tools, proceed as follows:

1. Start YaST and choose *Virtualization > Install Hypervisor and Tools*.
2. Select *Xen server* for a minimal installation of Xen tools. Select *Xen tools* if a `libvirt`-based management stack is also desired. Confirm with *Accept*.
3. To enable normal networking for the VM Guest, using a network bridge is recommended. YaST offers to automatically configure a bridge on the VM Host Server. Agree to do so by choosing *Yes*, otherwise choose *No*.

4. After the setup has been finished, you need to reboot the machine with the *Xen kernel*.



Tip: Default Boot Kernel

If everything works as expected, change the default boot kernel with YaST and make the Xen-enabled kernel the default. For more information about changing the default kernel, see *Book "Reference", Chapter 12 "The Boot Loader GRUB 2", Section 12.3 "Configuring the Boot Loader with YaST"*.

6.3 Installing Containers

To install containers, proceed as follows:

1. Start YaST and choose *Virtualization > Install Hypervisor and Tools*.
2. Select *libvirt lxc daemon* and confirm with *Accept*.

6.4 Patterns

It is possible using Zypper and patterns to install virtualization packages. Run the command `zypper in -t pattern PATTERN`. Available patterns are:

KVM

- `kvm_server`: sets up the KVM VM Host Server with QEMU tools for management
- `kvm_tools`: installs the `libvirt` tools for managing and monitoring VM Guests

Xen

- `xen_server`: sets up the Xen VM Host Server with Xen tools for management
- `xen_tools`: installs the `libvirt` tools for managing and monitoring VM Guests

Containers

There is no pattern for containers; install the `libvirt-daemon-lxc` package.

6.5 Installing UEFI Support

UEFI support is provided by *OVMF (Open Virtual Machine Firmware)*. To enable UEFI boot, first install the `qemu-ovmf-x86_64` or `qemu-uefi-aarch64` package.

The firmware used by virtual machines is auto-selected. The auto-selection is based on the `*.json` files in the `qemu-ovmf-ARCH` package. The `libvirt` QEMU driver parses those files when loading so it knows the capabilities of the various types of firmware. Then when the user selects the type of firmware and any desired features (for example, support for secure boot), `libvirt` will be able to find a firmware that satisfies the user's requirements.

For example, to specify EFI with secure boot, use the following configuration:

```
<os firmware='efi'>
  <loader secure='yes' />
</os>
```

The `qemu-ovmf-ARCH` packages contain the following files:

```
root # rpm -ql qemu-ovmf-x86_64
[...]
/usr/share/qemu/ovmf-x86_64-ms-code.bin
/usr/share/qemu/ovmf-x86_64-ms-vars.bin
/usr/ddshare/qemu/ovmf-x86_64-ms.bin
/usr/share/qemu/ovmf-x86_64-suse-code.bin
/usr/share/qemu/ovmf-x86_64-suse-vars.bin
/usr/share/qemu/ovmf-x86_64-suse.bin
[...]
```

The `*-code.bin` files are the UEFI firmware files. The `*-vars.bin` files are corresponding variable store images that can be used as a template for a per-VM non-volatile store. `libvirt` copies the specified `vars` template to a per-VM path under `/var/lib/libvirt/qemu/nvram/` when first creating the VM. Files without `code` or `vars` in the name can be used as a single UEFI image. They are not as useful since no UEFI variables persist across power cycles of the VM. The `*-ms*.bin` files contain Microsoft keys as found on real hardware. Therefore, they are configured as the default in `libvirt`. Likewise, the `*-suse*.bin` files contain preinstalled SUSE keys. There is also a set of files with no preinstalled keys.

For details, see *Using UEFI and Secure Boot* and <http://www.linux-kvm.org/downloads/lersek/ovmf-whitepaper-c770f8c.txt>.

6.6 Enable Support for Nested Virtualization in KVM

! Important: Technology Preview

KVM's nested virtualization is still a technology preview. It is provided for testing purposes and is not supported.

Nested guests are KVM guests run in a KVM guest. When describing nested guests, we will use the following virtualization layers:

L0

A bare metal host running KVM.

L1

A virtual machine running on L0. Because it can run another KVM, it is called a *guest hypervisor*.

L2

A virtual machine running on L1. It is called a *nested guest*.

Nested virtualization has many advantages. You can benefit from it in the following scenarios:

- Manage your own virtual machines directly with your hypervisor of choice in cloud environments.
- Enable the live migration of hypervisors and their guest virtual machines as a single entity.
- Use it for software development and testing.

To enable nesting temporarily, remove the module and reload it with the `nested` KVM module parameter:

- For Intel CPUs, run:

```
tux > sudo modprobe -r kvm_intel && modprobe kvm_intel nested=1
```

- For AMD CPUs, run:

```
tux > sudo modprobe -r kvm_amd && modprobe kvm_amd nested=1
```

To enable nesting permanently, enable the `nested` KVM module parameter in the `/etc/modprobe.d/kvm_*.conf` file, depending on your CPU:

- For Intel CPUs, edit `/etc/modprobe.d/kvm_intel.conf` and add the following line:

```
options kvm_intel nested=Y
```

- For AMD CPUs, edit `/etc/modprobe.d/kvm_amd.conf` and add the following line:

```
options kvm_amd nested=Y
```

When your L0 host is capable of nesting, you will be able to start an L1 guest with one of the following ways:

- Use the `-cpu host` QEMU command line option.
- Add the `vmx` (for Intel CPUs) or the `svm` (for AMD CPUs) CPU feature to the `-cpu` QEMU command line option, which enables virtualization for the virtual CPU.

II Managing Virtual Machines with libvirt

- 7 Starting and Stopping libvirtd 24
- 8 Guest Installation 26
- 9 Basic VM Guest Management 32
- 10 Connecting and Authorizing 58
- 11 Managing Storage 79
- 12 Managing Networks 98
- 13 Configuring Virtual Machines with Virtual Machine Manager 113
- 14 Configuring Virtual Machines with **virsh** 133

7 Starting and Stopping libvirtd

The communication between the virtualization solutions (KVM, Xen, LXC) and the libvirt API is managed by the daemon `libvirtd`. It needs to run on the VM Host Server. libvirt client applications such as `virt-manager`, possibly running on a remote machine, communicate with `libvirtd` running on the VM Host Server, which services the request using native hypervisor APIs. Use the following commands to start and stop `libvirtd` or check its status:

```
tux > sudo systemctl start libvirtd

tux > sudo systemctl status libvirtd
libvirtd.service - Virtualization daemon
Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled)
Active: active (running) since Mon 2014-05-12 08:49:40 EDT; 2s ago
[...]

tux > sudo systemctl stop libvirtd

tux > sudo systemctl status libvirtd
[...]
Active: inactive (dead) since Mon 2014-05-12 08:51:11 EDT; 4s ago
[...]
```

To automatically start `libvirtd` at boot time, either activate it using the YaST *Services Manager* module or by entering the following command:

```
tux > sudo systemctl enable libvirtd
```



Important: Conflicting Services: libvirtd and xendomains

If `libvirtd` fails to start, check if the service `xendomains` is loaded:

```
tux > systemctl is-active xendomains
active
```

If the command returns `active`, you need to stop `xendomains` before you can start the `libvirtd` daemon. If you want `libvirtd` to also start after rebooting, additionally prevent `xendomains` from starting automatically. Disable the service:

```
tux > sudo systemctl stop xendomains
tux > sudo systemctl disable xendomains
```

```
tux > sudo systemctl start libvirtd
```

xendomains and libvirtd provide the same service and when used in parallel may interfere with one another. As an example, xendomains may attempt to start a domU already started by libvirtd.

8 Guest Installation

A VM Guest consists of an image containing an operating system and data files and a configuration file describing the VM Guest's virtual hardware resources. VM Guests are hosted on and controlled by the VM Host Server. This section provides generalized instructions for installing a VM Guest.

Virtual machines have few if any requirements above those required to run the operating system. If the operating system has not been optimized for the virtual machine host environment, it can only run on *hardware-assisted* virtualization computer hardware, in full virtualization mode, and requires specific device drivers to be loaded. The hardware that is presented to the VM Guest depends on the configuration of the host.

You should be aware of any licensing issues related to running a single licensed copy of an operating system on multiple virtual machines. Consult the operating system license agreement for more information.

8.1 GUI-Based Guest Installation

The *New VM* wizard helps you through the steps required to create a virtual machine and install its operating system. There are two ways to start it: Within Virtual Machine Manager, either click *Create New Virtual Machine* or choose *File > New Virtual Machine*. Alternatively, start YaST and choose *Virtualization > Create Virtual Machines for Xen and KVM*.

1. Start the *New VM* wizard either from YaST or Virtual Machine Manager.
2. Choose an installation source—either a locally available media or a network installation source. If you want to set up your VM Guest from an existing image, choose *import existing disk image*.

On a VM Host Server running the Xen hypervisor, you can choose whether to install a paravirtualized or a fully virtualized guest. The respective option is available under *Architecture Options*. Depending on this choice, not all installation options may be available.

3. Depending on your choice in the previous step, you need to provide the following data:

Local Installation Media (ISO image or CDROM)

Specify the path on the VM Host Server to an ISO image containing the installation data. If it is available as a volume in a libvirt storage pool, you can also select it using *Browse*. For more information, see [Chapter 11, Managing Storage](#).

Alternatively, choose a physical CD-ROM or DVD inserted in the optical drive of the VM Host Server.

Network Installation (HTTP, FTP, or NFS)

Provide the *URL* pointing to the installation source. Valid URL prefixes are, for example, `ftp://`, `http://`, `https://`, and `nfs://`.

Under *URL Options*, provide a path to an auto-installation file (AutoYaST or Kickstart, for example) and kernel parameters. Having provided a URL, the operating system should be automatically detected correctly. If this is not the case, deselect *Automatically Detect Operating System Based on Install-Media* and manually select the *OS Type* and *Version*.

Network Boot (PXE)

When booting via PXE, you only need to provide the *OS Type* and the *Version*.

Import Existing Disk Image

To set up the VM Guest from an existing image, you need to specify the path on the VM Host Server to the image. If it is available as a volume in a libvirt storage pool, you can also select it using *Browse*. For more information, see [Chapter 11, Managing Storage](#).

4. Choose the memory size and number of CPUs for the new virtual machine.
5. This step is omitted when *Import an Existing Image* is chosen in the first step.
Set up a virtual hard disk for the VM Guest. Either create a new disk image or choose an existing one from a storage pool (for more information, see [Chapter 11, Managing Storage](#)). If you choose to create a disk, a `qcow2` image will be created. By default, it is stored under `/var/lib/libvirt/images`.
Setting up a disk is optional. If you are running a live system directly from CD or DVD, for example, you can omit this step by deactivating *Enable Storage for this Virtual Machine*.
6. On the last screen of the wizard, specify the name for the virtual machine. To be offered the possibility to review and make changes to the virtualized hardware selection, activate *Customize configuration before install*. Find options to specify the network device under *Network Selection*.
Click *Finish*.

7. (Optional) If you kept the defaults in the previous step, the installation will now start. If you selected *Customize configuration before install*, a VM Guest configuration dialog opens. For more information about configuring VM Guests, see [Chapter 13, Configuring Virtual Machines with Virtual Machine Manager](#).

When you are done configuring, click *Begin Installation*.



Tip: Passing Key Combinations to Virtual Machines

The installation starts in a Virtual Machine Manager console window. Some key combinations, such as `Ctrl-Alt-F1`, are recognized by the VM Host Server but are not passed to the virtual machine. To bypass the VM Host Server, Virtual Machine Manager provides the “sticky key” functionality. Pressing `Ctrl`, `Alt`, or `Shift` three times makes the key sticky, then you can press the remaining keys to pass the combination to the virtual machine.

For example, to pass `Ctrl-Alt-F2` to a Linux virtual machine, press `Ctrl` three times, then press `Alt-F2`. You can also press `Alt` three times, then press `Ctrl-F2`.

The sticky key functionality is available in the Virtual Machine Manager during and after installing a VM Guest.

8.2 Installing from the Command Line with **virt-install**

virt-install is a command line tool that helps you create new virtual machines using the `libvirt` library. It is useful if you cannot use the graphical user interface, or need to automatize the process of creating virtual machines.

virt-install is a complex script with a lot of command line switches. The following are required. For more information, see the man page of **virt-install** (1).

General Options

- `--name VM_GUEST_NAME`: Specify the name of the new virtual machine. The name must be unique across all guests known to the hypervisor on the same connection. It is used to create and name the guest's configuration file and you can access the guest with this name from `virsh`. Alphanumeric and `_-.:+` characters are allowed.
- `--memory REQUIRED_MEMORY`: Specify the amount of memory to allocate for the new virtual machine in megabytes.
- `--vcpus NUMBER_OF_CPUS`: Specify the number of virtual CPUs. For best performance, the number of virtual processors should be less than or equal to the number of physical processors.

Virtualization Type

- `--paravirt`: Set up a paravirtualized guest. This is the default if the VM Host Server supports paravirtualization and full virtualization.
- `--hvm`: Set up a fully virtualized guest.
- `--virt-type HYPERVISOR`: Specify the hypervisor. Supported values are `kvm`, `xen`, or `lxc`.

Guest Storage

Specify one of `--disk`, `--filesystem` or `--nodisks` the type of the storage for the new virtual machine. For example, `--disk size=10` creates 10 GB disk in the default image location for the hypervisor and uses it for the VM Guest. `--filesystem /export/path/on/vmhost` specifies the directory on the VM Host Server to be exported to the guest. And `--nodisks` sets up a VM Guest without a local storage (good for Live CDs).

Installation Method

Specify the installation method using one of `--location`, `--cdrom`, `--pxe`, `--import`, or `--boot`.

Accessing the Installation

Use the `--graphics VALUE` option to specify how to access the installation. openSUSE Leap supports the values `vnc` or `none`.

If using VNC, `virt-install` tries to launch `virt-viewer`. If it is not installed or cannot be run, connect to the VM Guest manually with your preferred viewer. To explicitly prevent `virt-install` from launching the viewer use `--noautoconsole`. To define a password for accessing the VNC session, use the following syntax: `--graphics vnc,password=PASSWORD`.

In case you are using `--graphics none`, you can access the VM Guest through operating system supported services, such as SSH or VNC. Refer to the operating system installation manual on how to set up these services in the installation system.

Passing Kernel and Initrd Files

It is possible to directly specify the Kernel and Initrd of the installer, for example from a network source.

To pass additional boot parameters, use the `--extra-args` option. This can be used to specify a network configuration. For details, see <https://en.opensuse.org/SDB:Linuxrc>.

EXAMPLE 8.1: LOADING KERNEL AND INITRD FROM HTTP SERVER

```
root # virt-install --location \
"http://download.opensuse.org/pub/opensuse/distribution/leap/15.0/repo/oss" \
--extra-args="textmode=1" --name "Leap15" --memory 2048 --virt-type kvm \
--connect qemu:///system --disk size=10 --graphics vnc --network \
network=vnet_nated
```

Enabling the Console

By default, the console is not enabled for new virtual machines installed using `virt-install`. To enable it, use `--extra-args="console=ttyS0 textmode=1"` as in the following example:

```
tux > virt-install --virt-type kvm --name sles12 --memory 1024 \
--disk /var/lib/libvirt/images/disk1.qcow2 --os-variant sles12
--extra-args="console=ttyS0 textmode=1" --graphics none
```

After the installation finishes, the `/etc/default/grub` file in the VM image will be updated with the `console=ttyS0` option on the `GRUB_CMDLINE_LINUX_DEFAULT` line.

Using UEFI and Secure Boot

Install OVMF as described in [Section 6.5, "Installing UEFI Support"](#). Then add the `--boot uefi` option to the `virt-install` command.

Secure boot will be used automatically when setting up a new VM with OVMF. To use a specific firmware, use `--boot loader=/usr/share/qemu/ovmf-VERSION.bin`. Replace `VERSION` with the file you need.

EXAMPLE 8.2: EXAMPLE OF A virt-install COMMAND LINE

The following command line example creates a new SUSE Linux Enterprise Desktop 12 virtual machine with a virtio accelerated disk and network card. It creates a new 10 GB qcow2 disk image as a storage, the source installation media being the host CD-ROM drive. It will use VNC graphics, and it will auto-launch the graphical client.

KVM

```
tux > virt-install --connect qemu:///system --virt-type kvm --name sled12 \  
--memory 1024 --disk size=10 --cdrom /dev/cdrom --graphics vnc \  
--os-variant sled12
```

Xen

```
tux > virt-install --connect xen:// --virt-type xen --name sled12 \  
--memory 1024 --disk size=10 --cdrom /dev/cdrom --graphics vnc \  
--os-variant sled12
```

8.3 Advanced Guest Installation Scenarios

This section provides instructions for operations exceeding the scope of a normal installation, such as memory ballooning and installing add-on products.

8.3.1 Including Add-on Products in the Installation

Some operating systems such as openSUSE Leap offer to include add-on products in the installation process. If the add-on product installation source is provided via SUSE Customer Center, no special VM Guest configuration is needed. If it is provided via CD/DVD or ISO image, it is necessary to provide the VM Guest installation system with both the standard installation medium image and the image of the add-on product.

If you are using the GUI-based installation, select *Customize Configuration Before Install* in the last step of the wizard and add the add-on product ISO image via *Add Hardware > Storage*. Specify the path to the image and set the *Device Type* to *CD-ROM*.

If you are installing from the command line, you need to set up the virtual CD/DVD drives with the `--disk` parameter rather than with `--cdrom`. The device that is specified first is used for booting. The following example will install SUSE Linux Enterprise Server 15 together with SUSE Enterprise Storage extension:

```
tux > virt-install --name sles15+storage --memory 2048 --disk size=10 \  
--disk /path/to/SLE-15-SP2-Full-ARCH-GM-media1.iso-x86_64-GM-DVD1.iso,device=cdrom \  
--disk /path/to/SUSE-Enterprise-Storage-VERSION-DVD-ARCH-Media1.iso,device=cdrom \  
--graphics vnc --os-variant sles15
```

9 Basic VM Guest Management

Most management tasks, such as starting or stopping a VM Guest, can either be done using the graphical application Virtual Machine Manager or on the command line using `virsh`. Connecting to the graphical console via VNC is only possible from a graphical user interface.



Note: Managing VM Guests on a Remote VM Host Server

If started on a VM Host Server, the `libvirt` tools Virtual Machine Manager, `virsh`, and `virt-viewer` can be used to manage VM Guests on the host. However, it is also possible to manage VM Guests on a remote VM Host Server. This requires configuring remote access for `libvirt` on the host. For instructions, see [Chapter 10, Connecting and Authorizing](#).

To connect to such a remote host with Virtual Machine Manager, you need to set up a connection as explained in [Section 10.2.2, “Managing Connections with Virtual Machine Manager”](#). If connecting to a remote host using `virsh` or `virt-viewer`, you need to specify a connection URI with the parameter `-c` (for example, `virsh -c qemu+tls://saturn.example.com/system` or `virsh -c xen+ssh://`). The form of connection URI depends on the connection type and the hypervisor—see [Section 10.2, “Connecting to a VM Host Server”](#) for details.

Examples in this chapter are all listed without a connection URI.

9.1 Listing VM Guests

The VM Guest listing shows all VM Guests managed by `libvirt` on a VM Host Server.

9.1.1 Listing VM Guests with Virtual Machine Manager

The main window of the Virtual Machine Manager lists all VM Guests for each VM Host Server it is connected to. Each VM Guest entry contains the machine's name, its status (*Running*, *Paused*, or *Shutoff*) displayed as an icon and literally, and a CPU usage bar.

9.1.2 Listing VM Guests with **virsh**

Use the command **virsh list** to get a list of VM Guests:

List all running guests

```
tux > virsh list
```

List all running and inactive guests

```
tux > virsh list --all
```

For more information and further options, see **virsh help list** or **man 1 virsh**.

9.2 Accessing the VM Guest via Console

VM Guests can be accessed via a VNC connection (graphical console) or, if supported by the guest operating system, via a serial console.

9.2.1 Opening a Graphical Console

Opening a graphical console to a VM Guest lets you interact with the machine like a physical host via a VNC connection. If accessing the VNC server requires authentication, you are prompted to enter a user name (if applicable) and a password.

When you click into the VNC console, the cursor is “grabbed” and cannot be used outside the console anymore. To release it, press **Alt-Ctrl**.



Tip: Seamless (Absolute) Cursor Movement

To prevent the console from grabbing the cursor and to enable seamless cursor movement, add a tablet input device to the VM Guest. See [Section 13.5, “Input Devices”](#) for more information.

Certain key combinations such as **Ctrl-Alt-Del** are interpreted by the host system and are not passed to the VM Guest. To pass such key combinations to a VM Guest, open the *Send Key* menu from the VNC window and choose the desired key combination entry. The *Send Key*

menu is only available when using Virtual Machine Manager and **virt-viewer**. With Virtual Machine Manager, you can alternatively use the “sticky key” feature as explained in *Tip: Passing Key Combinations to Virtual Machines*.



Note: Supported VNC Viewers

Principally all VNC viewers can connect to the console of a VM Guest. However, if you are using SASL authentication and/or TLS/SSL connection to access the guest, the options are limited. Common VNC viewers such as **tightvnc** or **tigervnc** support neither SASL authentication nor TLS/SSL. The only supported alternative to Virtual Machine Manager and **virt-viewer** is Remmina (refer to *Book “Reference”, Chapter 4 “Remote Graphical Sessions with VNC”, Section 4.2 “Remmina: the Remote Desktop Client”*).

9.2.1.1 Opening a Graphical Console with Virtual Machine Manager

1. In the Virtual Machine Manager, right-click a VM Guest entry.
2. Choose *Open* from the pop-up menu.

9.2.1.2 Opening a Graphical Console with **virt-viewer**

virt-viewer is a simple VNC viewer with added functionality for displaying VM Guest consoles. For example, it can be started in “wait” mode, where it waits for a VM Guest to start before it connects. It also supports automatically reconnecting to a VM Guest that is rebooted.

virt-viewer addresses VM Guests by name, by ID or by UUID. Use **virsh list --all** to get this data.

To connect to a guest that is running or paused, use either the ID, UUID, or name. VM Guests that are shut off do not have an ID—you can only connect to them by UUID or name.

Connect to guest with the ID 8

```
tux > virt-viewer 8
```

Connect to the inactive guest named sles12; the connection window will open once the guest starts

```
tux > virt-viewer --wait sles12
```

With the `--wait` option, the connection will be upheld even if the VM Guest is not running at the moment. When the guest starts, the viewer will be launched.

For more information, see `virt-viewer --help` or `man 1 virt-viewer`.



Note: Password Input on Remote Connections with SSH

When using `virt-viewer` to open a connection to a remote host via SSH, the SSH password needs to be entered twice. The first time for authenticating with `libvirt`, the second time for authenticating with the VNC server. The second password needs to be provided on the command line where `virt-viewer` was started.

9.2.2 Opening a Serial Console

Accessing the graphical console of a virtual machine requires a graphical environment on the client accessing the VM Guest. As an alternative, virtual machines managed with `libvirt` can also be accessed from the shell via the serial console and `virsh`. To open a serial console to a VM Guest named “sles12”, run the following command:

```
tux > virsh console sles12
```

`virsh console` takes two optional flags: `--safe` ensures exclusive access to the console, `--force` disconnects any existing sessions before connecting. Both features need to be supported by the guest operating system.

Being able to connect to a VM Guest via serial console requires that the guest operating system supports serial console access and is properly supported. Refer to the guest operating system manual for more information.



Tip: Enabling Serial Console Access for SUSE Linux Enterprise and openSUSE Guests

Serial console access in SUSE Linux Enterprise and openSUSE is disabled by default. To enable it, proceed as follows:

SLES 12 and Up/openSUSE

Launch the YaST Boot Loader module and switch to the *Kernel Parameters* tab. Add console=ttyS0 to the field *Optional Kernel Command Line Parameter*.

SLES 11

Launch the YaST Boot Loader module and select the boot entry for which to activate serial console access. Choose *Edit* and add console=ttyS0 to the field *Optional Kernel Command Line Parameter*. Additionally, edit /etc/inittab and uncomment the line with the following content:

```
#S0:12345:respawn:/sbin/agetty -L 9600 ttyS0 vt102
```

9.3 Changing a VM Guest's State: Start, Stop, Pause

Starting, stopping or pausing a VM Guest can be done with either Virtual Machine Manager or **virsh**. You can also configure a VM Guest to be automatically started when booting the VM Host Server.

When shutting down a VM Guest, you may either shut it down gracefully, or force the shutdown. The latter is equivalent to pulling the power plug on a physical host and is only recommended if there are no alternatives. Forcing a shutdown may cause file system corruption and loss of data on the VM Guest.



Tip: Graceful Shutdown

To be able to perform a graceful shutdown, the VM Guest must be configured to support *ACPI*. If you have created the guest with the Virtual Machine Manager, *ACPI* should be available in the VM Guest.

Depending on the guest operating system, availability of *ACPI* may not be sufficient to perform a graceful shutdown. It is strongly recommended to test shutting down and re-booting a guest before using it in production. openSUSE or SUSE Linux Enterprise Desktop, for example, can require PolKit authorization for shutdown and reboot. Make sure this policy is turned off on all VM Guests.

If *ACPI* was enabled during a Windows XP/Windows Server 2003 guest installation, turning it on in the VM Guest configuration only is not sufficient. For more information, see:

- <https://support.microsoft.com/en-us/kb/314088> ↗
- <https://support.microsoft.com/en-us/kb/309283> ↗

Regardless of the VM Guest's configuration, a graceful shutdown is always possible from within the guest operating system.

9.3.1 Changing a VM Guest's State with Virtual Machine Manager

Changing a VM Guest's state can be done either from Virtual Machine Manager's main window, or from a VNC window.

PROCEDURE 9.1: STATE CHANGE FROM THE VIRTUAL MACHINE MANAGER WINDOW

1. Right-click a VM Guest entry.
2. Choose *Run*, *Pause*, or one of the *Shutdown options* from the pop-up menu.

PROCEDURE 9.2: STATE CHANGE FROM THE VNC WINDOW

1. Open a VNC Window as described in *Section 9.2.1.1, "Opening a Graphical Console with Virtual Machine Manager"*.
2. Choose *Run*, *Pause*, or one of the *Shut Down options* either from the toolbar or from the *Virtual Machine* menu.

9.3.1.1 Automatically Starting a VM Guest

You can automatically start a guest when the VM Host Server boots. This feature is not enabled by default and needs to be enabled for each VM Guest individually. There is no way to activate it globally.

1. Double-click the VM Guest entry in Virtual Machine Manager to open its console.
2. Choose *View > Details* to open the VM Guest configuration window.
3. Choose *Boot Options* and check *Start virtual machine on host boot up*.
4. Save the new configuration with *Apply*.

9.3.2 Changing a VM Guest's State with **virsh**

In the following examples, the state of a VM Guest named “sles12” is changed.

Start

```
tux > virsh start sles12
```

Pause

```
tux > virsh suspend sles12
```

Resume (a Suspended VM Guest)

```
tux > virsh resume sles12
```

Reboot

```
tux > virsh reboot sles12
```

Graceful shutdown

```
tux > virsh shutdown sles12
```

Force shutdown

```
tux > virsh destroy sles12
```

Turn on automatic start

```
tux > virsh autostart sles12
```

Turn off automatic start


```
tux > virsh autostart --disable sles12
```

9.4 Saving and Restoring the State of a VM Guest

Saving a VM Guest preserves the exact state of the guest's memory. The operation is similar to *hibernating* a computer. A saved VM Guest can be quickly restored to its previously saved running condition.

When saved, the VM Guest is paused, its current memory state is saved to disk, and then the guest is stopped. The operation does not make a copy of any portion of the VM Guest's virtual disk. The amount of time taken to save the virtual machine depends on the amount of memory allocated. When saved, a VM Guest's memory is returned to the pool of memory available on the VM Host Server.

The restore operation loads a VM Guest's previously saved memory state file and starts it. The guest is not booted but instead resumed at the point where it was previously saved. The operation is similar to coming out of hibernation.

The VM Guest is saved to a state file. Make sure there is enough space on the partition you are going to save to. For an estimation of the file size in megabytes to be expected, issue the following command on the guest:

```
tux > free -mh | awk '/^Mem:/ {print $3}'
```



Warning: Always Restore Saved Guests

After using the save operation, do not boot or start the saved VM Guest. Doing so would cause the machine's virtual disk and the saved memory state to get out of synchronization. This can result in critical errors when restoring the guest.

To be able to work with a saved VM Guest again, use the restore operation. If you used virsh to save a VM Guest, you cannot restore it using Virtual Machine Manager. In this case, make sure to restore using virsh.



Important: Only for VM Guests with Disk Types *raw*, *qcow2*

Saving and restoring VM Guests is only possible if the VM Guest is using a virtual disk of the type raw (.img), or *qcow2*.

9.4.1 Saving/Restoring with Virtual Machine Manager

PROCEDURE 9.3: SAVING A VM GUEST

1. Open a VNC connection window to a VM Guest. Make sure the guest is running.
2. Choose *Virtual Machine* > *Shutdown* > *Save*.

PROCEDURE 9.4: RESTORING A VM GUEST

1. Open a VNC connection window to a VM Guest. Make sure the guest is not running.
2. Choose *Virtual Machine* > *Restore*.

If the VM Guest was previously saved using Virtual Machine Manager, you will not be offered an option to *Run* the guest. However, note the caveats on machines saved with **virsh** outlined in *Warning: Always Restore Saved Guests*.

9.4.2 Saving and Restoring with **virsh**

Save a running VM Guest with the command **virsh save** and specify the file which it is saved to.

Save the guest named opensuse13

```
tux > virsh save opensuse13 /virtual/saves/opensuse13.vmsav
```

Save the guest with the ID 37

```
tux > virsh save 37 /virtual/saves/opensuse13.vmsave
```

To restore a VM Guest, use **virsh restore**:

```
tux > virsh restore /virtual/saves/opensuse13.vmsave
```

9.5 Creating and Managing Snapshots

VM Guest snapshots are snapshots of the complete virtual machine including the state of CPU, RAM, devices, and the content of all writable disks. To use virtual machine snapshots, all the attached hard disks need to use the qcow2 disk image format, and at least one of them needs to be writable.

Snapshots let you restore the state of the machine at a particular point in time. This is useful when undoing a faulty configuration or the installation of a lot of packages. After starting a snapshot that was created while the VM Guest was shut off, you will need to boot it. Any changes written to the disk after that point in time will be lost when starting the snapshot.



Note

Snapshots are supported on KVM VM Host Servers only.

9.5.1 Terminology

There are several specific terms used to describe the types of snapshots:

Internal snapshots

Snapshots that are saved into the qcow2 file of the original VM Guest. The file holds both the saved state of the snapshot and the changes made since the snapshot was taken. The main advantage of internal snapshots is that they are all stored in one file and therefore it is easy to copy or move them across multiple machines.

External snapshots

When creating an external snapshot, the original qcow2 file is saved and made read-only, while a new qcow2 file is created to hold the changes. The original file is sometimes called a 'backing' or 'base' file, while the new file with all the changes is called an 'overlay' or 'derived' file. External snapshots are useful when performing backups of VM Guests. However, external snapshots are not supported by Virtual Machine Manager, and cannot be deleted by **virsh** directly. For more information on external snapshots in QEMU, refer to [Section 28.2.4, "Manipulate Disk Images Effectively"](#).

Live snapshots

Snapshots created when the original VM Guest is running. Internal live snapshots support saving the devices, and memory and disk states, while external live snapshots with **virsh** support saving either the memory state, or the disk state, or both.

Offline snapshots

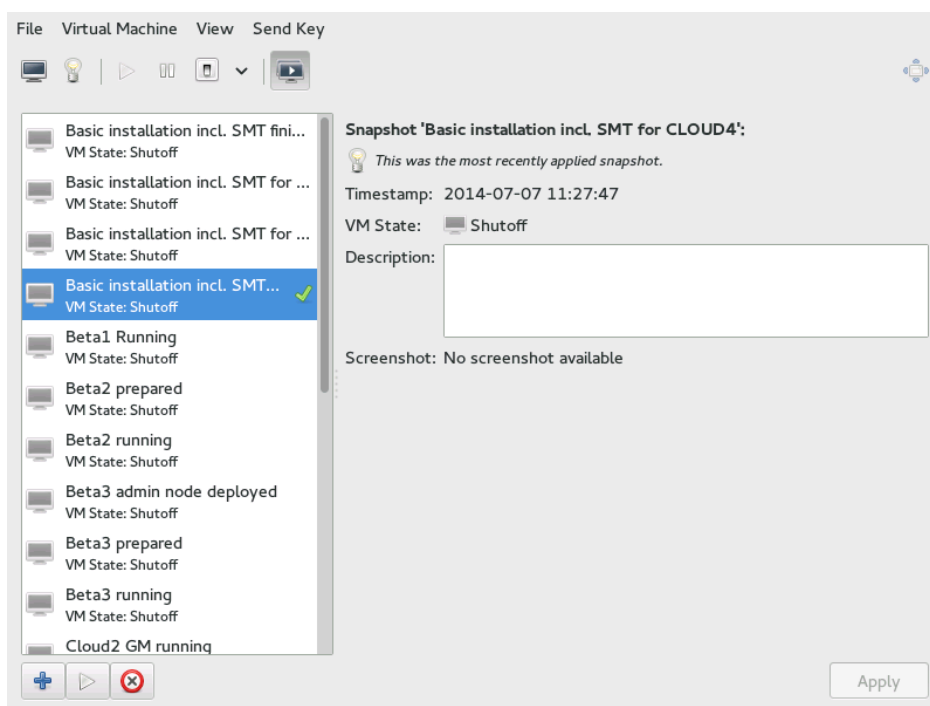
Snapshots created from a VM Guest that is shut off. This ensures data integrity as all the guest's processes are stopped and no memory is in use.

9.5.2 Creating and Managing Snapshots with Virtual Machine Manager

! Important: Internal Snapshots Only

Virtual Machine Manager supports only internal snapshots, either live or offline.

To open the snapshot management view in Virtual Machine Manager, open the VNC window as described in [Section 9.2.1.1, "Opening a Graphical Console with Virtual Machine Manager"](#). Now either choose *View > Snapshots* or click *Manage VM Snapshots* in the toolbar.



The list of existing snapshots for the chosen VM Guest is displayed in the left-hand part of the window. The snapshot that was last started is marked with a green tick. The right-hand part of the window shows details of the snapshot currently marked in the list. These details include the snapshot's title and time stamp, the state of the VM Guest at the time the snapshot was taken and a description. Snapshots of running guests also include a screenshot. The *Description* can be changed directly from this view. Other snapshot data cannot be changed.

9.5.2.1 Creating a Snapshot

To take a new snapshot of a VM Guest, proceed as follows:

1. Optionally, shut down the VM Guest if you want to create an offline snapshot.
2. Click *Add* in the bottom left corner of the VNC window.
The window *Create Snapshot* opens.
3. Provide a *Name* and, optionally, a description. The name cannot be changed after the snapshot has been taken. To be able to identify the snapshot later easily, use a “speaking name”.
4. Confirm with *Finish*.

9.5.2.2 Deleting a Snapshot

To delete a snapshot of a VM Guest, proceed as follows:

1. Click *Delete* in the bottom left corner of the VNC window.
2. Confirm the deletion with *Yes*.

9.5.2.3 Starting a Snapshot

To start a snapshot, proceed as follows:

1. Click *Run* in the bottom left corner of the VNC window.
2. Confirm the start with *Yes*.

9.5.3 Creating and Managing Snapshots with **virsh**

To list all existing snapshots for a domain (*admin_server* in the following), run the `snapshot-list` command:

```
tux > virsh snapshot-list --domain sle-ha-node1
Name                Creation Time      State
-----
sleha_12_sp2_b2_two_node_cluster 2016-06-06 15:04:31 +0200 shutoff
sleha_12_sp2_b3_two_node_cluster 2016-07-04 14:01:41 +0200 shutoff
sleha_12_sp2_b4_two_node_cluster 2016-07-14 10:44:51 +0200 shutoff
```

```
sleha_12_sp2_rc3_two_node_cluster 2016-10-10 09:40:12 +0200 shutoff
sleha_12_sp2_gmc_two_node_cluster 2016-10-24 17:00:14 +0200 shutoff
sleha_12_sp3_gm_two_node_cluster 2017-08-02 12:19:37 +0200 shutoff
sleha_12_sp3_rc1_two_node_cluster 2017-06-13 13:34:19 +0200 shutoff
sleha_12_sp3_rc2_two_node_cluster 2017-06-30 11:51:24 +0200 shutoff
sleha_15_b6_two_node_cluster 2018-02-07 15:08:09 +0100 shutoff
sleha_15_rc1_one-node 2018-03-09 16:32:38 +0100 shutoff
```

The snapshot that was last started is shown with the `snapshot-current` command:

```
tux > virsh snapshot-current --domain admin_server
Basic installation incl. SMT for CLOUD4
```

Details about a particular snapshot can be obtained by running the `snapshot-info` command:

```
tux > virsh snapshot-info --domain admin_server \
-name "Basic installation incl. SMT for CLOUD4"
Name:          Basic installation incl. SMT for CLOUD4
Domain:        admin_server
Current:       yes
State:         shutoff
Location:      internal
Parent:        Basic installation incl. SMT for CLOUD3-HA
Children:      0
Descendants:    0
Metadata:     yes
```

9.5.3.1 Creating Internal Snapshots

To take an internal snapshot of a VM Guest, either a live or offline, use the `snapshot-create-as` command as follows:

```
tux > virsh snapshot-create-as --domain admin_server ❶ --name "Snapshot 1" ❷ \
--description "First snapshot" ❸
```

- ❶ Domain name. Mandatory.
- ❷ Name of the snapshot. It is recommended to use a “speaking name”, since that makes it easier to identify the snapshot. Mandatory.
- ❸ Description for the snapshot. Optional.

9.5.3.2 Creating External Snapshots

With `virsh`, you can take external snapshots of the guest's memory state, disk state, or both.

To take both live and offline external snapshots of the guest's disk, specify the `--disk-only` option:

```
tux > virsh snapshot-create-as --domain admin_server --name \
"Offline external snapshot" --disk-only
```

You can specify the `--diskspec` option to control how the external files are created:

```
tux > virsh snapshot-create-as --domain admin_server --name \
"Offline external snapshot" \
--disk-only --diskspec vda,snapshot=external,file=/path/to/snapshot_file
```

To take a live external snapshot of the guest's memory, specify the `--live` and `--memspec` options:

```
tux > virsh snapshot-create-as --domain admin_server --name \
"Offline external snapshot" --live \
--memspec snapshot=external,file=/path/to/snapshot_file
```

To take a live external snapshot of both the guest's disk and memory states, combine the `--live`, `--diskspec`, and `--memspec` options:

```
tux > virsh snapshot-create-as --domain admin_server --name \
"Offline external snapshot" --live \
--memspec snapshot=external,file=/path/to/snapshot_file
--diskspec vda,snapshot=external,file=/path/to/snapshot_file
```

Refer to the *SNAPSHOT COMMANDS* section in [man 1 virsh](#) for more details.

9.5.3.3 Deleting a Snapshot

External snapshots cannot be deleted with `virsh`. To delete an internal snapshot of a VM Guest and restore the disk space it occupies, use the `snapshot-delete` command:

```
tux > virsh snapshot-delete --domain admin_server --snapshotname "Snapshot 2"
```

9.5.3.4 Starting a Snapshot

To start a snapshot, use the `snapshot-revert` command:

```
tux > virsh snapshot-revert --domain admin_server --snapshotname "Snapshot 1"
```

To start the current snapshot (the one the VM Guest was started off), it is sufficient to use `--current` rather than specifying the snapshot name:

```
tux > virsh snapshot-revert --domain admin_server --current
```

9.6 Deleting a VM Guest

By default, deleting a VM Guest using `virsh` removes only its XML configuration. Since attached storage is not deleted by default, you can reuse it with another VM Guest. With Virtual Machine Manager, you can also delete a guest's storage files as well—this will completely erase the guest.

9.6.1 Deleting a VM Guest with Virtual Machine Manager

1. In the Virtual Machine Manager, right-click a VM Guest entry.
2. From the context menu, choose *Delete*.
3. A confirmation window opens. Clicking *Delete* will permanently erase the VM Guest. The deletion is not recoverable.
You can also permanently delete the guest's virtual disk by activating *Delete Associated Storage Files*. The deletion is not recoverable either.

9.6.2 Deleting a VM Guest with `virsh`

To delete a VM Guest, it needs to be shut down first. It is not possible to delete a running guest. For information on shutting down, see [Section 9.3, “Changing a VM Guest's State: Start, Stop, Pause”](#).

To delete a VM Guest with `virsh`, run `virsh undefine VM_NAME`.

```
tux > virsh undefine sles12
```

There is no option to automatically delete the attached storage files. If they are managed by libvirt, delete them as described in [Section 11.2.4, “Deleting Volumes from a Storage Pool”](#).

9.7 Migrating VM Guests

One of the major advantages of virtualization is that VM Guests are portable. When a VM Host Server needs to go down for maintenance, or when the host gets overloaded, the guests can easily be moved to another VM Host Server. KVM and Xen even support “live” migrations during which the VM Guest is constantly available.

9.7.1 Migration Requirements

To successfully migrate a VM Guest to another VM Host Server, the following requirements need to be met:

- It is recommended that the source and destination systems have the same architecture.
- Storage devices must be accessible from both machines (for example, via NFS or iSCSI) and must be configured as a storage pool on both machines. For more information, see *Chapter 11, Managing Storage*.
This is also true for CD-ROM or floppy images that are connected during the move. However, you can disconnect them prior to the move as described in *Section 13.11, “Ejecting and Changing Floppy or CD/DVD-ROM Media with Virtual Machine Manager”*.
- `libvirtd` needs to run on both VM Host Servers and you must be able to open a remote `libvirt` connection between the target and the source host (or vice versa). Refer to *Section 10.3, “Configuring Remote Connections”* for details.
- If a firewall is running on the target host, ports need to be opened to allow the migration. If you do not specify a port during the migration process, `libvirt` chooses one from the range 49152:49215. Make sure that either this range (recommended) or a dedicated port of your choice is opened in the firewall on the *target host*.
- Host and target machine should be in the same subnet on the network, otherwise networking will not work after the migration.
- All VM Host Servers participating in migration must have the same UID for the `qemu` user and the same GIDs for the `kvm`, `qemu`, and `libvirt` groups.
- No running or paused VM Guest with the same name must exist on the target host. If a shut down machine with the same name exists, its configuration will be overwritten.
- All CPU models except *host cpu* model are supported when migrating VM Guests.

- *SATA* disk device type is not migratable.
- File system pass-through feature is incompatible with migration.
- The VM Host Server and VM Guest need to have proper timekeeping installed. See *Chapter 16, VM Guest Clock Settings*.
- No physical devices can be passed from host to guest. Live migration is currently not supported when using devices with PCI pass-through or *SR-IOV*. If live migration needs to be supported, you need to use software virtualization (paravirtualization or full virtualization).
- Cache mode setting is an important setting for migration. See: *Section 15.5, “Effect of Cache Modes on Live Migration”*.
- The image directory should be located in the same path on both hosts.
- All hosts should be on the same level of microcode (especially the spectre microcode updates). This can be achieved by installing the latest updates of openSUSE Leap on all hosts.

9.7.2 Migrating with Virtual Machine Manager

When using the Virtual Machine Manager to migrate VM Guests, it does not matter on which machine it is started. You can start Virtual Machine Manager on the source or the target host or even on a third host. In the latter case you need to be able to open remote connections to both the target and the source host.

1. Start Virtual Machine Manager and establish a connection to the target or the source host. If the Virtual Machine Manager was started neither on the target nor the source host, connections to both hosts need to be opened.
2. Right-click the VM Guest that you want to migrate and choose *Migrate*. Make sure the guest is running or paused—it is not possible to migrate guests that are shut down.



Tip: Increasing the Speed of the Migration

To increase the speed of the migration somewhat, pause the VM Guest. This is the equivalent of the former so-called “offline migration” option of Virtual Machine Manager.

3. Choose a *New Host* for the VM Guest. If the desired target host does not show up, make sure that you are connected to the host.

To change the default options for connecting to the remote host, under *Connection*, set the *Mode*, and the target host's *Address* (IP address or host name) and *Port*. If you specify a *Port*, you must also specify an *Address*.

Under *Advanced options*, choose whether the move should be permanent (default) or temporary, using *Temporary move*.

Additionally, there is the option *Allow unsafe*, which allows migrating without disabling the cache of the VM Host Server. This can speed up the migration but only works when the current configuration allows for a consistent view of the VM Guest storage without using `cache="none" / 0_DIRECT`.



Note: Bandwidth Option

In recent versions of Virtual Machine Manager, the option of setting a bandwidth for the migration has been removed. To set a specific bandwidth, use `virsh` instead.

4. To perform the migration, click *Migrate*.

When the migration is complete, the *Migrate* window closes and the VM Guest is now listed on the new host in the Virtual Machine Manager window. The original VM Guest will still be available on the target host (in shut down state).

9.7.3 Migrating with `virsh`

To migrate a VM Guest with `virsh migrate`, you need to have direct or remote shell access to the VM Host Server, because the command needs to be run on the host. The migration command looks like this:

```
tux > virsh migrate [OPTIONS] VM_ID_or_NAME CONNECTION_URI [--migrateuri
tcp://REMOTE_HOST:PORT]
```

The most important options are listed below. See `virsh help migrate` for a full list.

`--live`

Does a live migration. If not specified, the guest will be paused during the migration (“offline migration”).

`--suspend`

Does an offline migration and does not restart the VM Guest on the target host.

--persistent

By default a migrated VM Guest will be migrated temporarily, so its configuration is automatically deleted on the target host if it is shut down. Use this switch to make the migration persistent.

--undefinesource

When specified, the VM Guest definition on the source host will be deleted after a successful migration (however, virtual disks attached to this guest will *not* be deleted).

--parallel --parallel-connections NUM_OF_CONNECTIONS

Parallel migration can be used to increase migration data throughput in cases where a single migration thread is not capable of saturating the network link between source and destination hosts. On hosts with 40 GB network interfaces it may require four migration threads to saturate the link. With parallel migration, the time required to migrate large memory VMs can be significantly reduced.

The following examples use `mercury.example.com` as the source system and `jupiter.example.com` as the target system; the VM Guest's name is `opensuse131` with Id `37`.

Offline migration with default parameters

```
tux > virsh migrate 37 qemu+ssh://tux@jupiter.example.com/system
```

Transient live migration with default parameters

```
tux > virsh migrate --live opensuse131 qemu+ssh://tux@jupiter.example.com/system
```

Persistent live migration; delete VM definition on source

```
tux > virsh migrate --live --persistent --undefinesource 37 \
qemu+tls://tux@jupiter.example.com/system
```

Offline migration using port 49152

```
tux > virsh migrate opensuse131 qemu+ssh://tux@jupiter.example.com/system \
--migrateuri tcp://@jupiter.example.com:49152
```



Note: Transient Compared to Persistent Migrations

By default `virsh migrate` creates a temporary (transient) copy of the VM Guest on the target host. A shut down version of the original guest description remains on the source host. A transient copy will be deleted from the server after it is shut down.

To create a permanent copy of a guest on the target host, use the switch `--persistent`. A shut down version of the original guest description remains on the source host, too. Use the option `--undefinesource` together with `--persistent` for a “real” move where a permanent copy is created on the target host and the version on the source host is deleted.

It is not recommended to use `--undefinesource` without the `--persistent` option, since this will result in the loss of both VM Guest definitions when the guest is shut down on the target host.

9.7.4 Step-by-Step Example

9.7.4.1 Exporting the Storage

First you need to export the storage, to share the Guest image between host. This can be done by an NFS server. In the following example we want to share the `/volume1/VM` directory for all machines that are on the network `10.0.1.0/24`. We will use a SUSE Linux Enterprise NFS server. As root user, edit the `/etc/exports` file and add:

```
/volume1/VM 10.0.1.0/24 (rw, sync, no_root_squash)
```

You need to restart the NFS server:

```
tux > sudo systemctl restart nfsserver
tux > sudo exportfs
/volume1/VM      10.0.1.0/24
```

9.7.4.2 Defining the Pool on the Target Hosts

On each host where you want to migrate the VM Guest, the pool must be defined to be able to access the volume (that contains the Guest image). Our NFS server IP address is 10.0.1.99, its share is the `/volume1/VM` directory, and we want to get it mounted in the `/var/lib/libvirt/images/VM` directory. The pool name will be `VM`. To define this pool, create a `VM.xml` file with the following content:

```
<pool type='netfs'>
  <name>VM</name>
  <source>
    <host name='10.0.1.99' />
    <dir path='/volume1/VM' />
    <format type='auto' />
  </source>
  <target>
    <path>/var/lib/libvirt/images/VM</path>
    <permissions>
      <mode>0755</mode>
      <owner>-1</owner>
      <group>-1</group>
    </permissions>
  </target>
</pool>
```

Then load it into `libvirt` using the `pool-define` command:

```
root # virsh pool-define VM.xml
```

An alternative way to define this pool is to use the `virsh` command:

```
root # virsh pool-define-as VM --type netfs --source-host 10.0.1.99 \
  --source-path /volume1/VM --target /var/lib/libvirt/images/VM
Pool VM created
```

The following commands assume that you are in the interactive shell of `virsh` which can also be reached by using the command `virsh` without any arguments. Then the pool can be set to start automatically at host boot (autostart option):

```
virsh # pool-autostart VM
Pool VM marked as autostarted
```

If you want to disable the autostart:

```
virsh # pool-autostart VM --disable
```

```
Pool VM unmarked as autostarted
```

Check if the pool is present:

```
virsh # pool-list --all
Name                State      Autostart
-----
default             active    yes
VM                  active    yes

virsh # pool-info VM
Name:                VM
UUID:                42efe1b3-7eaa-4e24-a06a-ba7c9ee29741
State:               running
Persistent:         yes
Autostart:           yes
Capacity:            2,68 TiB
Allocation:          2,38 TiB
Available:           306,05 GiB
```



Warning: Pool Needs to Exist on All Target Hosts

Remember: this pool must be defined on each host where you want to be able to migrate your VM Guest.

9.7.4.3 Creating the Volume

The pool has been defined—now we need a volume which will contain the disk image:

```
virsh # vol-create-as VM sled12.qcow2 8G --format qcow2
Vol sled12.qcow2 created
```

The volume names shown will be used later to install the guest with `virt-install`.

9.7.4.4 Creating the VM Guest

Let's create a openSUSE Leap VM Guest with the `virt-install` command. The VM pool will be specified with the `--disk` option, `cache=none` is recommended if you do not want to use the `--unsafe` option while doing the migration.

```
root # virt-install --connect qemu:///system --virt-type kvm --name \
```

```
sled12 --memory 1024 --disk vol=VM/sled12.qcow2,cache=none --cdrom \  
/mnt/install/ISO/SLE-12-Desktop-DVD-x86_64-Build0327-Media1.iso --graphics \  
vnc --os-variant sled12  
Starting install...  
Creating domain...
```

9.7.4.5 Migrate the VM Guest

Everything is ready to do the migration now. Run the **migrate** command on the VM Host Server that is currently hosting the VM Guest, and choose the destination.

```
virsh # migrate --live sled12 --verbose qemu+ssh://IP/Hostname/system  
Password:  
Migration: [ 12 %]
```

9.8 Monitoring

9.8.1 Monitoring with Virtual Machine Manager

After starting Virtual Machine Manager and connecting to the VM Host Server, a CPU usage graph of all the running guests is displayed.

It is also possible to get information about disk and network usage with this tool, however, you must first activate this in *Preferences*:

1. Run **virt-manager**.
2. Select *Edit > Preferences*.
3. Change the tab from *General* to *Polling*.
4. Activate the check boxes for the kind of activity you want to see: *Poll Disk I/O*, *Poll Network I/O*, and *Poll Memory stats*.
5. If desired, also change the update interval using *Update status every n seconds*.
6. Close the *Preferences* dialog.
7. Activate the graphs that should be displayed under *View > Graph*.

Afterward, the disk and network statistics are also displayed in the main window of the Virtual Machine Manager.

More precise data is available from the VNC window. Open a VNC window as described in [Section 9.2.1, "Opening a Graphical Console"](#). Choose *Details* from the toolbar or the *View* menu. The statistics are displayed from the *Performance* entry of the left-hand tree menu.

9.8.2 Monitoring with **virt-top**

virt-top is a command line tool similar to the well-known process monitoring tool **top**. **virt-top** uses libvirt and therefore is capable of showing statistics for VM Guests running on different hypervisors. It is recommended to use **virt-top** instead of hypervisor-specific tools like **xentop**.

By default **virt-top** shows statistics for all running VM Guests. Among the data that is displayed is the percentage of memory used (**%MEM**) and CPU (**%CPU**) and the uptime of the guest (**TIME**). The data is updated regularly (every three seconds by default). The following shows the output on a VM Host Server with seven VM Guests, four of them inactive:

```
virt-top 13:40:19 - x86_64 8/8CPU 1283MHz 16067MB 7.6% 0.5%
7 domains, 3 active, 3 running, 0 sleeping, 0 paused, 4 inactive D:0 0:0 X:0
CPU: 6.1% Mem: 3072 MB (3072 MB by guests)

  ID S RDRQ WRRQ RXBY TXBY %CPU %MEM  TIME  NAME
  7 R 123  1 18K 196  5.8  6.0  0:24.35 sled12_sp1
  6 R  1  0 18K  0  0.2  6.0  0:42.51 sles12_sp1
  5 R  0  0 18K  0  0.1  6.0  85:45.67 opensuse_leap
  -                                     (Ubuntu_1410)
  -                                     (debian_780)
  -                                     (fedora_21)
  -                                     (sles11sp3)
```

By default the output is sorted by ID. Use the following key combinations to change the sort field:

Shift-P: CPU usage

Shift-M: Total memory allocated by the guest

Shift-T: Time

Shift-I: ID

To use any other field for sorting, press **Shift-F** and select a field from the list. To toggle the sort order, use **Shift-R**.

virt-top also supports different views on the VM Guests data, which can be changed on-the-fly by pressing the following keys:

- 0: default view
- 1: show physical CPUs
- 2: show network interfaces
- 3: show virtual disks

virt-top supports more hot keys to change the view of the data and many command line switches that affect the behavior of the program. For more information, see [man 1 virt-top](#).

9.8.3 Monitoring with **kvm_stat**

kvm_stat can be used to trace KVM performance events. It monitors `/sys/kernel/debug/kvm`, so it needs the debugfs to be mounted. On openSUSE Leap it should be mounted by default. In case it is not mounted, use the following command:

```
tux > sudo mount -t debugfs none /sys/kernel/debug
```

kvm_stat can be used in three different modes:

```
kvm_stat           # update in 1 second intervals
kvm_stat -1        # 1 second snapshot
kvm_stat -l > kvmstats.log # update in 1 second intervals in log format
                    # can be imported to a spreadsheet
```

EXAMPLE 9.1: TYPICAL OUTPUT OF **kvm_stat**

```
kvm statistics

efer_reload          0      0
exits                11378946 218130
fpu_reload           62144   152
halt_exits           414866  100
halt_wakeup          260358  50
host_state_reload    539650  249
hypercalls           0        0
insn_emulation       6227331 173067
insn_emulation_fail  0        0
invlpg               227281  47
io_exits             113148  18
irq_exits            168474  127
irq_injections       482804  123
```

irq_window	51270	18
largepages	0	0
mmio_exits	6925	0
mmu_cache_miss	71820	19
mmu_flooded	35420	9
mmu_pde_zapped	64763	20
mmu_pte_updated	0	0
mmu_pte_write	213782	29
mmu_recycled	0	0
mmu_shadow_zapped	128690	17
mmu_unsync	46	-1
nmi_injections	0	0
nmi_window	0	0
pf_fixed	1553821	857
pf_guest	1018832	562
remote_tlb_flush	174007	37
request_irq	0	0
signal_exits	0	0
tlb_flush	394182	148

See <http://clalance.blogspot.com/2009/01/kvm-performance-tools.html> for further information on how to interpret these values.

10 Connecting and Authorizing

Managing several VM Host Servers, each hosting multiple VM Guests, quickly becomes difficult. One benefit of `libvirt` is the ability to connect to several VM Host Servers at once, providing a single interface to manage all VM Guests and to connect to their graphical console.

To ensure only authorized users can connect, `libvirt` offers several connection types (via TLS, SSH, Unix sockets, and TCP) that can be combined with different authorization mechanisms (socket, PolKit, SASL and Kerberos).

10.1 Authentication

The power to manage VM Guests and to access their graphical console is something that should be restricted to a well defined circle of persons. To achieve this goal, you can use the following authentication techniques on the VM Host Server:

- Access control for Unix sockets with permissions and group ownership. This method is available for `libvirtd` connections only.
- Access control for Unix sockets with PolKit. This method is available for local `libvirtd` connections only.
- User name and password authentication with SASL (Simple Authentication and Security Layer). This method is available for both, `libvirtd` and VNC connections. Using SASL does not require real user accounts on the server, since it uses its own database to store user names and passwords. Connections authenticated with SASL are encrypted.
- Kerberos authentication. This method, available for `libvirtd` connections only, is not covered in this manual. Refer to http://libvirt.org/auth.html#ACL_server_kerberos for details.
- Single password authentication. This method is available for VNC connections only.

Important: Authentication for `libvirtd` and VNC need to be configured separately

Access to the VM Guest's management functions (via `libvirtd`) on the one hand, and to its graphical console on the other hand, always needs to be configured separately. When restricting access to the management tools, these restrictions do *not* automatically apply to VNC connections!

When accessing VM Guests from remote via TLS/SSL connections, access can be indirectly controlled on each client by restricting read permissions to the certificate's key file to a certain group. See [Section 10.3.2.5, "Restricting Access \(Security Considerations\)"](#) for details.

10.1.1 `libvirtd` Authentication

`libvirtd` authentication is configured in `/etc/libvirt/libvirtd.conf`. The configuration made here applies to all `libvirt` tools such as the Virtual Machine Manager or `virsh`.

`libvirt` offers two sockets: a read-only socket for monitoring purposes and a read-write socket to be used for management operations. Access to both sockets can be configured independently. By default, both sockets are owned by `root.root`. Default access permissions on the read-write socket are restricted to the user `root` (`0700`) and fully open on the read-only socket (`0777`).

In the following instructions, you will learn how to configure access permissions for the read-write socket. The same instructions also apply to the read-only socket. All configuration steps need to be carried out on the VM Host Server.

Note: Default Authentication Settings on openSUSE Leap

The default authentication method on openSUSE Leap is access control for Unix sockets. Only the user `root` may authenticate. When accessing the `libvirt` tools as a non-root user directly on the VM Host Server, you need to provide the `root` password through PolKit once. You are then granted access for the current and for future sessions.

Alternatively, you can configure `libvirt` to allow “system” access to non-privileged users. See [Section 10.2.1, ““system” Access for Non-Privileged Users”](#) for details.

RECOMMENDED AUTHORIZATION METHODS

Local Connections

Section 10.1.1.2, "Local Access Control for Unix Sockets with PolKit"

Section 10.1.1.1, "Access Control for Unix Sockets with Permissions and Group Ownership"

Remote Tunnel over SSH

Section 10.1.1.1, "Access Control for Unix Sockets with Permissions and Group Ownership"

Remote TLS/SSL Connection

Section 10.1.1.3, "User name and Password Authentication with SASL"

none (access controlled on the client side by restricting access to the certificates)

10.1.1.1 Access Control for Unix Sockets with Permissions and Group Ownership

To grant access for non-root accounts, configure the sockets to be owned and accessible by a certain group (libvirt in the following example). This authentication method can be used for local and remote SSH connections.

1. In case it does not exist, create the group that should own the socket:

```
tux > sudo groupadd libvirt
```

Important: Group Needs to Exist

The group must exist prior to restarting libvirtd. If not, the restart will fail.

2. Add the desired users to the group:

```
tux > sudo usermod --append --groups libvirt tux
```

3. Change the configuration in /etc/libvirt/libvirtd.conf as follows:

```
unix_sock_group = "libvirt" ❶  
unix_sock_rw_perms = "0770" ❷  
auth_unix_rw = "none" ❸
```

- ❶ Group ownership will be set to group libvirt.
- ❷ Sets the access permissions for the socket (srwxrwx---).
- ❸ Disables other authentication methods (PolKit or SASL). Access is solely controlled by the socket permissions.

4. Restart `libvirtd`:

```
tux > sudo systemctl start libvirtd
```

10.1.1.2 Local Access Control for Unix Sockets with PolKit

Access control for Unix sockets with PolKit is the default authentication method on openSUSE Leap for non-remote connections. Therefore, no `libvirt` configuration changes are needed. With PolKit authorization enabled, permissions on both sockets default to `0777` and each application trying to access a socket needs to authenticate via PolKit.

Important: PolKit Authentication for Local Connections Only

Authentication with PolKit can only be used for local connections on the VM Host Server itself, since PolKit does not handle remote authentication.

Two policies for accessing `libvirt`'s sockets exist:

- `org.libvirt.unix.monitor`: accessing the read-only socket
- `org.libvirt.unix.manage`: accessing the read-write socket

By default, the policy for accessing the read-write socket is to authenticate with the `root` password once and grant the privilege for the current and for future sessions.

To grant users access to a socket without having to provide the `root` password, you need to create a rule in `/etc/polkit-1/rules.d`. Create the file `/etc/polkit-1/rules.d/10-grant-libvirt` with the following content to grant access to the read-write socket to all members of the group `libvirt`:

```
polkit.addRule(function(action, subject) {
  if (action.id == "org.libvirt.unix.manage" && subject.isInGroup("libvirt")) {
    return polkit.Result.YES;
  }
});
```

10.1.1.3 User name and Password Authentication with SASL

SASL provides user name and password authentication and data encryption (digest-md5, by default). Since SASL maintains its own user database, the users do not need to exist on the VM Host Server. SASL is required by TCP connections and on top of TLS/SSL connections.

Important: Plain TCP and SASL with digest-md5 Encryption

Using digest-md5 encryption on an otherwise not encrypted TCP connection does not provide enough security for production environments. It is recommended to only use it in testing environments.

Tip: SASL Authentication on Top of TLS/SSL

Access from remote TLS/SSL connections can be indirectly controlled on the *client side* by restricting access to the certificate's key file. However, this might prove error-prone when dealing with many clients. Using SASL with TLS adds security by additionally controlling access on the server side.

To configure SASL authentication, proceed as follows:

1. Change the configuration in `/etc/libvirt/libvirtd.conf` as follows:

- a. To enable SASL for TCP connections:

```
auth_tcp = "sasl"
```

- b. To enable SASL for TLS/SSL connections:

```
auth_tls = "sasl"
```

2. Restart `libvirtd`:

```
tux > sudo systemctl restart libvirtd
```

3. The libvirt SASL configuration file is located at `/etc/sasl2/libvirtd.conf`. Normally, there is no need to change the defaults. However, if using SASL on top of TLS, you may turn off session encryption to avoid additional overhead (TLS connections are already encrypted) by commenting the line setting the `mech_list` parameter. Only do this for TLS/SASL, for TCP connections this parameter must be set to digest-md5.


```
#mech_list: digest-md5
```

4. By default, no SASL users are configured, so no logins are possible. Use the following commands to manage users:

Add the user `tux`

```
saslpasswd2 -a libvirt tux
```

Delete the user `tux`

```
saslpasswd2 -a libvirt -d tux
```

List existing users

```
sasldblistusers2 -f /etc/libvirt/passwd.db
```



Tip: `virsh` and SASL Authentication

When using SASL authentication, you will be prompted for a user name and password every time you issue a `virsh` command. Avoid this by using `virsh` in shell mode.

10.1.2 VNC Authentication

Since access to the graphical console of a VM Guest is not controlled by `libvirt`, but rather by the specific hypervisor, it is always necessary to additionally configure VNC authentication. The main configuration file is `/etc/libvirt/<hypervisor>.conf`. This section describes the QEMU/KVM hypervisor, so the target configuration file is `/etc/libvirt/qemu.conf`.



Note: VNC Authentication for Xen

In contrast to KVM and LXC, Xen does not yet offer more sophisticated VNC authentication than setting a password on a per VM basis. See the `<graphics type='vnc' ... libvirt` configuration option below.

Two authentication types are available: SASL and single password authentication. If you are using SASL for `libvirt` authentication, it is strongly recommended to use it for VNC authentication as well—it is possible to share the same database.

A third method to restrict access to the VM Guest is to enable the use of TLS encryption on the VNC server. This requires the VNC clients to have access to x509 client certificates. By restricting access to these certificates, access can indirectly be controlled on the client side. Refer to [Section 10.3.2.4.2, “VNC over TLS/SSL: Client Configuration”](#) for details.

10.1.2.1 User name and Password Authentication with SASL

SASL provides user name and password authentication and data encryption. Since SASL maintains its own user database, the users do not need to exist on the VM Host Server. As with SASL authentication for `libvirt`, you may use SASL on top of TLS/SSL connections. Refer to [Section 10.3.2.4.2, “VNC over TLS/SSL: Client Configuration”](#) for details on configuring these connections.

To configure SASL authentication for VNC, proceed as follows:

1. Create a SASL configuration file. It is recommended to use the existing `libvirt` file. If you have already configured SASL for `libvirt` and are planning to use the same settings including the same user name and password database, a simple link is suitable:

```
tux > sudo ln -s /etc/sasl2/libvirt.conf /etc/sasl2/qemu.conf
```

If are setting up SASL for VNC only or you are planning to use a different configuration than for `libvirt`, copy the existing file to use as a template:

```
tux > sudo cp /etc/sasl2/libvirt.conf /etc/sasl2/qemu.conf
```

Then edit it according to your needs.

2. Change the configuration in `/etc/libvirt/qemu.conf` as follows:

```
vnc_listen = "0.0.0.0"
vnc_sasl = 1
sasldb_path: /etc/libvirt/qemu_passwd.db
```

The first parameter enables VNC to listen on all public interfaces (rather than to the local host only), and the second parameter enables SASL authentication.

3. By default, no SASL users are configured, so no logins are possible. Use the following commands to manage users:

Add the user `tux`

```
tux > saslpasswd2 -f /etc/libvirt/qemu_passwd.db -a qemu tux
```

Delete the user tux

```
tux > saslpasswd2 -f /etc/libvirt/qemu_passwd.db -a qemu -d tux
```

List existing users

```
tux > sasldblistusers2 -f /etc/libvirt/qemu_passwd.db
```

4. Restart libvirtd:

```
tux > sudo systemctl restart libvirtd
```

5. Restart all VM Guests that have been running prior to changing the configuration. VM Guests that have not been restarted will not use SASL authentication for VNC connects.



Note: Supported VNC Viewers

SASL authentication is currently supported by Virtual Machine Manager and virt-viewer. Both of these viewers also support TLS/SSL connections.

10.1.2.2 Single Password Authentication

Access to the VNC server may also be controlled by setting a VNC password. You can either set a global password for all VM Guests or set individual passwords for each guest. The latter requires to edit the VM Guest's configuration files.



Note: Always Set a Global Password

If you are using single password authentication, it is good practice to set a global password even if setting passwords for each VM Guest. This will always leave your virtual machines protected with a “fallback” password if you forget to set a per-machine password. The global password will only be used if no other password is set for the machine.

PROCEDURE 10.1: SETTING A GLOBAL VNC PASSWORD

1. Change the configuration in /etc/libvirt/qemu.conf as follows:

```
vnc_listen = "0.0.0.0"
```

```
vnc_password = "PASSWORD"
```

The first parameter enables VNC to listen on all public interfaces (rather than to the local host only), and the second parameter sets the password. The maximum length of the password is eight characters.

2. Restart `libvirtd`:

```
tux > sudo systemctl restart libvirtd
```

3. Restart all VM Guests that have been running prior to changing the configuration. VM Guests that have not been restarted will not use password authentication for VNC connects.

PROCEDURE 10.2: SETTING A VM GUEST SPECIFIC VNC PASSWORD

1. Change the configuration in `/etc/libvirt/qemu.conf` as follows to enable VNC to listen on all public interfaces (rather than to the local host only).

```
vnc_listen = "0.0.0.0"
```

2. Open the VM Guest's XML configuration file in an editor. Replace `VM_NAME` in the following example with the name of the VM Guest. The editor that is used defaults to `$EDITOR`. If that variable is not set, `vi` is used.

```
tux > virsh edit VM_NAME
```

3. Search for the element `<graphics>` with the attribute `type='vnc'`, for example:

```
<graphics type='vnc' port='-1' autoport='yes'/>
```

4. Add the `passwd=PASSWORD` attribute, save the file and exit the editor. The maximum length of the password is eight characters.

```
<graphics type='vnc' port='-1' autoport='yes' passwd='PASSWORD'/>
```

5. Restart `libvirtd`:

```
tux > sudo systemctl restart libvirtd
```

6. Restart all VM Guests that have been running prior to changing the configuration. VM Guests that have not been restarted will not use password authentication for VNC connects.



Warning: Security of the VNC Protocol

The VNC protocol is not considered to be safe. Although the password is sent encrypted, it might be vulnerable when an attacker can sniff both the encrypted password and the encryption key. Therefore, it is recommended to use VNC with TLS/SSL or tunneled over SSH. `virt-viewer`, Virtual Machine Manager and Remmina (refer to *Book “Reference”, Chapter 4 “Remote Graphical Sessions with VNC”, Section 4.2 “Remmina: the Remote Desktop Client”*) support both methods.

10.2 Connecting to a VM Host Server

To connect to a hypervisor with `libvirt`, you need to specify a uniform resource identifier (URI). This URI is needed with `virsh` and `virt-viewer` (except when working as `root` on the VM Host Server) and is optional for the Virtual Machine Manager. Although the latter can be called with a connection parameter (for example, `virt-manager -c qemu:///system`), it also offers a graphical interface to create connection URIs. See [Section 10.2.2, “Managing Connections with Virtual Machine Manager”](#) for details.

```
HYPERVERSOR ① +PROTOCOL ② ://USER@REMOTE ③ /CONNECTION_TYPE ④
```

- ① Specify the hypervisor. openSUSE Leap currently supports the following hypervisors: `test` (dummy for testing), `qemu` (KVM), and `xen` (Xen). This parameter is mandatory.
- ② When connecting to a remote host, specify the protocol here. It can be one of: `ssh` (connection via SSH tunnel), `tcp` (TCP connection with SASL/Kerberos authentication), `tls` (TLS/SSL encrypted connection with authentication via x509 certificates).
- ③ When connecting to a remote host, specify the user name and the remote host name. If no user name is specified, the user name that has called the command (`$USER`) is used. See below for more information. For TLS connections, the host name needs to be specified exactly as in the x509 certificate.
- ④ When connecting to the `QEMU/KVM` hypervisor, two connection types are accepted: `system` for full access rights, or `session` for restricted access. Since `session` access is not supported on openSUSE Leap, this documentation focuses on `system` access.

EXAMPLE HYPERVISOR CONNECTION URIS

```
test:///default
```

Connect to the local dummy hypervisor. Useful for testing.

qemu:///system or xen:///system

Connect to the QEMU/Xen hypervisor on the local host having full access (type system).

qemu+ssh://tux@mercury.example.com/system or xen+ssh://tux@mercury.example.com/system

Connect to the QEMU/Xen hypervisor on the remote host mercury.example.com. The connection is established via an SSH tunnel.

qemu+tls://saturn.example.com/system or xen+tls://saturn.example.com/system

Connect to the QEMU/Xen hypervisor on the remote host mercury.example.com. The connection is established using TLS/SSL.

For more details and examples, refer to the libvirt documentation at <http://libvirt.org/uri.html>.



Note: User Names in URIs

A user name needs to be specified when using Unix socket authentication (regardless of whether using the user/password authentication scheme or PolKit). This applies to all SSH and local connections.

There is no need to specify a user name when using SASL authentication (for TCP or TLS connections) or when doing no additional server-side authentication for TLS connections. With SASL the user name will not be evaluated—you will be prompted for an SASL user/password combination in any case.

10.2.1 “system” Access for Non-Privileged Users

As mentioned above, a connection to the QEMU hypervisor can be established using two different protocols: session and system. A “session” connection is spawned with the same privileges as the client program. Such a connection is intended for desktop virtualization, since it is restricted (for example no USB/PCI device assignments, no virtual network setup, limited remote access to libvirtd).

The “system” connection intended for server virtualization has no functional restrictions but is, by default, only accessible by `root`. However, with the addition of the DAC (Discretionary Access Control) driver to `libvirt` it is now possible to grant non-privileged users “system” access. To grant “system” access to the user `tux`, proceed as follows:

PROCEDURE 10.3: GRANTING “SYSTEM” ACCESS TO A REGULAR USER

1. Enable access via Unix sockets as described in *Section 10.1.1.1, “Access Control for Unix Sockets with Permissions and Group Ownership”*. In that example access to `libvirt` is granted to all members of the group `libvirt` and `tux` made a member of this group. This ensures that `tux` can connect using `virsh` or Virtual Machine Manager.
2. Edit `/etc/libvirt/qemu.conf` and change the configuration as follows:

```
user = "tux"
group = "libvirt"
dynamic_ownership = 1
```

This ensures that the VM Guests are started by `tux` and that resources bound to the guest (for example virtual disks) can be accessed and modified by `tux`.

3. Make `tux` a member of the group `kvm`:

```
tux > sudo usermod --append --groups kvm tux
```

This step is needed to grant access to `/dev/kvm`, which is required to start VM Guests.

4. Restart `libvirtd`:

```
tux > sudo systemctl restart libvirtd
```

10.2.2 Managing Connections with Virtual Machine Manager

The Virtual Machine Manager uses a `Connection` for every VM Host Server it manages. Each connection contains all VM Guests on the respective host. By default, a connection to the local host is already configured and connected.

All configured connections are displayed in the Virtual Machine Manager main window. Active connections are marked with a small triangle, which you can click to fold or unfold the list of VM Guests for this connection.

Inactive connections are listed gray and are marked with Not Connected. Either double-click or right-click it and choose *Connect* from the context menu. You can also *Delete* an existing connection from this menu.



Note: Editing Existing Connections

It is not possible to edit an existing connection. To change a connection, create a new one with the desired parameters and delete the “old” one.

To add a new connection in the Virtual Machine Manager, proceed as follows:

1. Choose *File > Add Connection*
2. Choose the host's *Hypervisor (Xen or QEMU/KVM)*
3. (Optional) To set up a remote connection, choose *Connect to remote host*. For more information, see [Section 10.3, “Configuring Remote Connections”](#).
In case of a remote connection, specify the *Hostname* of the remote machine in the format *USERNAME@REMOTE _HOST*.



Important: Specifying a User Name

There is no need to specify a user name for TCP and TLS connections: In these cases, it will not be evaluated. However, in the case of SSH connections, specifying a user name is necessary when you want to connect as a user other than root.

4. If you do not want the connection to be automatically started when starting the Virtual Machine Manager, deactivate *Autoconnect*.
5. Finish the configuration by clicking *Connect*.

10.3 Configuring Remote Connections

A major benefit of libvirt is the ability to manage VM Guests on different remote hosts from a central location. This section gives detailed instructions on how to configure server and client to allow remote connections.

10.3.1 Remote Tunnel over SSH (qemu+ssh or xen+ssh)

Enabling a remote connection that is tunneled over SSH on the VM Host Server only requires the ability to accept SSH connections. Make sure the SSH daemon is started (`systemctl status sshd`) and that the ports for service `SSH` are opened in the firewall.

User authentication for SSH connections can be done using traditional file user/group ownership and permissions as described in *Section 10.1.1.1, "Access Control for Unix Sockets with Permissions and Group Ownership"*. Connecting as user `tux` (`qemu+ssh://tux@IVname;/system` or `xen+ssh://tux@IVname;/system`) works out of the box and does not require additional configuration on the `libvirt` side.

When connecting via SSH `qemu+ssh://USER@SYSTEM` or `xen+ssh://USER@SYSTEM` you need to provide the password for `USER`. This can be avoided by copying your public key to `~USER/.ssh/authorized_keys` on the VM Host Server as explained in *Book "Security Guide", Chapter 17 "SSH: Secure Network Operations", Section 17.5.2 "Copying an SSH Key"*. Using an `ssh-agent` on the machine from which you are connecting adds even more convenience. For more information, see *Book "Security Guide", Chapter 17 "SSH: Secure Network Operations", Section 17.5.3 "Using the ssh-agent"*.

10.3.2 Remote TLS/SSL Connection with x509 Certificate (qemu+tls or xen+tls)

Using TCP connections with TLS/SSL encryption and authentication via x509 certificates is much more complicated to set up than SSH, but it is a lot more scalable. Use this method if you need to manage several VM Host Servers with a varying number of administrators.

10.3.2.1 Basic Concept

TLS (Transport Layer Security) encrypts the communication between two computers by using certificates. The computer starting the connection is always considered the "client", using a "client certificate", while the receiving computer is always considered the "server", using a "server certificate". This scenario applies, for example, if you manage your VM Host Servers from a central desktop.

If connections are initiated from both computers, each needs to have a client *and* a server certificate. This is the case, for example, if you migrate a VM Guest from one host to another.

Each x509 certificate has a matching private key file. Only the combination of certificate and private key file can identify itself correctly. To assure that a certificate was issued by the assumed owner, it is signed and issued by a central certificate called certificate authority (CA). Both the client and the server certificates must be issued by the same CA.

Important: User Authentication

Using a remote TLS/SSL connection only ensures that two computers are allowed to communicate in a certain direction. Restricting access to certain users can indirectly be achieved on the client side by restricting access to the certificates. For more information, see [Section 10.3.2.5, “Restricting Access \(Security Considerations\)”](#).

`libvirt` also supports user authentication on the server with SASL. For more information, see [Section 10.3.2.6, “Central User Authentication with SASL for TLS Sockets”](#).

10.3.2.2 Configuring the VM Host Server

The VM Host Server is the machine receiving connections. Therefore, the *server* certificates need to be installed. The CA certificate needs to be installed, too. When the certificates are in place, TLS support can be turned on for `libvirt`.

1. Create the server certificate and export it together with the respective CA certificate.
2. Create the following directories on the VM Host Server:

```
tux > sudo mkdir -p /etc/pki/CA/ /etc/pki/libvirt/private/
```

Install the certificates as follows:

```
tux > sudo /etc/pki/CA/cacert.pem
tux > sudo /etc/pki/libvirt/servercert.pem
tux > sudo /etc/pki/libvirt/private/serverkey.pem
```

Important: Restrict Access to Certificates

Make sure to restrict access to certificates as explained in [Section 10.3.2.5, “Restricting Access \(Security Considerations\)”](#).

3. Enable TLS support by editing `/etc/libvirt/libvirtd.conf` and setting `listen_tls = 1`. Restart `libvirtd`:

```
tux > sudo systemctl restart libvirtd
```

4. By default, `libvirt` uses the TCP port 16514 for accepting secure TLS connections. Open this port in the firewall.

Important: Restarting `libvirtd` with TLS enabled

If you enable TLS for `libvirt`, the server certificates need to be in place, otherwise restarting `libvirtd` will fail. You also need to restart `libvirtd` in case you change the certificates.

10.3.2.3 Configuring the Client and Testing the Setup

The client is the machine initiating connections. Therefore the *client* certificates need to be installed. The CA certificate needs to be installed, too.

1. Create the client certificate and export it together with the respective CA certificate.
2. Create the following directories on the client:

```
tux > sudo mkdir -p /etc/pki/CA/ /etc/pki/libvirt/private/
```

Install the certificates as follows:

```
tux > sudo /etc/pki/CA/cacert.pem  
tux > sudo /etc/pki/libvirt/clientcert.pem  
tux > sudo /etc/pki/libvirt/private/clientkey.pem
```

Important: Restrict Access to Certificates

Make sure to restrict access to certificates as explained in [Section 10.3.2.5, "Restricting Access \(Security Considerations\)"](#).

3. Test the client/server setup by issuing the following command. Replace `mercury.example.com` with the name of your VM Host Server. Specify the same fully qualified host name as used when creating the server certificate.

```
#QEMU/KVM
virsh -c qemu+tls://mercury.example.com/system list --all

#Xen
virsh -c xen+tls://mercury.example.com/system list --all
```

If your setup is correct, you will see a list of all VM Guests registered with libvirt on the VM Host Server.

10.3.2.4 Enabling VNC for TLS/SSL connections

Currently, VNC communication over TLS is only supported by a few tools. Common VNC viewers such as tightvnc or tigervnc do not support TLS/SSL. The only supported alternative to Virtual Machine Manager and virt-viewer is remmina (refer to *Book "Reference", Chapter 4 "Remote Graphical Sessions with VNC", Section 4.2 "Remmina: the Remote Desktop Client"*).

10.3.2.4.1 VNC over TLS/SSL: VM Host Server Configuration

To access the graphical console via VNC over TLS/SSL, you need to configure the VM Host Server as follows:

1. Open ports for the service VNC in your firewall.
2. Create a directory /etc/pki/libvirt-vnc and link the certificates into this directory as follows:

```
tux > sudo mkdir -p /etc/pki/libvirt-vnc && cd /etc/pki/libvirt-vnc
tux > sudo ln -s /etc/pki/CA/cacert.pem ca-cert.pem
tux > sudo ln -s /etc/pki/libvirt/servercert.pem server-cert.pem
tux > sudo ln -s /etc/pki/libvirt/private/serverkey.pem server-key.pem
```

3. Edit /etc/libvirt/qemu.conf and set the following parameters:

```
vnc_listen = "0.0.0.0"
    vnc_tls = 1
    vnc_tls_x509_verify = 1
```

4. Restart the libvirtd:

```
tux > sudo systemctl restart libvirtd
```

! Important: VM Guests Need to be Restarted

The VNC TLS setting is only set when starting a VM Guest. Therefore, you need to restart all machines that have been running prior to making the configuration change.

10.3.2.4.2 VNC over TLS/SSL: Client Configuration

The only action needed on the client side is to place the x509 client certificates in a location recognized by the client of choice. Unfortunately, Virtual Machine Manager and **virt-viewer** expect the certificates in a different location. Virtual Machine Manager can either read from a system-wide location applying to all users, or from a per-user location. Remmina (refer to *Book "Reference", Chapter 4 "Remote Graphical Sessions with VNC", Section 4.2 "Remmina: the Remote Desktop Client"*) asks for the location of certificates when initializing the connection to the remote VNC session.

Virtual Machine Manager (**virt-manager**)

To connect to the remote host, Virtual Machine Manager requires the setup explained in [Section 10.3.2.3, "Configuring the Client and Testing the Setup"](#). To be able to connect via VNC, the client certificates also need to be placed in the following locations:

System-wide location

/etc/pki/CA/cacert.pem
/etc/pki/libvirt-vnc/clientcert.pem
/etc/pki/libvirt-vnc/private/clientkey.pem

Per-user location

/etc/pki/CA/cacert.pem
~/.pki/libvirt-vnc/clientcert.pem
~/.pki/libvirt-vnc/private/clientkey.pem

virt-viewer

virt-viewer only accepts certificates from a system-wide location:

/etc/pki/CA/cacert.pem
/etc/pki/libvirt-vnc/clientcert.pem
/etc/pki/libvirt-vnc/private/clientkey.pem



Important: Restrict Access to Certificates

Make sure to restrict access to certificates as explained in [Section 10.3.2.5, "Restricting Access \(Security Considerations\)"](#).

10.3.2.5 Restricting Access (Security Considerations)

Each x509 certificate consists of two pieces: the public certificate and a private key. A client can only authenticate using both pieces. Therefore, any user that has read access to the client certificate and its private key can access your VM Host Server. On the other hand, an arbitrary machine equipped with the full server certificate can pretend to be the VM Host Server. Since this is probably not desirable, access to at least the private key files needs to be restricted as much as possible. The easiest way to control access to a key file is to use access permissions.

Server Certificates

Server certificates need to be readable for QEMU processes. On openSUSE Leap QEMU, processes started from `libvirt` tools are owned by `root`, so it is sufficient if the `root` can read the certificates:

```
tux > chmod 700 /etc/pki/libvirt/private/  
tux > chmod 600 /etc/pki/libvirt/private/serverkey.pem
```

If you change the ownership for QEMU processes in `/etc/libvirt/qemu.conf`, you also need to adjust the ownership of the key file.

System Wide Client Certificates

To control access to a key file that is available system-wide, restrict read access to a certain group, so that only members of that group can read the key file. In the following example, a group `libvirt` is created, and group ownership of the `clientkey.pem` file and its parent directory is set to `libvirt`. Afterward, the access permissions are restricted to owner and group. Finally the user `tux` is added to the group `libvirt`, and thus can access the key file.

```
CERTPATH="/etc/pki/libvirt/"  
# create group libvirt  
groupadd libvirt  
# change ownership to user root and group libvirt  
chown root.libvirt $CERTPATH/private $CERTPATH/clientkey.pem  
# restrict permissions  
chmod 750 $CERTPATH/private
```

```
chmod 640 $CERTPATH/private/clientkey.pem
# add user tux to group libvirt
usermod --append --groups libvirt tux
```

Per-User Certificates

User-specific client certificates for accessing the graphical console of a VM Guest via VNC need to be placed in the user's home directory in `~/.pki`. Contrary to SSH, for example, the VNC viewer using these certificates do not check the access permissions of the private key file. Therefore, it is solely the user's responsibility to make sure the key file is not readable by others.

10.3.2.5.1 Restricting Access from the Server Side

By default, every client that is equipped with appropriate client certificates may connect to a VM Host Server accepting TLS connections. Therefore, it is possible to use additional server-side authentication with SASL as described in *Section 10.1.1.3, "User name and Password Authentication with SASL"*.

It is also possible to restrict access with a whitelist of DNs (distinguished names), so only clients with a certificate matching a DN from the list can connect.

Add a list of allowed DNs to `tls_allowed_dn_list` in `/etc/libvirt/libvirtd.conf`. This list may contain wild cards. Do not specify an empty list, since that would result in refusing all connections.

```
tls_allowed_dn_list = [
    "C=US,L=Provo,O=SUSE Linux Products GmbH,OU=*,CN=venus.example.com,EMAIL=*",
    "C=DE,L=Nuremberg,O=SUSE Linux Products GmbH,OU=Documentation,CN=*" ]
```

Get the distinguished name of a certificate with the following command:

```
tux > certtool -i --infile /etc/pki/libvirt/clientcert.pem | grep "Subject:"
```

Restart `libvirtd` after having changed the configuration:

```
tux > sudo systemctl restart libvirtd
```

10.3.2.6 Central User Authentication with SASL for TLS Sockets

A direct user authentication via TLS is not possible—this is handled indirectly on each client via the read permissions for the certificates as explained in *Section 10.3.2.5, "Restricting Access (Security Considerations)"*. However, if a central, server-based user authentication is needed, `libvirt` also

allows to use SASL (Simple Authentication and Security Layer) on top of TLS for direct user authentication. See [Section 10.1.1.3, "User name and Password Authentication with SASL"](#) for configuration details.

10.3.2.7 Troubleshooting

10.3.2.7.1 Virtual Machine Manager/`virsh` Cannot Connect to Server

Check the following in the given order:

Is it a firewall issue (TCP port 16514 needs to be open on the server)?

Is the client certificate (certificate and key) readable by the user that has started Virtual Machine Manager/`virsh`?

Has the same full qualified host name as in the server certificate been specified with the connection?

Is TLS enabled on the server (`listen_tls = 1`)?

Has `libvirtd` been restarted on the server?

10.3.2.7.2 VNC Connection fails

Ensure that you can connect to the remote server using Virtual Machine Manager. If so, check whether the virtual machine on the server has been started with TLS support. The virtual machine's name in the following example is `sles`.

```
tux > ps ax | grep qemu | grep "\-name sles" | awk -F" -vnc " '{ print FS $2 }'
```

If the output does not begin with a string similar to the following, the machine has not been started with TLS support and must be restarted.

```
-vnc 0.0.0.0:0,tls,x509verify=/etc/pki/libvirt
```


11 Managing Storage

When managing a VM Guest on the VM Host Server itself, you can access the complete file system of the VM Host Server to attach or create virtual hard disks or to attach existing images to the VM Guest. However, this is not possible when managing VM Guests from a remote host. For this reason, `libvirt` supports so called “Storage Pools”, which can be accessed from remote machines.



Tip: CD/DVD ISO images

To be able to access CD/DVD ISO images on the VM Host Server from remote, they also need to be placed in a storage pool.

`libvirt` knows two different types of storage: volumes and pools.

Storage Volume

A storage volume is a storage device that can be assigned to a guest—a virtual disk or a CD/DVD/floppy image. Physically (on the VM Host Server) it can be a block device (a partition, a logical volume, etc.) or a file.

Storage Pool

A storage pool is a storage resource on the VM Host Server that can be used for storing volumes, similar to network storage for a desktop machine. Physically it can be one of the following types:

File System Directory (*dir*)

A directory for hosting image files. The files can be either one of the supported disk formats (raw or qcow2), or ISO images.

Physical Disk Device (*disk*)

Use a complete physical disk as storage. A partition is created for each volume that is added to the pool.

Pre-Formatted Block Device (*fs*)

Specify a partition to be used in the same way as a file system directory pool (a directory for hosting image files). The only difference to using a file system directory is that `libvirt` takes care of mounting the device.

iSCSI Target (*iscsi*)

Set up a pool on an iSCSI target. You need to have been logged in to the volume once before, to use it with `libvirt`. Use the YaST *iSCSI Initiator* to detect and log in to a volume. Volume creation on iSCSI pools is not supported, instead each existing Logical Unit Number (LUN) represents a volume. Each volume/LUN also needs a valid (empty) partition table or disk label before you can use it. If missing, use `fdisk` to add it:

```
~ # fdisk -cu /dev/disk/by-path/ip-192.168.2.100:3260-iscsi-
iqn.2010-10.com.example:[...]-lun-2
Device contains neither a valid DOS partition table, nor Sun, SGI
or OSF disklabel
Building a new DOS disklabel with disk identifier 0xc15cdc4e.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

LVM Volume Group (logical)

Use an LVM volume group as a pool. You may either use a predefined volume group, or create a group by specifying the devices to use. Storage volumes are created as partitions on the volume.



Warning: Deleting the LVM-Based Pool

When the LVM-based pool is deleted in the Storage Manager, the volume group is deleted as well. This results in a non-recoverable loss of all data stored on the pool!

Multipath Devices (*mpath*)

At the moment, multipathing support is limited to assigning existing devices to the guests. Volume creation or configuring multipathing from within `libvirt` is not supported.

Network Exported Directory (*netfs*)

Specify a network directory to be used in the same way as a file system directory pool (a directory for hosting image files). The only difference to using a file system directory is that `libvirt` takes care of mounting the directory. Supported protocols are NFS and GlusterFS.

SCSI Host Adapter (*scsi*)

Use an SCSI host adapter in almost the same way as an iSCSI target. We recommend to use a device name from `/dev/disk/by-*` rather than `/dev/sdX`. The latter can change (for example, when adding or removing hard disks). Volume creation on iSCSI pools is not supported. Instead, each existing LUN (Logical Unit Number) represents a volume.



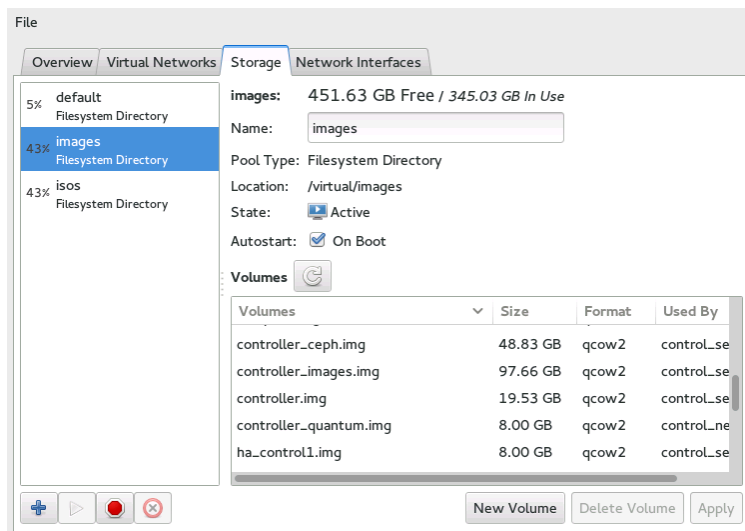
Warning: Security Considerations

To avoid data loss or data corruption, do not attempt to use resources such as LVM volume groups, iSCSI targets, etc., that are also used to build storage pools on the VM Host Server. There is no need to connect to these resources from the VM Host Server or to mount them on the VM Host Server—`libvirt` takes care of this.

Do not mount partitions on the VM Host Server by label. Under certain circumstances it is possible that a partition is labeled from within a VM Guest with a name already existing on the VM Host Server.

11.1 Managing Storage with Virtual Machine Manager

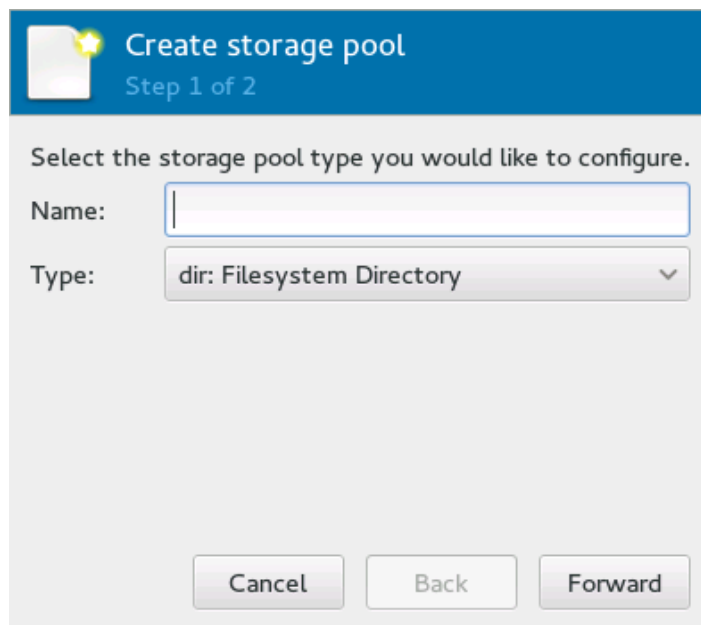
The Virtual Machine Manager provides a graphical interface—the Storage Manager—to manage storage volumes and pools. To access it, either right-click a connection and choose *Details*, or highlight a connection and choose *Edit > Connection Details*. Select the *Storage* tab.



11.1.1 Adding a Storage Pool

To add a storage pool, proceed as follows:

1. Click *Add* in the bottom left corner. The dialog *Add a New Storage Pool* appears.
2. Provide a *Name* for the pool (consisting of alphanumeric characters and `_ - .`) and select a *Type*. Proceed with *Forward*.



3. Specify the required details in the following window. The data that needs to be entered depends on the type of pool you are creating:

Typedir

- *Target Path:* Specify an existing directory.

Typedisk

- *Target Path:* The directory that hosts the devices. The default value `/dev` should usually fit.
- *Format:* Format of the device's partition table. Using `auto` should usually work. If not, get the required format by running the command `parted -l` on the VM Host Server.
- *Source Path:* Path to the device. It is recommended to use a device name from `/dev/disk/by-*` rather than the simple `/dev/sdX`, since the latter can change (for example, when adding or removing hard disks). You need to specify the path that resembles the whole disk, not a partition on the disk (if existing).
- *Build Pool:* Activating this option formats the device. Use with care—all data on the device will be lost!

Typefs

- *Target Path:* Mount point on the VM Host Server file system.
- *Format:* File system format of the device. The default value `auto` should work.
- *Source Path:* Path to the device file. It is recommended to use a device name from `/dev/disk/by-*` rather than `/dev/sdX`, because the latter can change (for example, when adding or removing hard disks).

Typeiscsi

Get the necessary data by running the following command on the VM Host Server:

```
tux > sudo iscsiadm --mode node
```

It will return a list of iSCSI volumes with the following format. The elements in bold text are required:

```
IP_ADDRESS:PORT,TPGT TARGET_NAME_(IQN)
```

- *Target Path*: The directory containing the device file. Use /dev/disk/by-path (default) or /dev/disk/by-id.
- *Host Name*: Host name or IP address of the iSCSI server.
- *Source Path*: The iSCSI target name (IQN).

Typelogical

- *Target Path*: In case you use an existing volume group, specify the existing device path. When building a new LVM volume group, specify a device name in the /dev directory that does not already exist.
- *Source Path*: Leave empty when using an existing volume group. When creating a new one, specify its devices here.
- *Build Pool*: Only activate when creating a new volume group.

Typempath

- *Target Path*: Support for multipathing is currently limited to making all multipath devices available. Therefore, specify an arbitrary string here that will then be ignored. The path is required, otherwise the XML parser will fail.

Typenetfs

- *Target Path*: Mount point on the VM Host Server file system.
- *Host Name*: IP address or host name of the server exporting the network file system.
- *Source Path*: Directory on the server that is being exported.

Typescsi

- *Target Path*: The directory containing the device file. Use /dev/disk/by-path (default) or /dev/disk/by-id.
- *Source Path*: Name of the SCSI adapter.



Note: File Browsing

Using the file browser by clicking *Browse* is not possible when operating from remote.

4. Click *Finish* to add the storage pool.

11.1.2 Managing Storage Pools

Virtual Machine Manager's Storage Manager lets you create or delete volumes in a pool. You may also temporarily deactivate or permanently delete existing storage pools. Changing the basic configuration of a pool is currently not supported by SUSE.

11.1.2.1 Starting, Stopping and Deleting Pools

The purpose of storage pools is to provide block devices located on the VM Host Server that can be added to a VM Guest when managing it from remote. To make a pool temporarily inaccessible from remote, click *Stop* in the bottom left corner of the Storage Manager. Stopped pools are marked with *State: Inactive* and are grayed out in the list pane. By default, a newly created pool will be automatically started *On Boot* of the VM Host Server.

To start an inactive pool and make it available from remote again, click *Start* in the bottom left corner of the Storage Manager.



Note: A Pool's State Does not Affect Attached Volumes

Volumes from a pool attached to VM Guests are always available, regardless of the pool's state (*Active* (stopped) or *Inactive* (started)). The state of the pool solely affects the ability to attach volumes to a VM Guest via remote management.

To permanently make a pool inaccessible, click *Delete* in the bottom left corner of the Storage Manager. You may only delete inactive pools. Deleting a pool does not physically erase its contents on VM Host Server—it only deletes the pool configuration. However, you need to be extra careful when deleting pools, especially when deleting LVM volume group-based tools:

Warning: Deleting Storage Pools

Deleting storage pools based on *local* file system directories, local partitions or disks has no effect on the availability of volumes from these pools currently attached to VM Guests. Volumes located in pools of type iSCSI, SCSI, LVM group or Network Exported Directory will become inaccessible from the VM Guest if the pool is deleted. Although the volumes themselves will not be deleted, the VM Host Server will no longer have access to the resources.

Volumes on iSCSI/SCSI targets or Network Exported Directory will become accessible again when creating an adequate new pool or when mounting/accessing these resources directly from the host system.

When deleting an LVM group-based storage pool, the LVM group definition will be erased and the LVM group will no longer exist on the host system. The configuration is not recoverable and all volumes from this pool are lost.

11.1.2.2 Adding Volumes to a Storage Pool

Virtual Machine Manager lets you create volumes in all storage pools, except in pools of types Multipath, iSCSI, or SCSI. A volume in these pools is equivalent to a LUN and cannot be changed from within `libvirt`.

1. A new volume can either be created using the Storage Manager or while adding a new storage device to a VM Guest. In either case, select a storage pool from the left panel, then click *Create new volume*.
2. Specify a *Name* for the image and choose an image format. SUSE currently only supports `raw` or `qcow2` images. The latter option is not available on LVM group-based pools.

Next to *Max Capacity*, specify the amount maximum size that the disk image is allowed to reach. Unless you are working with a `qcow2` image, you can also set an amount for *Allocation* that should be allocated initially. If both values differ, a sparse image file will be created which grows on demand.

For `qcow2` images, you can use a *Backing Store* (also called “backing file”) which constitutes a base image. The newly created `qcow2` image will then only record the changes that are made to the base image.

3. Start the volume creation by clicking *Finish*.

11.1.2.3 Deleting Volumes From a Storage Pool

Deleting a volume can only be done from the Storage Manager, by selecting a volume and clicking *Delete Volume*. Confirm with *Yes*.



Warning: Volumes Can Be Deleted Even While in Use

Volumes can be deleted even if they are currently used in an active or inactive VM Guest. There is no way to recover a deleted volume.

Whether a volume is used by a VM Guest is indicated in the *Used By* column in the Storage Manager.

11.2 Managing Storage with `virsh`

Managing storage from the command line is also possible by using `virsh`. However, creating storage pools is currently not supported by SUSE. Therefore, this section is restricted to documenting functions like starting, stopping and deleting pools and volume management.

A list of all `virsh` subcommands for managing pools and volumes is available by running `virsh help pool` and `virsh help volume`, respectively.

11.2.1 Listing Pools and Volumes

List all pools currently active by executing the following command. To also list inactive pools, add the option `--all`:

```
tux > virsh pool-list --details
```

Details about a specific pool can be obtained with the `pool-info` subcommand:

```
tux > virsh pool-info POOL
```

Volumes can only be listed per pool by default. To list all volumes from a pool, enter the following command.

```
tux > virsh vol-list --details POOL
```

At the moment `virsh` offers no tools to show whether a volume is used by a guest or not. The following procedure describes a way to list volumes from all pools that are currently used by a VM Guest.

PROCEDURE 11.1: LISTING ALL STORAGE VOLUMES CURRENTLY USED ON A VM HOST SERVER

1. Create an XSLT style sheet by saving the following content to a file, for example, `~/libvirt/guest_storage_list.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="text"/>
  <xsl:template match="text()"/>
  <xsl:strip-space elements="*" />
  <xsl:template match="disk">
    <xsl:text> </xsl:text>
    <xsl:value-of select="(source/@file|source/@dev|source/@dir)[1]"/>
    <xsl:text>&#10;</xsl:text>
  </xsl:template>
</xsl:stylesheet>
```

2. Run the following commands in a shell. It is assumed that the guest's XML definitions are all stored in the default location (`/etc/libvirt/qemu`). `xsltproc` is provided by the package `libxslt`.

```
SSHEET="$HOME/libvirt/guest_storage_list.xml"
cd /etc/libvirt/qemu
for FILE in *.xml; do
  basename $FILE .xml
  xsltproc $SSHEET $FILE
done
```

11.2.2 Starting, Stopping and Deleting Pools

Use the **virsh** pool subcommands to start, stop or delete a pool. Replace *PPOOL* with the pool's name or its UUID in the following examples:

Stopping a Pool

```
tux > virsh pool-destroy PPOOL
```



Note: A Pool's State Does not Affect Attached Volumes

Volumes from a pool attached to VM Guests are always available, regardless of the pool's state (*Active* (stopped) or *Inactive* (started)). The state of the pool solely affects the ability to attach volumes to a VM Guest via remote management.

Deleting a Pool

```
tux > virsh pool-delete PPOOL
```



Warning: Deleting Storage Pools

See [Warning: Deleting Storage Pools](#)

Starting a Pool

```
tux > virsh pool-start PPOOL
```

Enable Autostarting a Pool

```
tux > virsh pool-autostart PPOOL
```

Only pools that are marked to autostart will automatically be started if the VM Host Server reboots.

Disable Autostarting a Pool

```
tux > virsh pool-autostart PPOOL --disable
```

11.2.3 Adding Volumes to a Storage Pool

virsh offers two ways to add volumes to storage pools: either from an XML definition with `vol-create` and `vol-create-from` or via command line arguments with `vol-create-as`. The first two methods are currently not supported by SUSE, therefore this section focuses on the subcommand `vol-create-as`.

To add a volume to an existing pool, enter the following command:

```
tux > virsh vol-create-as POOL ① NAME ② 12G --format ③ raw|qcow2 ④ --allocation 4G ⑤
```

- ① Name of the pool to which the volume should be added
- ② Name of the volume
- ③ Size of the image, in this example 12 gigabytes. Use the suffixes k, M, G, T for kilobyte, megabyte, gigabyte, and terabyte, respectively.
- ④ Format of the volume. SUSE currently supports `raw`, and `qcow2`.
- ⑤ Optional parameter. By default **virsh** creates a sparse image file that grows on demand. Specify the amount of space that should be allocated with this parameter (4 gigabytes in this example). Use the suffixes k, M, G, T for kilobyte, megabyte, gigabyte, and terabyte, respectively.

When not specifying this parameter, a sparse image file with no allocation will be generated. To create a non-sparse volume, specify the whole image size with this parameter (would be `12G` in this example).

11.2.3.1 Cloning Existing Volumes

Another way to add volumes to a pool is to clone an existing volume. The new instance is always created in the same pool as the original.

```
tux > virsh vol-clone NAME_EXISTING_VOLUME ① NAME_NEW_VOLUME ② --pool POOL ③
```

- ① Name of the existing volume that should be cloned
- ② Name of the new volume
- ③ Optional parameter. `libvirt` tries to locate the existing volume automatically. If that fails, specify this parameter.

11.2.4 Deleting Volumes from a Storage Pool

To permanently delete a volume from a pool, use the subcommand `vol-delete`:

```
tux > virsh vol-delete NAME --pool POOL
```

`--pool` is optional. `libvirt` tries to locate the volume automatically. If that fails, specify this parameter.



Warning: No Checks Upon Volume Deletion

A volume will be deleted in any case, regardless of whether it is currently used in an active or inactive VM Guest. There is no way to recover a deleted volume.

Whether a volume is used by a VM Guest can only be detected by using by the method described in *Procedure 11.1, "Listing all Storage Volumes Currently Used on a VM Host Server"*.

11.2.5 Attaching Volumes to a VM Guest

After you create a volume as described in *Section 11.2.3, "Adding Volumes to a Storage Pool"*, you can attach it to a virtual machine and use it as a hard disk:

```
tux > virsh attach-disk DOMAIN SOURCE_IMAGE_FILE TARGET_DISK_DEVICE
```

For example:

```
tux > virsh attach-disk sles12sp3 /virt/images/example_disk.qcow2 sda2
```

To check if the new disk is attached, inspect the result of the `virsh dumpxml` command:

```
root # virsh dumpxml sles12sp3
[...]
<disk type='file' device='disk'>
  <driver name='qemu' type='raw' />
  <source file='/virt/images/example_disk.qcow2' />
  <backingStore />
  <target dev='sda2' bus='scsi' />
  <alias name='scsi0-0-0' />
  <address type='drive' controller='0' bus='0' target='0' unit='0' />
</disk>
[...]
```

11.2.5.1 Hotplug or Persistent Change

You can attach disks to both active and inactive domains. The attachment is controlled by the `--live` and `--config` options:

`--live`

Hotplugs the disk to an active domain. The attachment is not saved in the domain configuration. Using `--live` on an inactive domain is an error.

`--config`

Changes the domain configuration persistently. The attached disk is then available after the next domain start.

`--live --config`

Hotplugs the disk and adds it to the persistent domain configuration.



Tip: `virsh attach-device`

`virsh attach-device` is the more generic form of `virsh attach-disk`. You can use it to attach other types of devices to a domain.

11.2.6 Detaching Volumes from a VM Guest

To detach a disk from a domain, use `virsh detach-disk`:

```
root # virsh detach-disk DOMAIN TARGET_DISK_DEVICE
```

For example:

```
root # virsh detach-disk sles12sp3 sda2
```

You can control the attachment with the `--live` and `--config` options as described in [Section 11.2.5, "Attaching Volumes to a VM Guest"](#).

11.3 Locking Disk Files and Block Devices with `virtlockd`

Locking block devices and disk files prevents concurrent writes to these resources from different VM Guests. It provides protection against starting the same VM Guest twice, or adding the same disk to two different virtual machines. This will reduce the risk of a virtual machine's disk image becoming corrupted because of a wrong configuration.

The locking is controlled by a daemon called `virtlockd`. Since it operates independently from the `libvirtd` daemon, locks will endure a crash or a restart of `libvirtd`. Locks will even persist in the case of an update of the `virtlockd` itself, since it can re-execute itself. This ensures that VM Guests do *not* need to be restarted upon a `virtlockd` update. `virtlockd` is supported for KVM, QEMU, and Xen.

11.3.1 Enable Locking

Locking virtual disks is not enabled by default on openSUSE Leap. To enable and automatically start it upon rebooting, perform the following steps:

1. Edit `/etc/libvirt/qemu.conf` and set

```
lock_manager = "lockd"
```

2. Start the `virtlockd` daemon with the following command:

```
tux > sudo systemctl start virtlockd
```

3. Restart the `libvirtd` daemon with:

```
tux > sudo systemctl restart libvirtd
```

4. Make sure `virtlockd` is automatically started when booting the system:

```
tux > sudo systemctl enable virtlockd
```

11.3.2 Configure Locking

By default `virtlockd` is configured to automatically lock all disks configured for your VM Guests. The default setting uses a "direct" lockspace, where the locks are acquired against the actual file paths associated with the VM Guest `<disk>` devices. For example, `flock(2)` will be called directly on `/var/lib/libvirt/images/my-server/disk0.raw` when the VM Guest contains the following `<disk>` device:

```
<disk type='file' device='disk'>
  <driver name='qemu' type='raw' />
  <source file='/var/lib/libvirt/images/my-server/disk0.raw' />
  <target dev='vda' bus='virtio' />
</disk>
```

The `virtlockd` configuration can be changed by editing the file `/etc/libvirt/qemu-lockd.conf`. It also contains detailed comments with further information. Make sure to activate configuration changes by reloading `virtlockd`:

```
tux > sudo systemctl reload virtlockd
```

11.3.2.1 Enabling an Indirect Lockspace

The default configuration of `virtlockd` uses a "direct" lockspace. This means that the locks are acquired against the actual file paths associated with the `<disk>` devices.

If the disk file paths are not accessible to all hosts, `virtlockd` can be configured to allow an "indirect" lockspace. This means that a hash of the disk file path is used to create a file in the indirect lockspace directory. The locks are then held on these hash files instead of the actual disk file paths. Indirect lockspace is also useful if the file system containing the disk files does not support `fcntl()` locks. An indirect lockspace is specified with the `file_lockspace_dir` setting:

```
file_lockspace_dir = "/MY_LOCKSPACE_DIRECTORY"
```


11.3.2.2 Enable Locking on LVM or iSCSI Volumes

When wanting to lock virtual disks placed on LVM or iSCSI volumes shared by several hosts, locking needs to be done by UUID rather than by path (which is used by default). Furthermore, the lockspace directory needs to be placed on a shared file system accessible by all hosts sharing the volume. Set the following options for LVM and/or iSCSI:

```
lvm_lockspace_dir = "/MY_LOCKSPACE_DIRECTORY"  
iscsi_lockspace_dir = "/MY_LOCKSPACE_DIRECTORY"
```

11.4 Online Resizing of Guest Block Devices

Sometimes you need to change—extend or shrink—the size of the block device used by your guest system. For example, when the disk space originally allocated is no longer enough, it is time to increase its size. If the guest disk resides on a *logical volume*, you can resize it while the guest system is running. This is a big advantage over an offline disk resizing (see the **virt-resize** command from the *Section 17.3, “Guestfs Tools”* package) as the service provided by the guest is not interrupted by the resizing process. To resize a VM Guest disk, follow these steps:

PROCEDURE 11.2: ONLINE RESIZING OF GUEST DISK

1. Inside the guest system, check the current size of the disk (for example /dev/vda).

```
root # fdisk -l /dev/vda  
Disk /dev/sda: 160.0 GB, 160041885696 bytes, 312581808 sectors  
Units = sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

2. On the host, resize the logical volume holding the /dev/vda disk of the guest to the required size, for example 200 GB.

```
root # lvresize -L 200G /dev/mapper/vg00-home  
Extending logical volume home to 200 GiB  
Logical volume home successfully resized
```

3. On the host, resize the block device related to the disk /dev/mapper/vg00-home of the guest. Note that you can find the DOMAIN_ID with **virsh list**.

```
root # virsh blockresize --path /dev/vg00/home --size 200G DOMAIN_ID  
Block device '/dev/vg00/home' is resized
```

4. Check that the new disk size is accepted by the guest.

```
root # fdisk -l /dev/vda
Disk /dev/sda: 200.0 GB, 200052357120 bytes, 390727260 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

11.5 Sharing Directories between Host and Guests (File System Pass-Through)

libvirt allows to share directories between host and guests using QEMU's file system pass-through (also called VirtFS) feature. Such a directory can be also be accessed by several VM Guests at once and therefore be used to exchange files between VM Guests.



Note: Windows Guests and File System Pass-Through

Note that sharing directories between VM Host Server and Windows guests via File System Pass-Through does not work, because Windows lacks the drivers required to mount the shared directory.

To make a shared directory available on a VM Guest, proceed as follows:

1. Open the guest's console in Virtual Machine Manager and either choose *View > Details* from the menu or click *Show virtual hardware details* in the toolbar. Choose *Add Hardware > Filesystem* to open the *Filesystem Passthrough* dialog.
2. *Driver* allows you to choose between a *Handle* or *Path* base driver. The default setting is *Path*. *Mode* lets you choose the security model, which influences the way file permissions are set on the host. Three options are available:

Passthrough (Default)

Files on the file system are directly created with the client-user's credentials. This is very similar to what NFSv3 is using.

Squash

Same as *Passthrough*, but failure of privileged operations like chown are ignored. This is required when KVM is not run with root privileges.

Mapped

Files are created with the file server's credentials (`qemu.qemu`). The user credentials and the client-user's credentials are saved in extended attributes. This model is recommended when host and guest domains should be kept completely isolated.

3. Specify the path to the directory on the VM Host Server with *Source Path*. Enter a string at *Target Path* that will be used as a tag to mount the shared directory. Note that the string of this field is a tag only, not a path on the VM Guest.
4. Apply the setting. If the VM Guest is currently running, you need to shut it down to apply the new setting (rebooting the guest is not sufficient).
5. Boot the VM Guest. To mount the shared directory, enter the following command:

```
tux > sudo mount -t 9p -o trans=virtio,version=9p2000.L,rw TAG /MOUNT_POINT
```

To make the shared directory permanently available, add the following line to the `/etc/fstab` file:

```
TAG /MOUNT_POINT 9p trans=virtio,version=9p2000.L,rw 0 0
```

11.6 Using RADOS Block Devices with `libvirt`

RADOS Block Devices (RBD) store data in a Ceph cluster. They allow snapshotting, replication, and data consistency. You can use an RBD from your `libvirt`-managed VM Guests similarly to how you use other block devices.

12 Managing Networks

This chapter describes common network configurations for a VM Host Server, including those supported natively by the VM Host Server and `libvirt`. The configurations are valid for all hypervisors supported by openSUSE Leap, such as KVM or Xen.

There are two common network configurations to provide a VM Guest with a network connection:

- A *network bridge* acting as a Layer 2 switch
- A *virtual network* managed by `libvirt` with Layer 3 forwarding enabled

12.1 Network Bridge

The network bridge configuration provides a Layer 2 switch for VM Guests, switching Layer 2 Ethernet packets between ports on the bridge based on MAC addresses associated with the ports. This gives the VM Guest Layer 2 access to the VM Host Server's network. This configuration is analogous to connecting the VM Guest's virtual Ethernet cable into a hub that is shared with the host and other VM Guests running on the host. The configuration is often referred to as *shared physical device*.

The network bridge configuration is the default configuration of openSUSE Leap when configured as a KVM or Xen hypervisor. It is the preferred configuration when you simply want to connect VM Guests to the VM Host Server's LAN.

12.1.1 Managing Network Bridges with YaST

This section includes procedures to add or remove network bridges with YaST.

12.1.1.1 Adding a Network Bridge

To add a network bridge on VM Host Server, follow these steps:

1. Start *YaST* > *System* > *Network Settings*.

2. Activate the *Overview* tab and click *Add*.
3. Select *Bridge* from the *Device Type* list and enter the bridge device interface name in the *Configuration Name* entry. Press the *Next* button to proceed.
4. In the *Address* tab, specify networking details such as DHCP/static IP address, subnet mask or host name.
Using *Dynamic Address* is only useful when also assigning a device to a bridge that is connected to a DHCP server.
If you intend to create a virtual bridge that has no connection to a real Ethernet device, use *Statically assigned IP Address*. In this case, it is a good idea to use addresses from the private IP address ranges, for example, 192.168.0.0/16, 172.16.0.0/12, or 10.0.0.0/8.
To create a bridge that should only serve as a connection between the different guests without connection to the host system, set the IP address to 0.0.0.0 and the subnet mask to 255.255.255.255. The network scripts handle this special address as an unset IP address.
5. Activate the *Bridged Devices* tab and activate the network devices you want to include in the network bridge.
6. Click *Next* to return to the *Overview* tab and confirm with *OK*. The new network bridge should now be active on VM Host Server.

12.1.1.2 Deleting a Network Bridge

To delete an existing network bridge, follow these steps:

1. Start *YaST* > *System* > *Network Settings*.
2. Select the bridge device you want to delete from the list in the *Overview* tab.
3. Delete the bridge with *Delete* and confirm with *OK*.

12.1.2 Managing Network Bridges from the Command Line

This section includes procedures to add or remove network bridges using the command line.

12.1.2.1 Adding a Network Bridge

To add a new network bridge device on VM Host Server, follow these steps:

1. Log in as `root` on the VM Host Server where you want to create a new network bridge.
2. Choose a name for the new bridge—`virbr_test` in our example—and run

```
root # ip link add name VIRBR_TEST type bridge
```

3. Check if the bridge was created on VM Host Server:

```
root # bridge vlan
[...]  
virbr_test 1 PVID Egress Untagged
```

`virbr_test` is present, but is not associated with any physical network interface.

4. Bring the network bridge up and add a network interface to the bridge:

```
root # ip link set virbr_test up  
root # ip link set eth1 master virbr_test
```

Important: Network Interface Must Be Unused

You can only assign a network interface that is not yet used by other network bridge.

5. Optionally, enable STP (see [Spanning Tree Protocol \(https://en.wikipedia.org/wiki/Spanning_Tree_Protocol\)](https://en.wikipedia.org/wiki/Spanning_Tree_Protocol)):

```
root # bridge link set dev virbr_test cost 4
```

12.1.2.2 Deleting a Network Bridge

To delete an existing network bridge device on VM Host Server from the command line, follow these steps:

1. Log in as `root` on the VM Host Server where you want to delete an existing network bridge.
2. List existing network bridges to identify the name of the bridge to remove:

```
root # bridge vlan
```

```
[...]
virbr_test 1 PVID Egress Untagged
```

3. Delete the bridge:

```
root # ip link delete dev virbr_test
```

12.1.3 Using VLAN Interfaces

Sometimes, it is necessary to create a private connection either between two VM Host Servers or between VM Guest systems. For example, to migrate VM Guest to hosts in a different network segment, or to create a private bridge that only VM Guest systems may connect to (even when running on different VM Host Server systems). An easy way to build such connections is to set up VLAN networks.

VLAN interfaces are commonly set up on the VM Host Server. They either interconnect the different VM Host Server systems, or they may be set up as a physical interface to an otherwise virtual-only bridge. It is even possible to create a bridge with a VLAN as a physical interface that has no IP address in the VM Host Server. That way, the guest systems have no possibility to access the host over this network.

Run the YaST module *System > Network Settings*. Follow this procedure to set up the VLAN device:

PROCEDURE 12.1: SETTING UP VLAN INTERFACES WITH YAST

1. Click *Add* to create a new network interface.
2. In the *Hardware Dialog*, select *Device Type VLAN*.
3. Change the value of *Configuration Name* to the ID of your VLAN. Note that VLAN ID 1 is commonly used for management purposes.
4. Click *Next*.
5. Select the interface that the VLAN device should connect to below *Real Interface for VLAN*. If the desired interface does not appear in the list, first set up this interface without an IP address.
6. Select the desired method for assigning an IP address to the VLAN device.
7. Click *Next* to finish the configuration.

It is also possible to use the VLAN interface as a physical interface of a bridge. This makes it possible to connect several VM Host Server-only networks and allows live migration of VM Guest systems that are connected to such a network.

YaST does not always allow setting no IP address. However, this may be a desired feature, especially if VM Host Server-only networks should be connected. In this case, use the special address `0.0.0.0` with netmask `255.255.255.255`. The system scripts handle this address as no IP address set.

12.2 Virtual Networks

`libvirt`-managed virtual networks are similar to bridged networks, but typically have no Layer 2 connection to the VM Host Server. Connectivity to the VM Host Server's physical network is accomplished with Layer 3 forwarding, which introduces additional packet processing on the VM Host Server as compared to a Layer 2 bridged network. Virtual networks also provide DHCP and DNS services for VM Guests. For more information on `libvirt`'s virtual networks see the *Network XML format* documentation at <https://libvirt.org/formatnetwork.html>.

A standard `libvirt` installation on openSUSE Leap already comes with a predefined virtual network named `default` that provides DHCP and DNS services for the network, along with connectivity to the VM Host Server's physical network using the network address translation (NAT) forwarding mode. Although it is predefined, the `default` virtual network must be explicitly enabled by the administrator. For more information on the forwarding modes supported by `libvirt` see the *Connectivity* section of the *Network XML format* documentation at <https://libvirt.org/formatnetwork.html#elementsConnect>.

`libvirt`-managed virtual networks can be used to satisfy a wide range of use-cases, but are commonly used on VM Host Servers that have a wireless connection or dynamic/sporadic network connectivity such as laptops. Virtual networks are also useful when the VM Host Server's network has limited IP addresses, allowing forwarding of packets between the virtual network and the VM Host Server's network. However, most server use-cases are better suited for the network bridge configuration, where VM Guests are connected to the VM Host Server's LAN.



Warning: Enabling Forwarding Mode

Enabling forwarding mode in a `libvirt` virtual network enables forwarding in the VM Host Server by setting `/proc/sys/net/ipv4/ip_forward` and `/proc/sys/net/ipv6/conf/all/forwarding` to 1, which essentially turns the VM Host Server into a router.

Restarting the VM Host Server's network may reset the values and disable forwarding. To avoid this behavior, explicitly enable forwarding in the VM Host Server by editing the `/etc/sysctl.conf` file and adding:

```
net.ipv4.ip_forward = 1
```

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

12.2.1 Managing Virtual Networks with Virtual Machine Manager

You can define, configure, and operate virtual networks with Virtual Machine Manager.

12.2.1.1 Defining Virtual Networks

1. Start Virtual Machine Manager. In the list of available connections, right-click the name of the connection for which you need to configure the virtual network, and then select *Details*.
2. In the *Connection Details* window, click the *Virtual Networks* tab. You can see the list of all virtual networks available for the current connection. On the right, there are details of the selected virtual network.

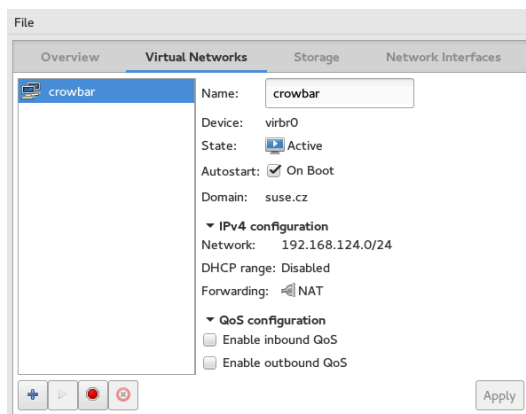


FIGURE 12.1: CONNECTION DETAILS

3. To add a new virtual network, click *Add*.
4. Specify a name for the new virtual network and click *Forward*.

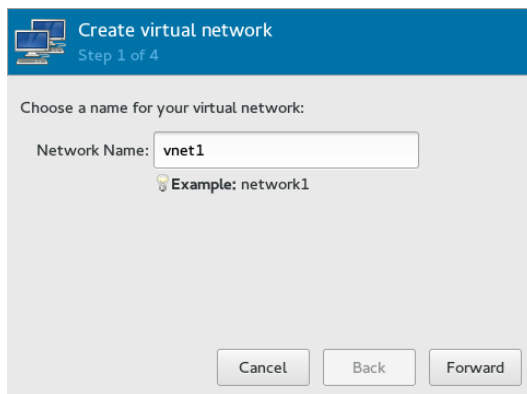


FIGURE 12.2: CREATE VIRTUAL NETWORK

5. To specify an IPv4 network address space definition, activate the relevant option and type it into the *Network* text entry.

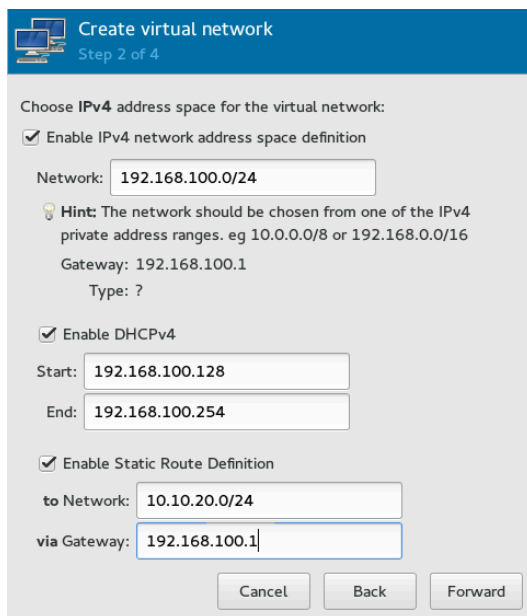


FIGURE 12.3: CREATE VIRTUAL NETWORK

6. `libvirt` can provide your virtual network with a DHCP server. If you need it, activate *Enable DHCPv4*, then type the start and end IP address range of assignable addresses.
7. To enable static routing for the new virtual network, activate the relevant option and type the network and gateway addresses.
8. Click *Forward* to proceed.

9. To specify IPv6-related options—network address space, DHCPv6 server, or static route—activate *Enable IPv6 network address space definition* and activate the relevant options and fill in the relevant boxes.
10. Click *Forward* to proceed.
11. Select whether you want an isolated or forwarded virtual network.

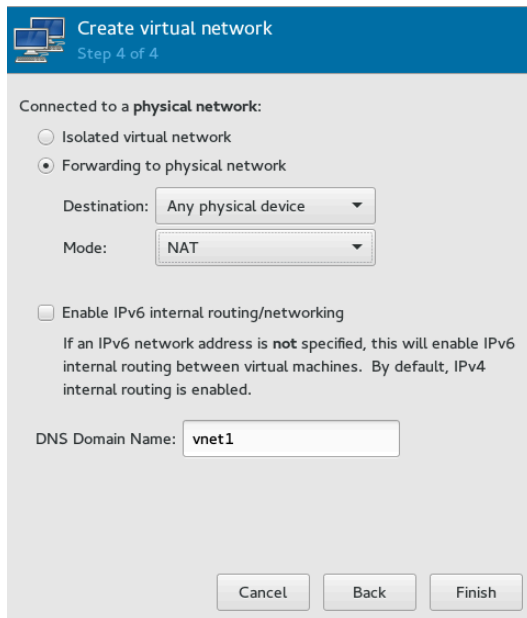


FIGURE 12.4: CREATE VIRTUAL NETWORK

For forwarded networks, specify the network interface to which the requests will be forwarded, and one of the forwarding modes: While *NAT* (network address translation) remaps the virtual network address space and allows sharing a single IP address, *Routed* forwards packets from the virtual network to the VM Host Server's physical network with no translation.

12. If you did not specify an IPv6 network address space definition earlier, you can enable IPv6 internal routing between virtual machines.
13. (*Optional*) Optionally, change the DNS domain name.
14. Click *Finish* to create the new virtual network. On the VM Host Server, a new virtual network bridge `virbrX` is available, which corresponds to the newly-created virtual network. You can check with `bridge link`. `libvirt` automatically adds iptables rules to allow traffic to/from guests attached to the new `virbrX` device.

12.2.1.2 Starting Virtual Networks

To start a virtual network that is temporarily stopped, follow these steps:

1. Start Virtual Machine Manager. In the list of available connections, right-click the name of the connection for which you need to configure the virtual network, and then select *Details*.
2. In the *Connection Details* window, click the *Virtual Networks* tab. You can see the list of all virtual networks available for the current connection.
3. To start the virtual network, click *Start*.

12.2.1.3 Stopping Virtual Networks

To stop an active virtual network, follow these steps:

1. Start Virtual Machine Manager. In the list of available connections, right-click the name of the connection for which you need to configure the virtual network, and then select *Details*.
2. In the *Connection Details* window, click the *Virtual Networks* tab. You can see the list of all virtual networks available for the current connection.
3. Select the virtual network to be stopped, then click *Stop*.

12.2.1.4 Deleting Virtual Networks

To delete a virtual network from VM Host Server, follow these steps:

1. Start Virtual Machine Manager. In the list of available connections, right-click the name of the connection for which you need to configure the virtual network, and then select *Details*.
2. In the *Connection Details* window, click the *Virtual Networks* tab. You can see the list of all virtual networks available for the current connection.
3. Select the virtual network to be deleted, then click *Delete*.

12.2.1.5 Obtaining IP Addresses with **nsswitch** for NAT Networks (in KVM)

- On VM Host Server, install `libvirt-nss`, which provides NSS support for `libvirt`:

```
tux > sudo zypper in libvirt-nss
```

- Add `libvirt` to `/etc/nsswitch.conf`:

```
...
hosts: files libvirt mdns_minimal [NOTFOUND=return] dns
...
```

- If `NSCD` is running, restart it:

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

Now you can reach the guest system by name from the host.

The NSS module has limited functionality. It reads `/var/lib/libvirt/dnsmasq/*.status` files to find the host name and corresponding IP addresses in a JSON record describing each lease provided by `dnsmasq`. Host name translation can only be done on those VM Host Servers using a `libvirt`-managed bridged network backed by `dnsmasq`.

For more information, see http://wiki.libvirt.org/page/NSS_module.

12.2.2 Managing Virtual Networks with **virsh**

You can manage `libvirt`-provided virtual networks with the `virsh` command line tool. To view all network related `virsh` commands, run

```
tux > sudo virsh help network
Networking (help keyword 'network'):
net-autostart          autostart a network
  net-create           create a network from an XML file
  net-define           define (but don't start) a network from an XML
file
  net-destroy          destroy (stop) a network
  net-dumpxml          network information in XML
  net-edit             edit XML configuration for a network
  net-event            Network Events
  net-info             network information
  net-list             list networks
  net-name             convert a network UUID to network name
  net-start            start a (previously defined) inactive network
```

net-undefine	undefine an inactive network
net-update	update parts of an existing network's configuration
net-uuid	convert a network name to network UUID

To view brief help information for a specific `virsh` command, run `virsh help VIRSH_COMMAND`:

```
tux > sudo virsh help net-create
NAME
  net-create - create a network from an XML file

SYNOPSIS
  net-create <file>

DESCRIPTION
  Create a network.

OPTIONS
  [--file] <string> file containing an XML network description
```

12.2.2.1 Creating a Network

To create a new *running* virtual network, run

```
tux > sudo virsh net-create VNET_DEFINITION.xml
```

The `VNET_DEFINITION.xml` XML file includes the definition of the virtual network that `libvirt` accepts.

To define a new virtual network without activating it, run

```
tux > sudo virsh net-define VNET_DEFINITION.xml
```

The following examples illustrate definitions of different types of virtual networks.

EXAMPLE 12.1: NAT BASED NETWORK

The following configuration allows VM Guests outgoing connectivity if it is available on VM Host Server. In the absence of VM Host Server networking, it allows guests to talk directly to each other.

```
<network>
<name>vnet_nated</name> ①
<bridge name="virbr1"/> ②
```

```

<forward mode="nat"/> ❸
<ip address="192.168.122.1" netmask="255.255.255.0"> ❹
  <dhcp>
    <range start="192.168.122.2" end="192.168.122.254"/> ❺
    <host mac="52:54:00:c7:92:da" name="host1.testing.com" \
      ip="192.168.1.23.101"/> ❻
    <host mac="52:54:00:c7:92:db" name="host2.testing.com" \
      ip="192.168.1.23.102"/>
    <host mac="52:54:00:c7:92:dc" name="host3.testing.com" \
      ip="192.168.1.23.103"/>
  </dhcp>
</ip>
</network>

```

- ❶ The name of the new virtual network.
- ❷ The name of the bridge device used to construct the virtual network. When defining a new network with a `<forward>` mode of "nat" or "route" (or an isolated network with no `<forward>` element), `libvirt` will automatically generate a unique name for the bridge device if none is given.
- ❸ Inclusion of the `<forward>` element indicates that the virtual network will be connected to the physical LAN. The `mode` attribute specifies the forwarding method. The most common modes are "nat" (default, network address translation), "route" (direct forwarding to the physical network, no address translation), and "bridge" (network bridge configured outside of `libvirt`). If the `<forward>` element is not specified, the virtual network will be isolated from other networks. For a complete list of forwarding modes, see <http://libvirt.org/formatnetwork.html#elementsConnect>.
- ❹ The IP address and netmask for the network bridge.
- ❺ Enable DHCP server for the virtual network, offering IP addresses ranging from the specified `start` and `end` attribute.
- ❻ The optional `<host>` elements specify hosts that will be given names and predefined IP addresses by the built-in DHCP server. Any IPv4 host element must specify the following: the MAC address of the host to be assigned a given name, the IP to be assigned to that host, and the name to be given to that host by the DHCP server. An IPv6 host element differs slightly from that for IPv4: there is no `mac` attribute since a MAC address has no defined meaning in IPv6. Instead, the `name` attribute is used to identify the host to be assigned the IPv6 address. For DHCPv6, the `name` is the plain name of the client host sent by the client to the server. Note that this method of assigning a specific IP address can also be used instead of the `mac` attribute for IPv4.

EXAMPLE 12.2: ROUTED NETWORK

The following configuration routes traffic from the virtual network to the LAN without applying any NAT. The IP address range must be preconfigured in the routing tables of the router on the VM Host Server network.

```
<network>
  <name>vnet_routed</name>
  <bridge name="virbr1"/>
  <forward mode="route" dev="eth1"/> ❶
  <ip address="192.168.122.1" netmask="255.255.255.0">
    <dhcp>
      <range start="192.168.122.2" end="192.168.122.254"/>
    </dhcp>
  </ip>
</network>
```

❶ The guest traffic may only go out via the `eth1` network device on the VM Host Server.

EXAMPLE 12.3: ISOLATED NETWORK

This configuration provides a completely isolated private network. The guests can talk to each other, and to VM Host Server, but cannot reach any other machines on the LAN, as the `<forward>` element is missing in the XML description.

```
<network>
  <name>vnet_isolated</name>
  <bridge name="virbr3"/>
  <ip address="192.168.152.1" netmask="255.255.255.0">
    <dhcp>
      <range start="192.168.152.2" end="192.168.152.254"/>
    </dhcp>
  </ip>
</network>
```

EXAMPLE 12.4: USING AN EXISTING BRIDGE ON VM HOST SERVER

This configuration shows how to use an existing VM Host Server's network bridge `br0`. VM Guests are directly connected to the physical network. Their IP addresses will all be on the subnet of the physical network, and there will be no restrictions on incoming or outgoing connections.

```
<network>
  <name>host-bridge</name>
  <forward mode="bridge"/>
  <bridge name="br0"/>
</network>
```


12.2.2.2 Listing Networks

To list all virtual networks available to `libvirt`, run:

```
tux > sudo virsh net-list --all
```

Name	State	Autostart	Persistent
crowbar	active	yes	yes
vnet_nated	active	yes	yes
vnet_routed	active	yes	yes
vnet_isolated	inactive	yes	yes

To list available domains, run:

```
tux > sudo virsh list
```

Id	Name	State
1	nated_sles12sp3	running
...		

To get a list of interfaces of a running domain, run `domifaddr DOMAIN`, or optionally specify the interface to limit the output to this interface. By default, it additionally outputs their IP and MAC addresses:

```
tux > sudo virsh domifaddr nated_sles12sp3 --interface vnet0 --source lease
```

Name	MAC address	Protocol	Address
vnet0	52:54:00:9e:0d:2b	ipv6	fd00:dead:beef:55::140/64
-	-	ipv4	192.168.100.168/24

To print brief information of all virtual interfaces associated with the specified domain, run:

```
tux > sudo virsh domiflist nated_sles12sp3
```

Interface	Type	Source	Model	MAC
vnet0	network	vnet_nated	virtio	52:54:00:9e:0d:2b

12.2.2.3 Getting Details about a Network

To get detailed information about a network, run:

```
tux > sudo virsh net-info vnet_routed
```

Name: vnet_routed

```
UUID:          756b48ff-d0c6-4c0a-804c-86c4c832a498
Active:        yes
Persistent:    yes
Autostart:     yes
Bridge:        virbr5
```

12.2.2.4 Starting a Network

To start an inactive network that was already defined, find its name (or unique identifier, UUID) with:

```
tux > sudo virsh net-list --inactive
Name                State    Autostart    Persistent
-----
vnet_isolated       inactive yes           yes
```

Then run:

```
tux > sudo virsh net-start vnet_isolated
Network vnet_isolated started
```

12.2.2.5 Stopping a Network

To stop an active network, find its name (or unique identifier, UUID) with:

```
tux > sudo virsh net-list --inactive
Name                State    Autostart    Persistent
-----
vnet_isolated       active   yes           yes
```

Then run:

```
tux > sudo virsh net-destroy vnet_isolated
Network vnet_isolated destroyed
```

12.2.2.6 Removing a Network

To remove the definition of an inactive network from VM Host Server permanently, run:

```
tux > sudo virsh net-undefine vnet_isolated
Network vnet_isolated has been undefined
```

13 Configuring Virtual Machines with Virtual Machine Manager

Virtual Machine Manager's *Details* view offers in-depth information about the VM Guest's complete configuration and hardware equipment. Using this view, you can also change the guest configuration or add and modify virtual hardware. To access this view, open the guest's console in Virtual Machine Manager and either choose *View > Details* from the menu, or click *Show virtual hardware details* in the toolbar.

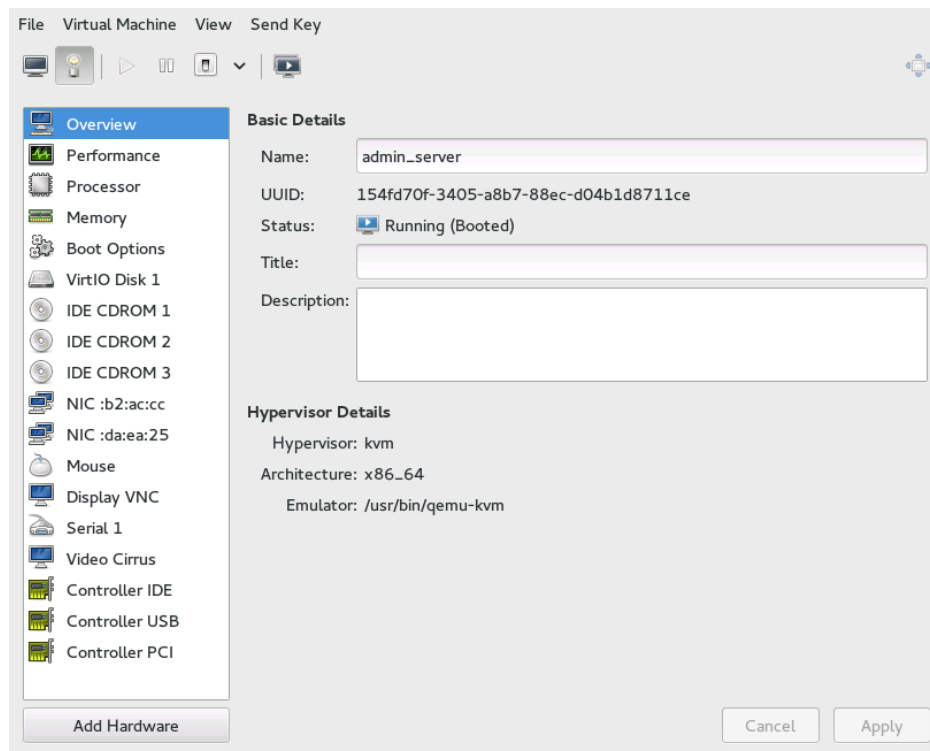


FIGURE 13.1: DETAILS VIEW OF A VM GUEST

The left panel of the window lists VM Guest overview and already installed hardware. After clicking an item in the list, you can access its detailed settings in the details view. You can change the hardware parameters to match your needs, then click *Apply* to confirm them. Some changes take effect immediately, while others need a reboot of the machine—and `virt-manager` warns you about that fact.

To remove installed hardware from a VM Guest, select the appropriate list entry in the left panel and then click *Remove* in the bottom right of the window.

To add new hardware, click *Add Hardware* below the left panel, then select the type of the hardware you want to add in the *Add New Virtual Hardware* window. Modify its parameters and confirm with *Finish*.

The following sections describe configuration options for the specific hardware type *being added*. They do not focus on modifying an existing piece of hardware as the options are identical.

13.1 Machine Setup

This section describes the setup of the virtualized processor and memory hardware. These components are vital to a VM Guest, therefore you cannot remove them. It also shows how to view the overview and performance information, and how to change boot parameters.

13.1.1 Overview

Overview shows basic details about VM Guest and the hypervisor.

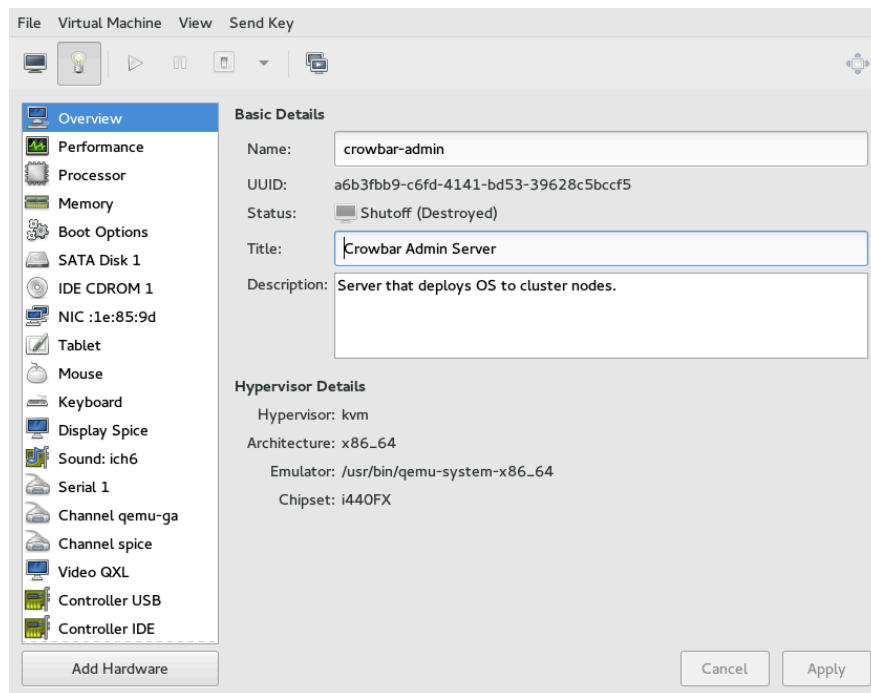


FIGURE 13.2: OVERVIEW DETAILS

Name, *Title*, and *Description* are editable and help you identify VM Guest in the *Virtual Machine Manager* list of machines.

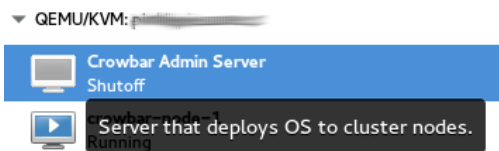


FIGURE 13.3: VM GUEST TITLE AND DESCRIPTION

UUID shows the universally unique identifier of the virtual machine, while *Status* shows its current status—*Running*, *Paused*, or *Shutoff*.

The *Hypervisor Details* section shows the hypervisor type, CPU architecture, used emulator, and chipset type. None of the hypervisor parameters can be changed.

13.1.2 Performance

Performance shows regularly updated charts of CPU and memory usage, and disk and network I/O.

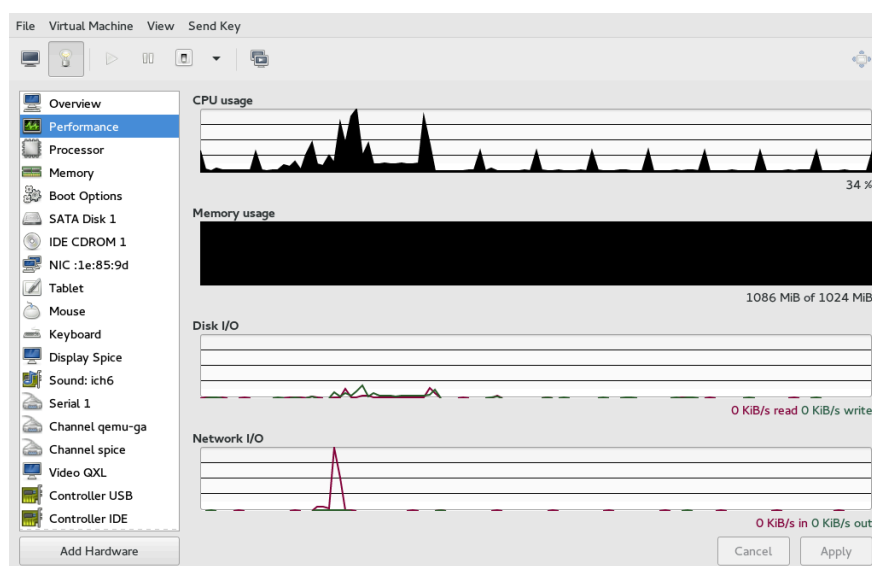


FIGURE 13.4: PERFORMANCE

Tip: Enabling Disabled Charts

Not all the charts in the *Graph* view are enabled by default. To enable these charts, go to *File > View Manager*, then select *Edit > Preferences > Polling*, and check the charts that you want to see regularly updated.

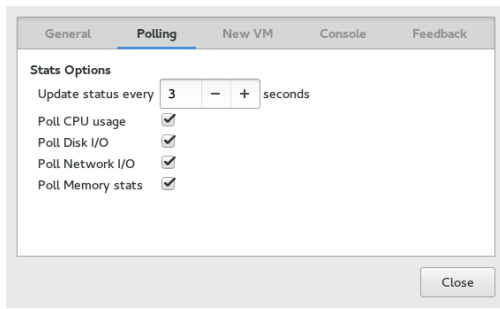


FIGURE 13.5: STATISTICS CHARTS

13.1.3 Processor

Processor includes detailed information about VM Guest processor configuration.

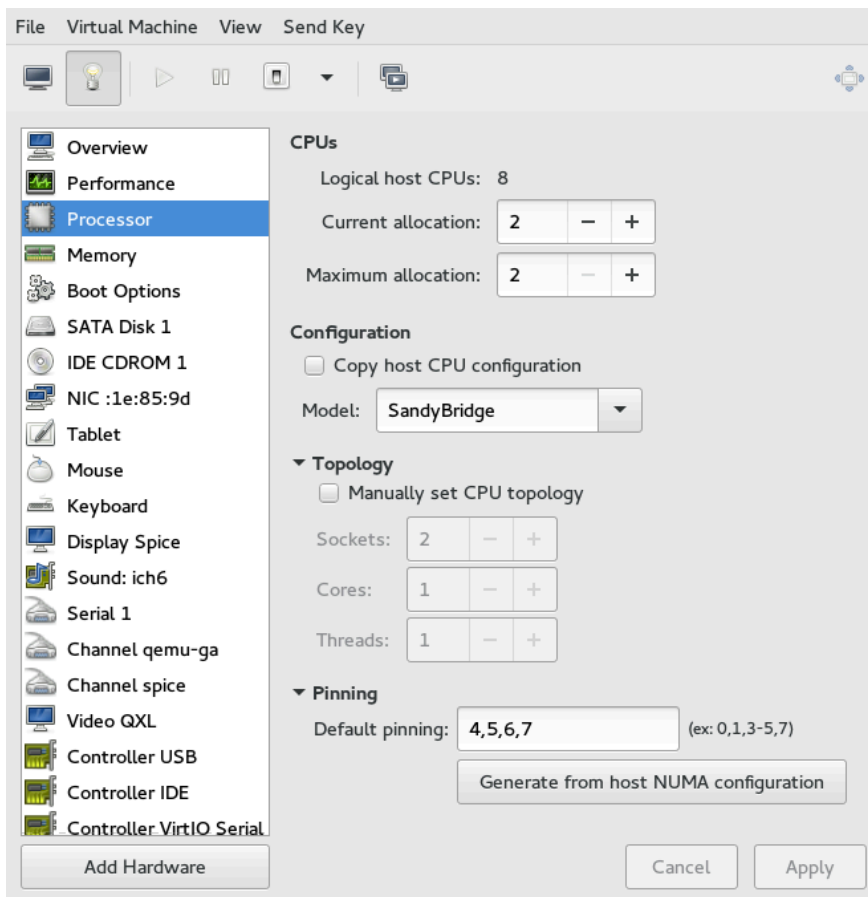


FIGURE 13.6: PROCESSOR VIEW

In the *CPUs* section, you can configure several parameters related to the number of allocated CPUs.

Logical host CPUs

The real number of CPUs installed on VM Host Server.

Current allocation

The number of currently allocated CPUs. You can hotplug more CPUs by increasing this value up to the *Maximum allocation* value.

Maximum allocation

Maximum number of allocatable CPUs for the current session. Any change to this value will take effect after the next VM Guest reboot.

The *Configuration* section lets you configure the CPU model and topology.

When activated, the *Copy host CPU configuration* option uses the host CPU model for VM Guest. Otherwise you need to specify the CPU model from the drop-down box.

After you activate *Manually set CPU topology*, you can specify a custom number of sockets, cores and threads for the CPU.

13.1.4 Memory

Memory contains information about the memory that is available to VM Guest.

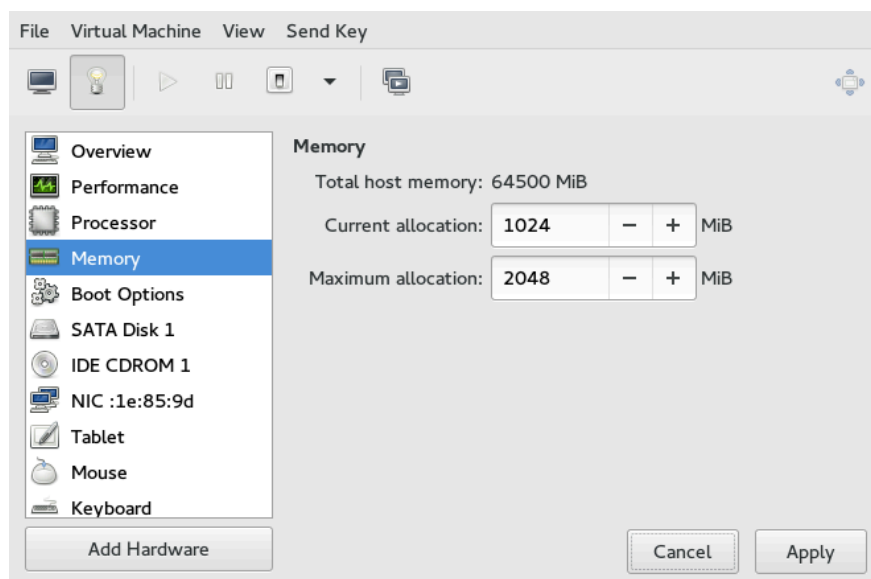


FIGURE 13.7: MEMORY VIEW

Total host memory

Total amount of memory installed on VM Host Server.

Current allocation

The amount of memory currently available to VM Guest. You can hotplug more memory by increasing this value up to the value of *Maximum allocation*.

Maximum allocation

The maximum value to which you can hotplug the currently available memory. Any change to this value will take effect after the next VM Guest reboot.

13.1.5 Boot Options

Boot Options introduces options affecting the VM Guest boot process.

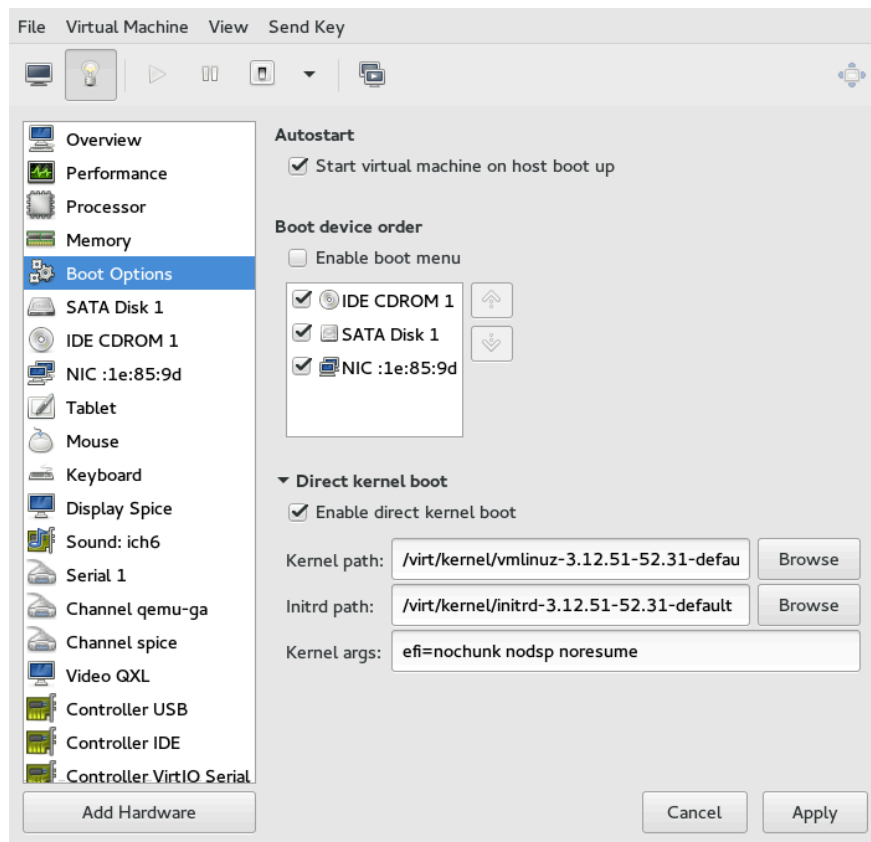


FIGURE 13.8: BOOT OPTIONS

In the *Autostart* section, you can specify whether the virtual machine should automatically start during the VM Host Server boot phase.

In the *Boot device order*, activate the devices that will be used for booting VM Guest. You can change their order with the up and down arrow buttons on the right side of the list. To choose from a list of bootable devices on VM Guest start, activate *Enable boot menu*.

To boot a different kernel than the one on the boot device, activate *Enable direct kernel boot* and specify the paths to the alternative kernel and *initrd* placed on the VM Host Server file system. You can also specify kernel arguments that will be passed to the loaded kernel.

13.2 Storage

This section gives you a detailed description of configuration options for storage devices. It includes both hard disks and removable media, such as USB or CD-ROM drives.

PROCEDURE 13.1: ADDING A NEW STORAGE DEVICE

1. Below the left panel, click *Add Hardware* to open the *Add New Virtual Hardware* window. There, select *Storage*.

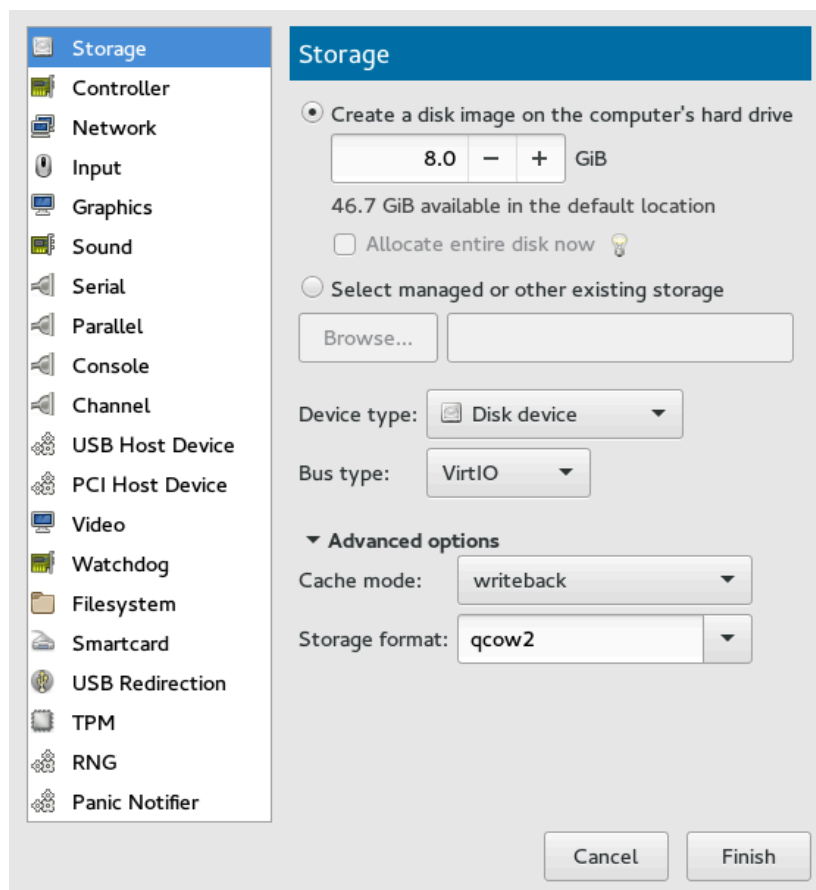


FIGURE 13.9: ADD A NEW STORAGE

2. To create a qcow2 disk image in the default location, activate *Create a disk image for the virtual machine* and specify its size in gigabytes.

To gain more control over the disk image creation, activate *Select or create custom storage* and click *Manage* to manage storage pools and images. The window *Choose Storage Volume* opens which has almost identical functionality as the *Storage* tab described in [Section 11.1, “Managing Storage with Virtual Machine Manager”](#).



Tip: Supported Storage Formats

SUSE only supports the following storage formats: raw and qcow2.

3. After you manage to create and specify the disk image file, specify the *Device type*. It can be one of the following options:
 - *Disk device*
 - *CDROM device*: Does not allow using *Create a disk image for the virtual machine*.
 - *Floppy device*: Does not allow using *Create a disk image for the virtual machine*.
 - *LUN Passthrough*: Required to use an existing SCSI storage directly without adding it into a storage pool.
4. Select the *Bus type* for your device. The list of available options depends on the device type you selected in the previous step. The types based on *VirtIO* use paravirtualized drivers.
5. In the *Advanced options* section, select the preferred *Cache mode*. For more information on cache modes, see [Chapter 15, Disk Cache Modes](#).
6. Confirm your settings with *Finish*. A new storage device appears in the left panel.

13.3 Controllers

This section focuses on adding and configuring new controllers.

PROCEDURE 13.2: ADDING A NEW CONTROLLER

1. Below the left panel, click *Add Hardware* to open the *Add New Virtual Hardware* window. There, select *Controller*.

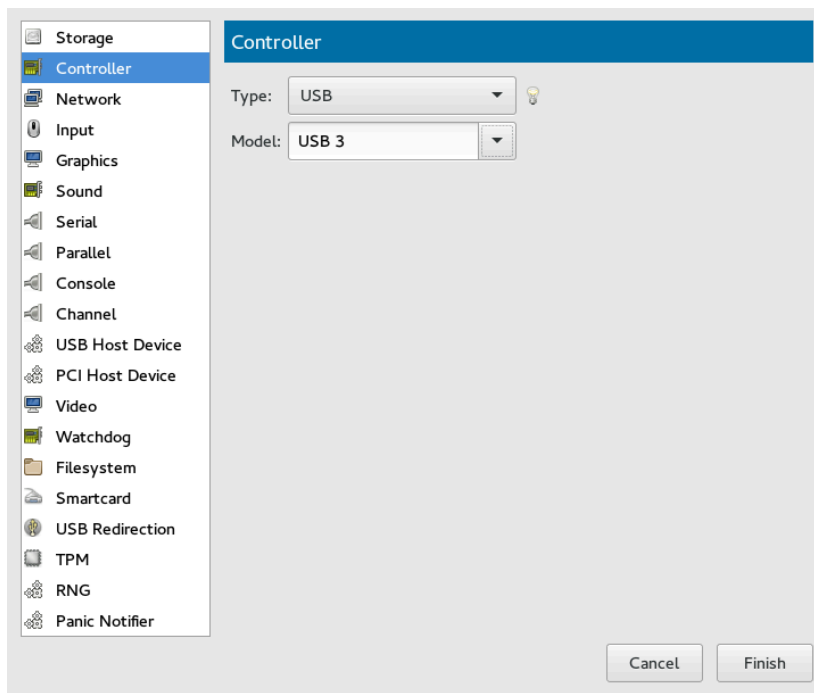


FIGURE 13.10: ADD A NEW CONTROLLER

2. Select the type of the controller. You can choose from *IDE*, *Floppy*, *SCSI*, *SATA*, *VirtIO Serial* (paravirtualized), *USB*, or *CCID* (smart card devices).
3. Optionally, in the case of a USB or SCSI controller, select a controller model.
4. Confirm your settings with *Finish*. A new controller appears in the left panel.

13.4 Networking

This section describes how to add and configure new network devices.

PROCEDURE 13.3: ADDING A NEW NETWORK DEVICE

1. Below the left panel, click *Add Hardware* to open the *Add New Virtual Hardware* window. There, select *Network*.

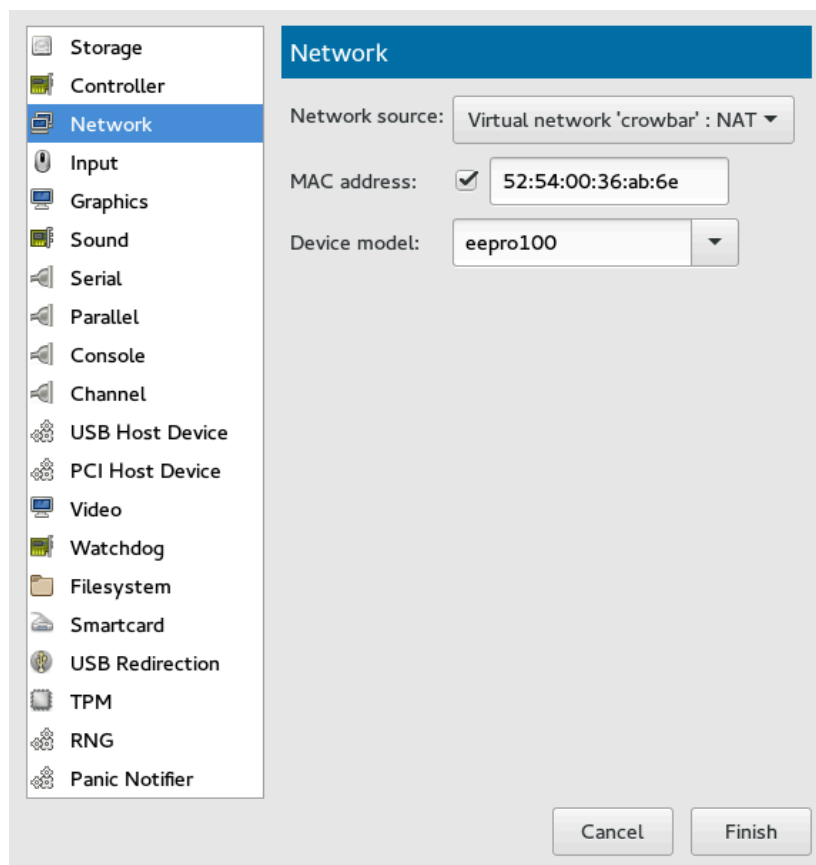


FIGURE 13.11: ADD A NEW CONTROLLER

2. From the *Network source* list, select the source for the network connection. The list includes VM Host Server's available physical network interfaces, network bridges, or network bonds. You can also assign the VM Guest to an already defined virtual network. See [Chapter 12, Managing Networks](#) for more information on setting up virtual networks with Virtual Machine Manager.
3. Specify a *MAC address* for the network device. While Virtual Machine Manager pre-fills a random value for your convenience, it is recommended to supply a MAC address appropriate for your network environment to avoid network conflicts.
4. Select a device model from the list. You can either leave the *Hypervisor default*, or specify one of *e1000*, *rtl8139*, or *virtio* models. Note that *virtio* uses paravirtualized drivers.
5. Confirm your settings with *Finish*. A new network device appears in the left panel.

13.5 Input Devices

This section focuses on adding and configuring new input devices such as mouse, keyboard, or tablet.

PROCEDURE 13.4: ADDING A NEW INPUT DEVICE

1. Below the left panel, click *Add Hardware* to open the *Add New Virtual Hardware* window. There, select *Input*.

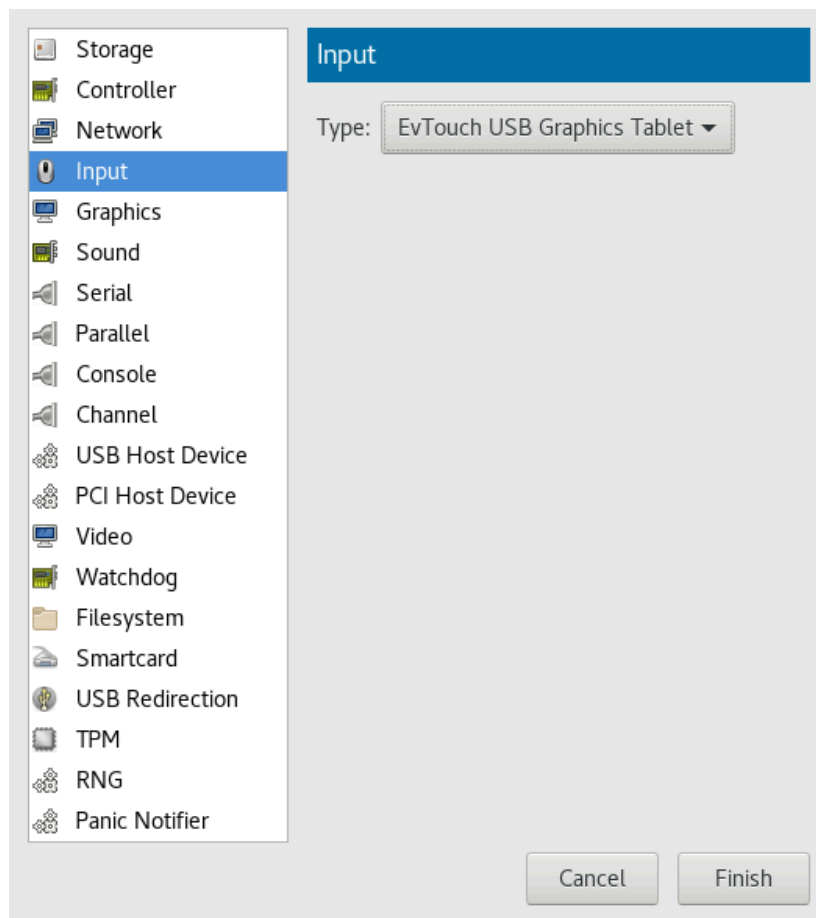


FIGURE 13.12: ADD A NEW INPUT DEVICE

2. Select a device type from the list.
3. Confirm your settings with *Finish*. A new input device appears in the left panel.



Tip: Enabling Seamless and Synchronized Mouse Pointer Movement

When you click within a VM Guest's console with the mouse, the pointer is captured by the console window and cannot be used outside the console unless it is explicitly released (by pressing `Alt-Ctrl`). To prevent the console from grabbing the key and to enable seamless pointer movement between host and guest instead, follow the instructions in *Procedure 13.4, "Adding a New Input Device"* to add an *EvTouch USB Graphics Tablet* to the VM Guest.

Adding a tablet has the additional advantage of synchronizing the mouse pointer movement between VM Host Server and VM Guest when using a graphical environment on the guest. With no tablet configured on the guest, you will often see two pointers with one dragging behind the other.

13.6 Video

This section describes how to add and configure new video devices.

PROCEDURE 13.5: ADDING A VIDEO DEVICE

1. Below the left panel, click *Add Hardware* to open the *Add New Virtual Hardware* window. There, select *Video*.

2.

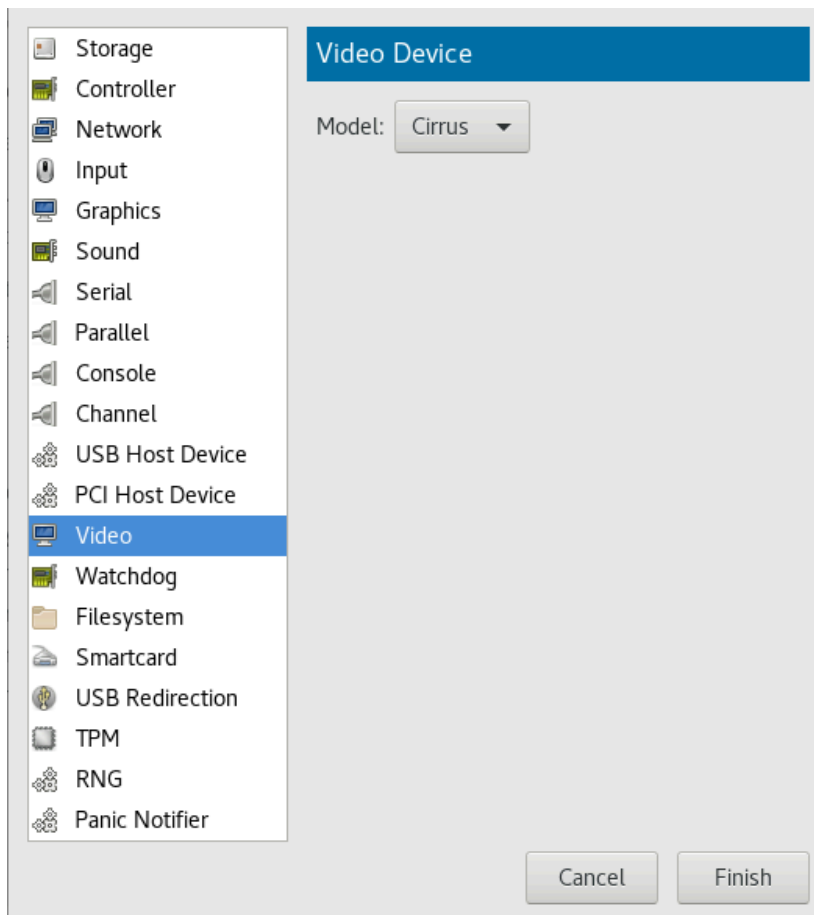


FIGURE 13.13: ADD A NEW VIDEO DEVICE

3. Select a model from the list. You can choose from:

- Cirrus
- QXL
- VGA
- Virtio
- VMVGA
- Xen



Note: Secondary Video Devices

Only *QXL* and *Virtio* can be added as secondary video devices.

4. Confirm your settings with *Finish*. A new video device appears in the left panel.

13.7 USB Redirectors

USB devices that are connected to the client machine can be redirected to the VM Guest by using *USB Redirectors*.

PROCEDURE 13.6: ADDING A USB REDIRECTOR

1. Below the left panel, click *Add Hardware* to open the *Add New Virtual Hardware* window. There, select *USB Redirection*.

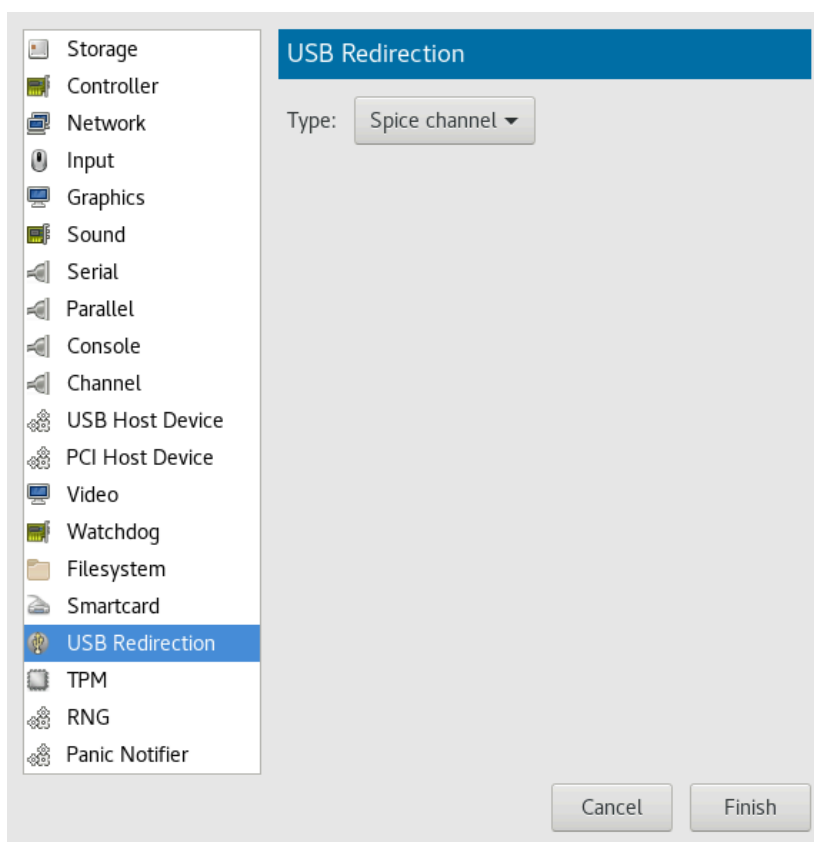


FIGURE 13.14: ADD A NEW USB REDIRECTOR

2. Select a device type from the list. You can either choose a *Spice channel* or a *TCP* redirector.
3. Confirm your settings with *Finish*. A new USB redirector appears in the left panel.

13.8 Miscellaneous

Smartcard

Smartcard functionality can be added via the *Smartcard* element. A physical USB smartcard reader can then be passed through to the VM Guest.

Watchdog

Virtual watchdog devices are also supported. They can be created via the *Watchdog* element. The model as well as the action of the device can be specified.



Tip: Requirements for Virtual Watchdog Devices

QA virtual watchdog devices require a specific driver and daemon to be installed in the VM Guest. Otherwise the virtual watchdog device does not work.

TPM

You can use the Host TPM device in the VM Guest by adding TPM functionality via the *TPM* element.



Tip: Virtual TPMs

The Host TPM can only be used in one VM Guest at a time.

13.9 Adding a CD/DVD-ROM Device with Virtual Machine Manager

KVM supports CD or DVD-ROMs in VM Guest either by directly accessing a physical drive on the VM Host Server or by accessing ISO images. To create an ISO image from an existing CD or DVD, use `dd`:

```
tux > sudo dd if=/dev/CD_DVD_DEVICE of=my_distro.iso bs=2048
```

To add a CD/DVD-ROM device to your VM Guest, proceed as follows:

1. Double-click a VM Guest entry in the Virtual Machine Manager to open its console and switch to the *Details* view with *View > Details*.

2. Click *Add Hardware* and choose *Storage* in the pop-up window.
3. Change the *Device Type* to *IDE CDROM*.
4. Select *Select or create custom storage*.
 - a. To assign the device to a physical medium, enter the path to the VM Host Server's CD/DVD-ROM device (for example, `/dev/cdrom`) next to *Manage*. Alternatively, use *Manage* to open a file browser and then click *Browse Local* to select the device. Assigning the device to a physical medium is only possible when the Virtual Machine Manager was started on the VM Host Server.
 - b. To assign the device to an existing image, click *Manage* to choose an image from a storage pool. If the Virtual Machine Manager was started on the VM Host Server, alternatively choose an image from another location on the file system by clicking *Browse Local*. Select an image and close the file browser with *Choose Volume*.
5. Save the new virtualized device with *Finish*.
6. Reboot the VM Guest to make the new device available. For more information, see [Section 13.11, “Ejecting and Changing Floppy or CD/DVD-ROM Media with Virtual Machine Manager”](#).

13.10 Adding a Floppy Device with Virtual Machine Manager

Currently KVM only supports the use of floppy disk images—using a physical floppy drive is not supported. Create a floppy disk image from an existing floppy using `dd`:

```
tux > sudo dd if=/dev/fd0 of=/var/lib/libvirt/images/floppy.img
```

To create an empty floppy disk image use one of the following commands:

Raw Image

```
tux > sudo dd if=/dev/zero of=/var/lib/libvirt/images/floppy.img bs=512 count=2880
```

FAT Formatted Image

```
tux > sudo mkfs.msdos -C /var/lib/libvirt/images/floppy.img 1440
```

To add a floppy device to your VM Guest, proceed as follows:

1. Double-click a VM Guest entry in the Virtual Machine Manager to open its console and switch to the *Details* view with *View > Details*.
2. Click *Add Hardware* and choose *Storage* in the pop-up window.
3. Change the *Device Type* to *Floppy Disk*.
4. Choose *Select or create custom storage* and click *Manage* to choose an existing image from a storage pool. If Virtual Machine Manager was started on the VM Host Server, alternatively choose an image from another location on the file system by clicking *Browse Local*. Select an image and close the file browser with *Choose Volume*.
5. Save the new virtualized device with *Finish*.
6. Reboot the VM Guest to make the new device available. For more information, see [Section 13.11, “Ejecting and Changing Floppy or CD/DVD-ROM Media with Virtual Machine Manager”](#).

13.11 Ejecting and Changing Floppy or CD/DVD-ROM Media with Virtual Machine Manager

Whether you are using the VM Host Server's physical CD/DVD-ROM device or an ISO/floppy image: Before you can change the media or image of an existing device in the VM Guest, you first need to disconnect the media from the guest.

1. Double-click a VM Guest entry in the Virtual Machine Manager to open its console and switch to the *Details* view with *View > Details*.
2. Choose the Floppy or CD/DVD-ROM device and “eject” the medium by clicking *Disconnect*.
3. To “insert” a new medium, click *Connect*.
 - a. If using the VM Host Server's physical CD/DVD-ROM device, first change the media in the device (this may require unmounting it on the VM Host Server before it can be ejected). Then choose *CD-ROM or DVD* and select the device from the drop-down box.
 - b. If you are using an ISO image, choose *ISO image Location* and select an image by clicking *Manage*. When connecting from a remote host, you may only choose images from existing storage pools.

4. Click *OK* to finish. The new media can now be accessed in the VM Guest.

13.12 Assigning a Host PCI Device to a VM Guest

You can directly assign host-PCI devices to guests (PCI pass-through). When the PCI device is assigned to one VM Guest, it cannot be used on the host or by another VM Guest unless it is re-assigned. A prerequisite for this feature is a VM Host Server configuration as described in *Important: Requirements for VFIO and SR-IOV*.

13.12.1 Adding a PCI Device with Virtual Machine Manager

The following procedure describes how to add a PCI device to a VM Guest using Virtual Machine Manager:

1. Double-click a VM Guest entry in the Virtual Machine Manager to open its console and switch to the *Details* view with *View > Details*.
2. Click *Add Hardware* and choose the *PCI Host Device* category in the left panel. A list of available PCI devices appears in the right part of the window.

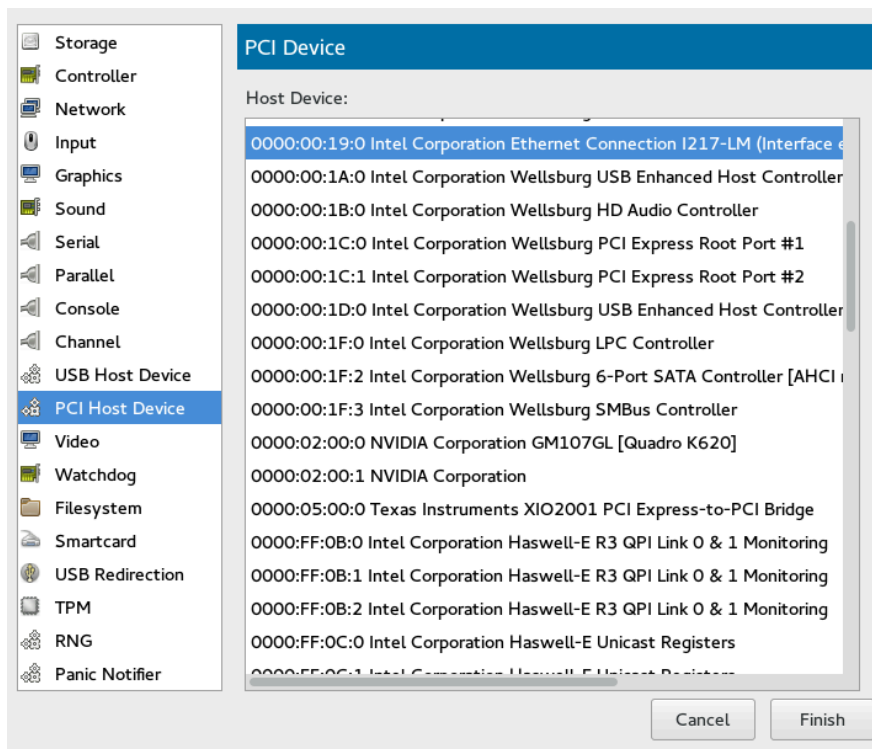


FIGURE 13.15: ADDING A PCI DEVICE

- From the list of available PCI devices, choose the one you want to pass to the guest. Confirm with *Finish*.



Tip: Assigning a PCI Device Requires a VM Guest Shutdown

Although it is possible to assign a PCI device to a running VM Guest as described above, the device will not become available until you shut down the VM Guest and reboot it afterward.

13.13 Assigning a Host USB Device to a VM Guest

Analogous to assigning host PCI devices (see *Section 13.12, "Assigning a Host PCI Device to a VM Guest"*), you can directly assign host USB devices to guests. When the USB device is assigned to one VM Guest, it cannot be used on the host or by another VM Guest unless it is re-assigned.

13.13.1 Adding a USB Device with Virtual Machine Manager

To assign a host USB device to VM Guest using Virtual Machine Manager, follow these steps:

1. Double-click a VM Guest entry in the Virtual Machine Manager to open its console and switch to the *Details* view with *View > Details*.
2. Click *Add Hardware* and choose the *USB Host Device* category in the left panel. A list of available USB devices appears in the right part of the window.

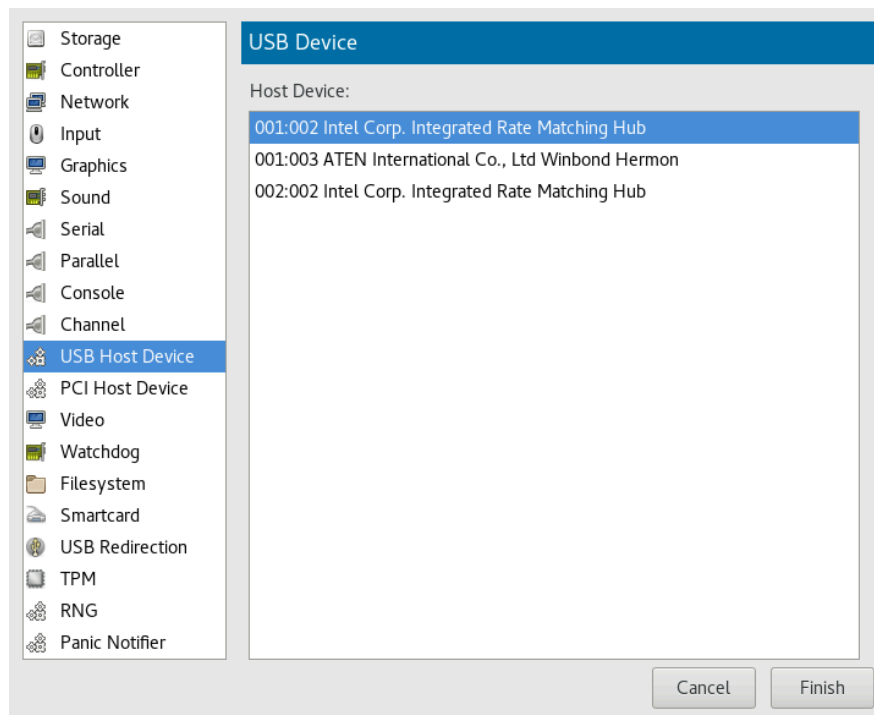


FIGURE 13.16: ADDING A USB DEVICE

3. From the list of available USB devices, choose the one you want to pass to the guest. Confirm with *Finish*. The new USB device appears in the left pane of the *Details* view.



Tip: USB Device Removal

To remove the host USB device assignment, click it in the left pane of the *Details* view and confirm with *Remove*.

14 Configuring Virtual Machines with `virsh`

You can use `virsh` to configure virtual machines (VM) on the command line as an alternative to using the Virtual Machine Manager. With `virsh`, you can control the state of a VM, edit the configuration of a VM or even migrate a VM to another host. The following sections describe how to manage VMs by using `virsh`.

14.1 Editing the VM Configuration

The configuration of a VM is stored in an XML file in `/etc/libvirt/qemu/` and looks like this:

EXAMPLE 14.1: EXAMPLE XML CONFIGURATION FILE

```
<domain type='kvm'>
  <name>sles15</name>
  <uuid>ab953e2f-9d16-4955-bb43-1178230ee625</uuid>
  <memory unit='KiB'>2097152</memory>
  <currentMemory unit='KiB'>2097152</currentMemory>
  <vcpu placement='static'>2</vcpu>
  <os>
    <type arch='x86_64' machine='pc-i440fx-2.11'>hvm</type>
  </os>
  <features>...</features>
  <cpu mode='custom' match='exact' check='partial'>
    <model fallback='allow'>Skylake-Client-IBRS</model>
  </cpu>
  <clock>...</clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>destroy</on_crash>
  <pm>
    <suspend-to-mem enabled='no' />
    <suspend-to-disk enabled='no' />
  </pm>
  <devices>
    <emulator>/usr/bin/qemu-system-x86_64</emulator>
    <disk type='file' device='disk'>...</disk>
  </devices>
  ...
</domain>
```

If you want to edit the configuration of a VM Guest, check if it is offline:

```
tux > sudo virsh list --inactive
```

If your VM Guest is in this list, you can safely edit its configuration:

```
tux > sudo virsh edit NAME_OF_VM_GUEST
```

Before saving the changes, **virsh** validates your input against a RelaxNG schema.

14.2 Changing the Machine Type

When installing with the **virt-install** tool, the machine type for a VM Guest is *pc-i440fx* by default. The machine type is stored in the VM Guest's configuration file in the `type` element:

```
<type arch='x86_64' machine='pc-i440fx-2.3'>hvm</type>
```

As an example, the following procedure shows how to change this value to the machine type `q35`. The value `q35` is an Intel* chipset and includes *PCIe*, supports up to 12 USB ports, and has support for *SATA* and *IOMMU*.

PROCEDURE 14.1: CHANGING MACHINE TYPE

1. Check whether your VM Guest is inactive:

```
tux > sudo virsh list --inactive
Id      Name                               State
-----
-       sles15                             shut off
```

2. Edit the configuration for this VM Guest:

```
tux > sudo virsh edit sles15
```

3. Replace the value of the `machine` attribute with `pc-q35-2.0` :

```
<type arch='x86_64' machine='pc-q35-2.0'>hvm</type>
```

4. Restart the VM Guest:

```
tux > sudo virsh start sles15
```


5. Check if the machine type has changed. Log in to the VM Guest and run the following command:

```
tux > sudo dmidecode | grep Product
Product Name: Standard PC (Q35 + ICH9, 2009)
```



Tip: Machine Type Update Recommendations

Whenever the QEMU version on the host system is upgraded (for example, when upgrading the VM Host Server to a new service pack), upgrade the machine type of the VM Guests to the latest available version. To check, use the command `qemu-system-x86_64 -M help` on the VM Host Server.

The default machine type `pc-i440fx`, for example, is regularly updated. If your VM Guest still runs with a machine type of `pc-i440fx-1.X`, we strongly recommend an update to `pc-i440fx-2.X`. This allows taking advantage of the most recent updates and corrections in machine definitions, and ensures better future compatibility.

14.3 Configuring Hypervisor Features

libvirt automatically enables a default set of hypervisor features that are sufficient in most circumstances, but also allows enabling and disabling features as needed. As an example, Xen does not support enabling PCI passthrough by default. It must be enabled with the `passthrough` setting. Hypervisor features can be configured with `virsh`. Look for the `<features>` element in the VM Guest's configuration file and adjust the various features as required. Continuing with the Xen passthrough example:

```
tux > sudo virsh edit sle15sp1
<features>
  <xen>
    <passthrough/>
  </xen>
</features>
```

Save your changes and restart the VM Guest.

See the *Hypervisor features* section of the *libvirt Domain XML format* manual at <https://libvirt.org/formatdomain.html#elementsFeatures> for more information.

14.4 Configuring CPU Allocation

The number of allocated CPUs is stored in the VM Guest's XML configuration file in `/etc/libvirt/qemu/` in the `vcpu` element:

```
<vcpu placement='static'>1</vcpu>
```

In this example, the VM Guest has only one allocated CPU. The following procedure shows how to change the number of allocated CPUs for the VM Guest:

1. Check whether your VM Guest is inactive:

```
tux > sudo virsh list --inactive
Id      Name                               State
-----
-      sles15                             shut off
```

2. Edit the configuration for an existing VM Guest:

```
tux > sudo virsh edit sles15
```

3. Change the number of allocated CPUs:

```
<vcpu placement='static'>2</vcpu>
```

4. Restart the VM Guest:

```
tux > sudo virsh start sles15
```

5. Check if the number of CPUs in the VM has changed.

```
tux > sudo virsh vcpuinfo sles15
VCPU:      0
CPU:       N/A
State:     N/A
CPU time   N/A
CPU Affinity: yy

VCPU:      1
CPU:       N/A
State:     N/A
CPU time   N/A
CPU Affinity: yy
```

14.5 Changing Boot Options

The boot menu of the VM Guest can be found in the `os` element and usually looks like this:

```
<os>
  <type>hvm</type>
  <loader>readonly='yes' secure='no' type='rom'>/usr/lib/xen/boot/hvmloder</loader>
  <nvram template='/usr/share/OVMF/OVMF_VARS.fd'>/var/lib/libvirt/nvram/guest_VARS.fd</
nvram>
  <boot dev='hd' />
  <boot dev='cdrom' />
  <bootmenu enable='yes' timeout='3000' />
  <smbios mode='sysinfo' />
  <bios useserial='yes' rebootTimeout='0' />
</os>
```

In this example, two devices are available, `hd` and `cdrom`. The configuration also reflects the actual boot order, so the `cdrom` comes before the `hd`.

14.5.1 Changing Boot Order

The VM Guest's boot order is represented through the order of devices in the XML configuration file. As the devices are interchangeable, it is possible to change the boot order of the VM Guest.

1. Open the VM Guest's XML configuration.

```
tux > sudo virsh edit sles15
```

2. Change the sequence of the bootable devices.

```
...
<boot dev='cdrom' />
<boot dev='hd' />
...
```

3. Check if the boot order was changed successfully by looking at the boot menu in the BIOS of the VM Guest.

14.5.2 Using Direct Kernel Boot

Direct Kernel Boot allows you to boot from a kernel and initrd stored on the host. Set the path to both files in the `kernel` and `initrd` elements:

```
<os>
...
<kernel>/root/f8-i386-vmlinuz</kernel>
<initrd>/root/f8-i386-initrd</initrd>
...
</os>
```

To enable Direct Kernel Boot:

1. Open the VM Guest's XML configuration:

```
tux > sudo virsh edit sles15
```

2. Inside the `os` element, add a `kernel` element and the path to the kernel file on the host:

```
...
<kernel>/root/f8-i386-vmlinuz</kernel>
...
```

3. Add an `initrd` element and the path to the initrd file on the host:

```
...
<initrd>/root/f8-i386-initrd</initrd>
...
```

4. Start your VM to boot from the new kernel:

```
tux > sudo virsh start sles15
```

14.6 Configuring Memory Allocation

The amount of memory allocated for the VM Guest can also be configured with `virsh`. It is stored in the `memory` element. Follow these steps:

1. Open the VM Guest's XML configuration:

```
tux > sudo virsh edit sles15
```

2. Search for the `memory` element and set the amount of allocated RAM:

```
...
<memory unit='KiB'>524288</memory>
...
```

3. Check the amount of allocated RAM in your VM by running:

```
tux > cat /proc/meminfo
```

14.7 Adding a PCI Device

To assign a PCI device to VM Guest with `virsh`, follow these steps:

1. Identify the host PCI device to assign to the VM Guest. In the following example, we are assigning a DEC network card to the guest:

```
tux > sudo lspci -nn
[...]
03:07.0 Ethernet controller [0200]: Digital Equipment Corporation DECchip \
21140 [FasterNet] [1011:0009] (rev 22)
[...]
```

Write down the device ID (`03:07.0` in this case).

2. Gather detailed information about the device using `virsh nodedev-dumpxml ID`. To get the `ID`, replace the colon and the period in the device ID (`03:07.0`) with underscores. Prefix the result with “`pci_0000_`”: `pci_0000_03_07_0`.

```
tux > sudo virsh nodedev-dumpxml pci_0000_03_07_0
<device>
  <name>pci_0000_03_07_0</name>
  <path>/sys/devices/pci0000:00/0000:00:14.4/0000:03:07.0</path>
  <parent>pci_0000_00_14_4</parent>
  <driver>
    <name>tulip</name>
  </driver>
  <capability type='pci'>
    <domain>0</domain>
    <bus>3</bus>
    <slot>7</slot>
    <function>0</function>
    <product id='0x0009'>DECchip 21140 [FasterNet]</product>
```

```
<vendor id='0x1011'>Digital Equipment Corporation</vendor>
<numa node='0' />
</capability>
</device>
```

Write down the values for domain, bus, and function (see the previous XML code printed in bold).

3. Detach the device from the host system prior to attaching it to the VM Guest:

```
tux > sudo virsh nodedev-detach pci_0000_03_07_0
Device pci_0000_03_07_0 detached
```

Tip: Multi-Function PCI Devices

When using a multi-function PCI device that does not support FLR (function level reset) or PM (power management) reset, you need to detach all its functions from the VM Host Server. The whole device must be reset for security reasons. `libvirt` will refuse to assign the device if one of its functions is still in use by the VM Host Server or another VM Guest.

4. Convert the domain, bus, slot, and function value from decimal to hexadecimal. In our example, domain = 0, bus = 3, slot = 7, and function = 0. Ensure that the values are inserted in the right order:

```
tux > printf "<address domain='0x%x' bus='0x%x' slot='0x%x' function='0x%x' />\n" 0 3
7 0
```

This results in:

```
<address domain='0x0' bus='0x3' slot='0x7' function='0x0' />
```

5. Run `virsh edit` on your domain, and add the following device entry in the `<devices>` section using the result from the previous step:

```
<hostdev mode='subsystem' type='pci' managed='yes'>
  <source>
    <address domain='0x0' bus='0x03' slot='0x07' function='0x0' />
  </source>
</hostdev>
```



Tip: managed Compared to unmanaged

`libvirt` recognizes two modes for handling PCI devices: they can be either managed or unmanaged. In the managed case, `libvirt` handles all details of unbinding the device from the existing driver if needed, resetting the device, binding it to `vfio-pci` before starting the domain, etc. When the domain is terminated or the device is removed from the domain, `libvirt` unbinds from `vfio-pci` and rebinds to the original driver in the case of a managed device. If the device is unmanaged, the user must ensure all of these management aspects of the device are done before assigning it to a domain, and after the device is no longer used by the domain.

In the example above, the `managed='yes'` option means that the device is managed. To switch the device mode to unmanaged, set `managed='no'` in the listing above. If you do so, you need to take care of the related driver with the `virsh nodedev-detach` and `virsh nodedev-reattach` commands. Prior to starting the VM Guest you need to detach the device from the host by running `virsh nodedev-detach pci_0000_03_07_0`. In case the VM Guest is not running, you can make the device available for the host by running `virsh nodedev-reattach pci_0000_03_07_0`.

6. Shut down the VM Guest and disable SELinux if it is running on the host.

```
tux > sudo setsebool -P virt_use_sysfs 1
```

7. Start your VM Guest to make the assigned PCI device available:

```
tux > sudo virsh start sles15
```

14.8 Adding a USB Device

To assign a USB device to VM Guest using `virsh`, follow these steps:

1. Identify the host USB device to assign to the VM Guest:

```
tux > sudo lsusb
[...]
Bus 001 Device 003: ID 0557:2221 ATEN International Co., Ltd Winbond Hermon
[...]
```

Write down the vendor and product IDs. In our example, the vendor ID is 0557 and the product ID is 2221.

2. Run `virsh edit` on your domain, and add the following device entry in the `<devices>` section using the values from the previous step:

```
<hostdev mode='subsystem' type='usb'>
  <source startupPolicy='optional'>
    <vendor id='0557' />
    <product id='2221' />
  </source>
</hostdev>
```



Tip: Vendor/Product or Device's Address

Instead of defining the host device with `<vendor/>` and `<product/>` IDs, you can use the `<address/>` element as described for host PCI devices in [Section 14.7](#), “Adding a PCI Device”.

3. Shut down the VM Guest and disable SELinux if it is running on the host:

```
tux > sudo setsebool -P virt_use_sysfs 1
```

4. Start your VM Guest to make the assigned PCI device available:

```
tux > sudo virsh start sles15
```

14.9 Adding SR-IOV Devices

Single Root I/O Virtualization (*SR-IOV*) capable *PCIe* devices can replicate their resources, so they appear to be multiple devices. Each of these “pseudo-devices” can be assigned to a VM Guest.

SR-IOV is an industry specification that was created by the Peripheral Component Interconnect Special Interest Group (PCI-SIG) consortium. It introduces physical functions (PF) and virtual functions (VF). PFs are full *PCIe* functions used to manage and configure the device. PFs also can move data. VFs lack the configuration and management part—they only can move data and a reduced set of configuration functions. As VFs do not have all *PCIe* functions, the host operating system or the *Hypervisor* must support *SR-IOV* to be able to access and initialize VFs.

The theoretical maximum for VFs is 256 per device (consequently the maximum for a dual-port Ethernet card would be 512). In practice this maximum is much lower, since each VF consumes resources.

14.9.1 Requirements

The following requirements must be met to use *SR-IOV*:

- An *SR-IOV*-capable network card (as of , only network cards support *SR-IOV*)
- An AMD64/Intel 64 host supporting hardware virtualization (AMD-V or Intel VT-x)
- A chipset that supports device assignment (AMD-Vi or Intel *VT-d*)
- `libvirt` 0.9.10 or better
- *SR-IOV* drivers must be loaded and configured on the host system
- A host configuration that meets the requirements listed at *Important: Requirements for VFIO and SR-IOV*
- A list of the PCI addresses of the VF(s) that will be assigned to VM Guests



Tip: Checking if a Device is SR-IOV-Capable

The information whether a device is SR-IOV-capable can be obtained from its PCI descriptor by running `lspci`. A device that supports *SR-IOV* reports a capability similar to the following:

```
Capabilities: [160 v1] Single Root I/O Virtualization (SR-IOV)
```



Note: Adding an SR-IOV Device at VM Guest Creation

Before adding an SR-IOV device to a VM Guest when initially setting it up, the VM Host Server already needs to be configured as described in *Section 14.9.2, "Loading and Configuring the SR-IOV Host Drivers"*.

14.9.2 Loading and Configuring the SR-IOV Host Drivers

To access and initialize VFs, an SR-IOV-capable driver needs to be loaded on the host system.

1. Before loading the driver, make sure the card is properly detected by running `lspci`. The following example shows the `lspci` output for the dual-port Intel 82576NS network card:

```
tux > sudo /sbin/lspci | grep 82576
01:00.0 Ethernet controller: Intel Corporation 82576NS Gigabit Network Connection
(rev 01)
01:00.1 Ethernet controller: Intel Corporation 82576NS Gigabit Network Connection
(rev 01)
04:00.0 Ethernet controller: Intel Corporation 82576NS Gigabit Network Connection
(rev 01)
04:00.1 Ethernet controller: Intel Corporation 82576NS Gigabit Network Connection
(rev 01)
```

In case the card is not detected, it is likely that the hardware virtualization support in the BIOS/EFI has not been enabled. To check if hardware virtualization support is enabled, look at the settings in the host's BIOS.

2. Check whether the `SR-IOV` driver is already loaded by running `lsmod`. In the following example a check for the `igb` driver (for the Intel 82576NS network card) returns a result. That means the driver is already loaded. If the command returns nothing, the driver is not loaded.

```
tux > sudo /sbin/lsmod | egrep "^igb "
igb                185649  0
```

3. Skip the following step if the driver is already loaded. If the `SR-IOV` driver is not yet loaded, the non-`SR-IOV` driver needs to be removed first, before loading the new driver. Use `rmmod` to unload a driver. The following example unloads the non-`SR-IOV` driver for the Intel 82576NS network card:

```
tux > sudo /sbin/rmmod igbvf
```

4. Load the `SR-IOV` driver subsequently using the `modprobe` command—the VF parameter (`max_vfs`) is mandatory:

```
tux > sudo /sbin/modprobe igb max_vfs=8
```

As an alternative, you can also load the driver via `SYSFS`:

1. Find the PCI ID of the physical NIC by listing Ethernet devices:

```
tux > sudo lspci | grep Eth
```

```
06:00.0 Ethernet controller: Emulex Corporation OneConnect NIC (Skyhawk) (rev 10)
06:00.1 Ethernet controller: Emulex Corporation OneConnect NIC (Skyhawk) (rev 10)
```

2. To enable VFs, echo the number of desired VFs to load to the `sriov_numvfs` parameter:

```
tux > sudo echo 1 > /sys/bus/pci/devices/0000:06:00.1/sriov_numvfs
```

3. Verify that the VF NIC was loaded:

```
tux > sudo lspci | grep Eth
06:00.0 Ethernet controller: Emulex Corporation OneConnect NIC (Skyhawk) (rev 10)
06:00.1 Ethernet controller: Emulex Corporation OneConnect NIC (Skyhawk) (rev 10)
06:08.0 Ethernet controller: Emulex Corporation OneConnect NIC (Skyhawk) (rev 10)
```

4. Obtain the maximum number of VFs available:

```
tux > sudo lspci -vvv -s 06:00.1 | grep 'Initial VFs'
                Initial VFs: 32, Total VFs: 32, Number of VFs: 0,
Function Dependency Link: 01
```

5. Create a `/etc/systemd/system/before.service` file which loads VF via SYSFS on boot:

```
[Unit]
Before=
[Service]
Type=oneshot
RemainAfterExit=true
ExecStart=/bin/bash -c "echo 1 > /sys/bus/pci/devices/0000:06:00.1/sriov_numvfs"
# 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
```

6. Prior to starting the VM, it is required to create another service file (`after-local.service`) pointing to the `/etc/init.d/after.local` script that detaches the NIC. Otherwise the VM would fail to start:

```
[Unit]
Description=/etc/init.d/after.local Compatibility
After=libvirtd.service
Requires=libvirtd.service
[Service]
Type=oneshot
```

```
ExecStart=/etc/init.d/after.local
RemainAfterExit=true

[Install]
WantedBy=multi-user.target
```

7. Copy it to /etc/systemd/system.

```
#!/bin/sh
# ...
virsh nodedev-detach pci_0000_06_08_0
```

Save it as /etc/init.d/after.local.

8. Reboot the machine and check if the SR-IOV driver is loaded by re-running the lspci command from the first step of this procedure. If the SR-IOV driver was loaded successfully you should see additional lines for the VFs:

```
01:00.0 Ethernet controller: Intel Corporation 82576NS Gigabit Network Connection
(rev 01)
01:00.1 Ethernet controller: Intel Corporation 82576NS Gigabit Network Connection
(rev 01)
01:10.0 Ethernet controller: Intel Corporation 82576 Virtual Function (rev 01)
01:10.1 Ethernet controller: Intel Corporation 82576 Virtual Function (rev 01)
01:10.2 Ethernet controller: Intel Corporation 82576 Virtual Function (rev 01)
[...]
04:00.0 Ethernet controller: Intel Corporation 82576NS Gigabit Network Connection
(rev 01)
04:00.1 Ethernet controller: Intel Corporation 82576NS Gigabit Network Connection
(rev 01)
04:10.0 Ethernet controller: Intel Corporation 82576 Virtual Function (rev 01)
04:10.1 Ethernet controller: Intel Corporation 82576 Virtual Function (rev 01)
04:10.2 Ethernet controller: Intel Corporation 82576 Virtual Function (rev 01)
[...]
```

14.9.3 Adding a VF Network Device to a VM Guest

When the *SR-IOV* hardware is properly set up on the VM Host Server, you can add VFs to VM Guests. To do so, you need to collect some data first.

PROCEDURE 14.2: ADDING A VF NETWORK DEVICE TO AN EXISTING VM GUEST

The following procedure uses example data. Make sure to replace it by appropriate data from your setup.

1. Use the `virsh nodedev-list` command to get the PCI address of the VF you want to assign and its corresponding PF. Numerical values from the `lspci` output shown in [Section 14.9.2, "Loading and Configuring the SR-IOV Host Drivers"](#) (for example `01:00.0` or `04:00.1`) are transformed by adding the prefix "pci_0000_" and by replacing colons and dots with underscores. So a PCI ID listed as "04:00.0" by `lspci` is listed as "pci_0000_04_00_0" by `virsh`. The following example lists the PCI IDs for the second port of the Intel 82576NS network card:

```
tux > sudo virsh nodedev-list | grep 0000_04_
pci_0000_04_00_0
pci_0000_04_00_1
pci_0000_04_10_0
pci_0000_04_10_1
pci_0000_04_10_2
pci_0000_04_10_3
pci_0000_04_10_4
pci_0000_04_10_5
pci_0000_04_10_6
pci_0000_04_10_7
pci_0000_04_11_0
pci_0000_04_11_1
pci_0000_04_11_2
pci_0000_04_11_3
pci_0000_04_11_4
pci_0000_04_11_5
```

The first two entries represent the **PFs**, whereas the other entries represent the **VFs**.

2. Run the following `virsh nodedev-dumpxml` command on the PCI ID of the VF you want to add:

```
tux > sudo virsh nodedev-dumpxml pci_0000_04_10_0
<device>
  <name>pci_0000_04_10_0</name>
  <parent>pci_0000_00_02_0</parent>
  <capability type='pci'>
    <domain>0</domain>
    <bus>4</bus>
    <slot>16</slot>
    <function>0</function>
    <product id='0x10ca'>82576 Virtual Function</product>
    <vendor id='0x8086'>Intel Corporation</vendor>
    <capability type='phys_function'>
      <address domain='0x0000' bus='0x04' slot='0x00' function='0x0' />
    </capability>
```

```
</capability>
</device>
```

The following data is needed for the next step:

- <domain>0</domain>
- <bus>4</bus>
- <slot>16</slot>
- <function>0</function>

3. Create a temporary XML file (for example /tmp/vf-interface.xml containing the data necessary to add a VF network device to an existing VM Guest. The minimal content of the file needs to look like the following:

```
<interface type='hostdev'>❶
  <source>
    <address type='pci' domain='0' bus='11' slot='16' function='0'2/>❷
  </source>
</interface>
```

- ❶ VFs do not get a fixed MAC address; it changes every time the host reboots. When adding network devices the “traditional” way with hostdev, it would require to reconfigure the VM Guest's network device after each reboot of the host, because of the MAC address change. To avoid this kind of problem, libvirt introduced the hostdev value, which sets up network-specific data *before* assigning the device.
 - ❷ Specify the data you acquired in the previous step here.
4. In case a device is already attached to the host, it cannot be attached to a VM Guest. To make it available for guests, detach it from the host first:

```
tux > sudo virsh nodedev-detach pci_0000_04_10_0
```

5. Add the VF interface to an existing VM Guest:

```
tux > sudo virsh attach-device GUEST /tmp/vf-interface.xml --OPTION
```

GUEST needs to be replaced by the domain name, ID or UUID of the VM Guest. --OPTION can be one of the following:

--persistent

This option will always add the device to the domain's persistent XML. In addition, if the domain is running, it will be hotplugged.

--config

This option will only affect the persistent XML, even if the domain is running. The device will only show up in the VM Guest on next boot.

--live

This option will only affect a running domain. If the domain is inactive, the operation will fail. The device is not persisted in the XML and will not be available in the VM Guest on next boot.

--current

This option affects the current state of the domain. If the domain is inactive, the device is added to the persistent XML and will be available on next boot. If the domain is active, the device is hotplugged but not added to the persistent XML.

6. To detach a VF interface, use the **virsh detach-device** command, which also takes the options listed above.

14.9.4 Dynamic Allocation of VFs from a Pool

If you define the PCI address of a VF into a VM Guest's configuration statically as described in [Section 14.9.3, "Adding a VF Network Device to a VM Guest"](#), it is hard to migrate such guest to another host. The host must have identical hardware in the same location on the PCI bus, or the VM Guest configuration must be modified prior to each start.

Another approach is to create a libvirt network with a device pool that contains all the VFs of an *SR-IOV* device. The VM Guest then references this network, and each time it is started, a single VF is dynamically allocated to it. When the VM Guest is stopped, the VF is returned to the pool, available for another guest.

14.9.4.1 Defining Network with Pool of VFs on VM Host Server

The following example of network definition creates a pool of all VFs for the *SR-IOV* device with its physical function (PF) at the network interface eth0 on the host:

```
<network>
  <name>passthrough</name>
```

```
<forward mode='hostdev' managed='yes'>
  <pf dev='eth0' />
</forward>
</network>
```

To use this network on the host, save the above code to a file, for example `/tmp/passthrough.xml`, and execute the following commands. Remember to replace `eth0` with the real network interface name of your *SR-IOV* device's PF:

```
tux > sudo virsh net-define /tmp/passthrough.xml
tux > sudo virsh net-autostart passthrough
tux > sudo virsh net-start passthrough
```

14.9.4.2 Configuring VM Guests to Use VF from the Pool

The following example of VM Guest device interface definition uses a VF of the *SR-IOV* device from the pool created in [Section 14.9.4.1, "Defining Network with Pool of VFs on VM Host Server"](#). `libvirt` automatically derives the list of all VFs associated with that PF the first time the guest is started.

```
<interface type='network'>
  <source network='passthrough'>
</interface>
```

After the first VM Guest starts that uses the network with the pool of VFs, verify the list of associated VFs. Do so by running `virsh net-dumpxml passthrough` on the host.

```
<network connections='1'>
  <name>passthrough</name>
  <uuid>a6a26429-d483-d4ed-3465-4436ac786437</uuid>
  <forward mode='hostdev' managed='yes'>
    <pf dev='eth0' />
    <address type='pci' domain='0x0000' bus='0x02' slot='0x10' function='0x1' />
    <address type='pci' domain='0x0000' bus='0x02' slot='0x10' function='0x3' />
    <address type='pci' domain='0x0000' bus='0x02' slot='0x10' function='0x5' />
    <address type='pci' domain='0x0000' bus='0x02' slot='0x10' function='0x7' />
    <address type='pci' domain='0x0000' bus='0x02' slot='0x11' function='0x1' />
    <address type='pci' domain='0x0000' bus='0x02' slot='0x11' function='0x3' />
    <address type='pci' domain='0x0000' bus='0x02' slot='0x11' function='0x5' />
  </forward>
</network>
```


14.10 Configuring Storage Devices

Storage devices are defined within the `disk` element. The usual `disk` element supports several attributes. The following two attributes are the most important:

- The `type` attribute describes the source of the virtual disk device. Valid values are `file`, `block`, `dir`, `network`, or `volume`.
- The `device` attribute indicates how the disk is exposed to the VM Guest OS. As an example, possible values can include `floppy`, `disk`, `cdrom`, and others.

The following child elements are the most important:

- `driver` contains the driver and the bus. These are used by the VM Guest to work with the new disk device.
- The `target` element contains the device name under which the new disk is shown in the VM Guest. It also contains the optional bus attribute, which defines the type of bus on which the new disk should operate.

The following procedure shows how to add storage devices to the VM Guest:

1. Edit the configuration for an existing VM Guest:

```
tux > sudo virsh edit sles15
```

2. Add a `disk` element inside the `disk` element together with the attributes `type` and `device`:

```
<disk type='file' device='disk'>
```

3. Specify a `driver` element and use the default values:

```
<driver name='qemu' type='qcow2' />
```

4. Create a disk image, which will be used as a source for the new virtual disk device:

```
tux > sudo qemu-img create -f qcow2 /var/lib/libvirt/images/sles15.qcow2 32G
```

5. Add the path for the disk source:

```
<source file='/var/lib/libvirt/images/sles15.qcow2' />
```

6. Define the target device name in the VM Guest and the bus on which the disk should work:

```
<target dev='vda' bus='virtio' />
```

7. Restart your VM:

```
tux > sudo virsh start sles15
```

Your new storage device should be available in the VM Guest OS.

14.11 Configuring Controller Devices

libvirt generally manages controllers automatically based on the type of virtual devices used by the VM Guest. If the VM Guest contains PCI and SCSI devices, PCI and SCSI controllers will be created and managed automatically. **libvirt** will also model controllers that are hypervisor-specific, for example, a `virtio-serial` controller for KVM VM Guests or a `xenbus` controller for Xen VM Guests. Although the default controllers and their configuration are generally fine, there may be use cases where controllers or their attributes need to be adjusted manually. For example a `virtio-serial` controller may need more ports, or a `xenbus` controller may need more memory or more virtual interrupts.

The `xenbus` controller is unique, in that it serves as the controller for all Xen paravirtual devices. If a VM Guest has many disk and/or network devices the controller may need more memory. Xen's `max_grant_frames` attribute sets how many grant frames, or blocks of shared memory, are allocated to the `xenbus` controller for each VM Guest.

The default of 32 is enough in most circumstances, but a VM Guest with a large number of I/O devices and an I/O-intensive workload may experience performance issues due to grant frame exhaustion. The `xen-diag` can be used to check the current and maximum `max_grant_frames` values for `dom0` and your VM Guests. The VM Guests must be running:

```
tux > sudo virsh list
  Id   Name          State
-----
  0    Domain-0      running
  3    sle15spl      running

tux > sudo xen-diag gnttab_query_size 0
domid=0: nr_frames=1, max_nr_frames=256

tux > sudo xen-diag gnttab_query_size 3
```

```
domid=3: nr_frames=3, max_nr_frames=32
```

The `sle15sp1` guest is using only three frames out of 32. If you are seeing performance issues, and log entries that point to insufficient frames, increase the value with `virsh`. Look for the `<controller type='xenbus'>` line in the guest's configuration file, and add the `maxGrantFrames` control element:

```
tux > sudo virsh edit sle15sp1
<controller type='xenbus' index='0' maxGrantFrames='40' />
```

Save your changes and restart the guest. Now it should show your change:

```
tux > sudo xen-diag gnttab_query_size 3
domid=3: nr_frames=3, max_nr_frames=40
```

Similar to `maxGrantFrames`, the `xenbus` controller also supports `maxEventChannels`. Event channels are like paravirtual interrupts, and in conjunction with grant frames, form a data transfer mechanism for paravirtual drivers. They are also used for inter-processor interrupts. VM Guests with a large number of vcpus and/or many paravirtual devices may need to increase the maximum default value of 1023. `maxEventChannels` can be changed similar to `maxGrantFrames`:

```
tux > sudo virsh edit sle15sp1
<controller type='xenbus' index='0' maxGrantFrames='128' maxEventChannels='2047' />
```

See the *Controllers* section of the libvirt *Domain XML format* manual at <https://libvirt.org/format-domain.html#elementsControllers> for more information.

14.12 Configuring Video Devices

When using the Virtual Machine Manager, only the Video device model can be defined. The amount of allocated VRAM or 2D/3D acceleration can only be changed in the XML configuration.

14.12.1 Changing the Amount of Allocated VRAM

1. Edit the configuration for an existing VM Guest:

```
tux > sudo virsh edit sles15
```

2. Change the size of the allocated VRAM:

```
<video>
```

```
<model type='vga' vram='65535' heads='1'>
...
</model>
</video>
```

3. Check if the amount of VRAM in the VM has changed by looking at the amount in the Virtual Machine Manager.

14.12.2 Changing the State of 2D/3D Acceleration

1. Edit the configuration for an existing VM Guest:

```
tux > sudo virsh edit sles15
```

2. To enable/disable 2D/3D acceleration, change the value of `accel3d` and `accel2d` accordingly:

```
<video>
<acceleration accel3d='yes' accel2d='no'>
...
</model>
</video>
```



Tip: Enabling 2D/3D Acceleration

Only `vbox` video devices are capable of 2D/3D acceleration. You cannot enable it on other video devices.

14.13 Configuring Network Devices

This section describes how to configure specific aspects of virtual network devices by using `virsh`.

Find more details about `libvirt` network interface specification in <https://libvirt.org/formatdomain.html#elementsDriverBackendOptions>.

14.13.1 Scaling Network Performance with Multiqueue virtio-net

The multiqueue virtio-net feature scales the network performance by allowing VM Guest's virtual CPUs to transfer packets in parallel. Refer to [Section 27.3.3, "Scaling Network Performance with Multiqueue virtio-net"](#) for more general information.

To enable multiqueue virtio-net for a specific VM Guest, edit its XML configuration as described in [Section 14.1, "Editing the VM Configuration"](#) and modify its network interface as follows:

```
<interface type='network'>
  [...]
  <model type='virtio' />
  <driver name='vhost' queues='NUMBER_OF_QUEUES' />
</interface>
```

14.14 Using Macvtap to Share VM Host Server Network Interfaces

Macvtap provides direct attachment of a VM Guest virtual interface to a host network interface. The macvtap-based interface extends the VM Host Server network interface and has its own MAC address on the same Ethernet segment. Typically, this is used to make both the VM Guest and the VM Host Server show up directly on the switch that the VM Host Server is connected to.



Note: Macvtap Cannot Be Used with a Linux Bridge

Macvtap cannot be used with network interfaces already connected to a Linux bridge. Before attempting to create the macvtap interface, remove the interface from the bridge.



Note: VM Guest to VM Host Server Communication with Macvtap

When using macvtap, a VM Guest can communicate with other VM Guests, and with other external hosts on the network. But it cannot communicate with the VM Host Server on which the VM Guest runs. This is the defined behavior of macvtap, because of the way the VM Host Server's physical Ethernet is attached to the macvtap bridge. Traffic from the VM Guest into that bridge that is forwarded to the physical interface cannot be bounced

back up to the VM Host Server's IP stack. Similarly, traffic from the VM Host Server's IP stack that is sent to the physical interface cannot be bounced back up to the macvtap bridge for forwarding to the VM Guest.

Virtual network interfaces based on macvtap are supported by libvirt by specifying an interface type of `direct`. For example:

```
<interface type='direct'>
  <mac address='aa:bb:cc:dd:ee:ff' />
  <source dev='eth0' mode='bridge' />
  <model type='virtio' />
</interface>
```

The operation mode of the macvtap device can be controlled with the `mode` attribute. The following list show its possible values and a description for each:

- `vepa`: All VM Guest packets are sent to an external bridge. Packets whose destination is a VM Guest on the same VM Host Server as where the packet originates from are sent back to the VM Host Server by the VEPA capable bridge (today's bridges are typically not VEPA capable).
- `bridge`: Packets whose destination is on the same VM Host Server as where they originate from are directly delivered to the target macvtap device. Both origin and destination devices need to be in `bridge` mode for direct delivery. If either one of them is in `vepa` mode, a VEPA capable bridge is required.
- `private`: All packets are sent to the external bridge and will only be delivered to a target VM Guest on the same VM Host Server if they are sent through an external router or gateway and that device sends them back to the VM Host Server. This procedure is followed if either the source or destination device is in private mode.
- `passthrough`: A special mode that gives more power to the network interface. All packets will be forwarded to the interface, allowing virtio VM Guests to change the MAC address or set promiscuous mode to bridge the interface or create VLAN interfaces on top of it. Note that a network interface is not shareable in `passthrough` mode. Assigning an interface to a VM Guest will disconnect it from the VM Host Server. For this reason SR-IOV virtual functions are often assigned to the VM Guest in `passthrough` mode.

14.15 Disabling a Memory Balloon Device

Memory Balloon has become a default option for KVM. The device will be added to the VM Guest explicitly, so you do not need to add this element in the VM Guest's XML configuration. However, if you want to disable Memory Balloon in the VM Guest for any reason, you need to set `model='none'` as shown below:

```
<devices>
  <memballoon model='none' />
</device>
```

14.16 Configuring Multiple Monitors (Dual Head)

`libvirt` supports a dual head configuration to display the video output of the VM Guest on multiple monitors.



Important: No Support for Xen

The Xen hypervisor does not support dual head configuration.

PROCEDURE 14.3: CONFIGURING DUAL HEAD

1. While the virtual machine is running, verify that the `xf86-video-qxl` package is installed in the VM Guest:

```
tux > rpm -q xf86-video-qxl
```

2. Shutdown the VM Guest and start editing its configuration XML as described in [Section 14.1, "Editing the VM Configuration"](#).
3. Verify that the model of the virtual graphics card is 'qxl':

```
<video>
  <model type='qxl' ... />
```

4. Increase the `heads` parameter in the graphics card model specification from the default '1' to for example '2':

```
<video>
  <model type='qxl' ram='65536' vram='65536' vgamem='16384' heads='2' primary='yes' />
```

```
<alias name='video0' />
<address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x0' />
</video>
```

5. Configure the virtual machine to use the Spice display instead of VNC:

```
<graphics type='spice' port='5916' autoport='yes' listen='0.0.0.0'>
  <listen type='address' address='0.0.0.0' />
</graphics>
```

6. Start the virtual machine and connect to its display with **virt-viewer**, for example:

```
tux > virt-viewer --connect qemu+ssh://USER@VM_HOST/system
```

7. From the list of VMs, select the one whose configuration you have modified and confirm with *Connect*.
8. After the graphical subsystem (Xorg) loads in the VM Guest, select *View > Displays > Display 2* to open a new window with the second monitor's output.

III Hypervisor-Independent Features

- 15 Disk Cache Modes **160**
- 16 VM Guest Clock Settings **164**
- 17 libguestfs **166**

15 Disk Cache Modes

15.1 Disk Interface Cache Modes

Hypervisors allow for various storage caching strategies to be specified when configuring a VM Guest. Each guest disk interface can have one of the following cache modes specified: *writethrough*, *writeback*, *none*, *directsync*, or *unsafe*. If no cache mode is specified, an appropriate default cache mode is used. These cache modes influence how host-based storage is accessed, as follows:

- Read/write data may be cached in the host page cache.
- The guest's storage controller is informed whether a write cache is present, allowing for the use of a flush command.
- Synchronous write mode may be used, in which write requests are reported complete only when committed to the storage device.
- Flush commands (generated by the guest storage controller) may be ignored for performance reasons.

If a disorderly disconnection between the guest and its storage occurs, the cache mode in use will affect whether data loss occurs. The cache mode can also affect disk performance significantly. Additionally, some cache modes are incompatible with live migration, depending on several factors. There are no simple rules about what combination of cache mode, disk image format, image placement, or storage sub-system is best. The user should plan each guest's configuration carefully and experiment with various configurations to determine the optimal performance.

15.2 Description of Cache Modes

cache mode unspecified

In older QEMU versions, not specifying a cache mode meant that *writethrough* would be used as the default. With modern versions—as shipped with openSUSE Leap—the various guest storage interfaces have been fixed to handle *writeback* or *writethrough* semantics more correctly. This allows for the default caching mode to be switched to *writeback*. The guest

driver for each of ide, scsi, and virtio have within their power to disable the write back cache, causing the caching mode used to revert to *writethrough*. The typical guest's storage drivers will maintain the default caching mode as *writeback*, however.

writethrough

This mode causes the hypervisor to interact with the disk image file or block device with O_DSYNC semantics. Writes are reported as completed only when the data has been committed to the storage device. The host page cache is used in what can be termed a writethrough caching mode. The guest's virtual storage adapter is informed that there is no writeback cache, so the guest would not need to send down flush commands to manage data integrity. The storage behaves as if there is a writethrough cache.

writeback

This mode causes the hypervisor to interact with the disk image file or block device with neither O_DSYNC nor O_DIRECT semantics. The host page cache is used and writes are reported to the guest as completed when they are placed in the host page cache. The normal page cache management will handle commitment to the storage device. Additionally, the guest's virtual storage adapter is informed of the writeback cache, so the guest would be expected to send down flush commands as needed to manage data integrity. Analogous to a raid controller with RAM cache.

none

This mode causes the hypervisor to interact with the disk image file or block device with O_DIRECT semantics. The host page cache is bypassed and I/O happens directly between the hypervisor user space buffers and the storage device. Because the actual storage device may report a write as completed when placed in its write queue only, the guest's virtual storage adapter is informed that there is a writeback cache. The guest would be expected to send down flush commands as needed to manage data integrity. Performance-wise, it is equivalent to direct access to your host's disk.

unsafe

This mode is similar to the writeback mode discussed above. The key aspect of this “unsafe” mode, is that all flush commands from the guests are ignored. Using this mode implies that the user has accepted the trade-off of performance over risk of data loss in case of a host failure. Useful, for example, during guest installation, but not for production workloads.

directsync

This mode causes the hypervisor to interact with the disk image file or block device with both `O_DSYNC` and `O_DIRECT` semantics. This means, writes are reported as completed only when the data has been committed to the storage device, and when it is also desirable to bypass the host page cache. Like *writethrough*, it is helpful to guests that do not send flushes when needed. It was the last cache mode added, completing the possible combinations of caching and direct access semantics.

15.3 Data Integrity Implications of Cache Modes

writethrough, *none*, *directsync*

These are the safest modes, and considered equally safe, given that the guest operating system is “modern and well behaved”, which means that it uses flushes as needed. If you have a suspect guest, use *writethrough*, or *directsync*. Note that some file systems are not compatible with *none* or *directsync*, as they do not support `O_DIRECT`, which these cache modes rely on.

writeback

This mode informs the guest of the presence of a write cache, and relies on the guest to send flush commands as needed to maintain data integrity within its disk image. This is a common storage design which is completely accounted for within modern file systems. This mode exposes the guest to data loss in the unlikely case of a host failure, because there is a window of time between the time a write is reported as completed, and that write being committed to the storage device.

unsafe

This mode is similar to *writeback* caching except for the following: the guest flush commands are ignored, nullifying the data integrity control of these flush commands, and resulting in a higher risk of data loss because of host failure. The name “unsafe” should serve as a warning that there is a much higher potential for data loss because of a host failure than with the other modes. As the guest terminates, the cached data is flushed at that time.

15.4 Performance Implications of Cache Modes

The choice to make full use of the page cache, or to write through it, or to bypass it altogether can have dramatic performance implications. Other factors that influence disk performance include the capabilities of the actual storage system, what disk image format is used, the potential size

of the page cache and the IO scheduler used. Additionally, not flushing the write cache increases performance, but with risk, as noted above. As a general rule, high-end systems typically perform best with the cache mode `none`, because of the reduced data copying that occurs. The potential benefit of having multiple guests share the common host page cache, the ratio of reads to writes, and the use of AIO mode `native` (see below) should also be considered.

15.5 Effect of Cache Modes on Live Migration

The caching of storage data and metadata restricts the configurations that support live migration. Currently, only `raw` and `qcow2` image formats can be used for live migration. If a clustered file system is used, all cache modes support live migration. Otherwise the only cache mode that supports live migration on read/write shared storage is `none`.

The `libvirt` management layer includes checks for migration compatibility based on several factors. If the guest storage is hosted on a clustered file system, is read-only or is marked shareable, then the cache mode is ignored when determining if migration can be allowed. Otherwise `libvirt` will not allow migration unless the cache mode is set to `none`. However, this restriction can be overridden with the “unsafe” option to the migration APIs, which is also supported by `virsh`, as for example in

```
tux > virsh migrate --live --unsafe
```



Tip

The cache mode `none` is required for the AIO mode setting `native`. If another cache mode is used, then the AIO mode will silently be switched back to the default `threads`. The guest flush within the host is implemented using `fdatasync()`.

16 VM Guest Clock Settings

Keeping the correct time in a VM Guest is one of the more difficult aspects of virtualization. Keeping the correct time is especially important for network applications and is also a prerequisite to do a live migration of a VM Guest.



Tip: Timekeeping on the VM Host Server

It is strongly recommended to ensure the VM Host Server keeps the correct time as well, for example, by using NTP (see *Book "Reference", Chapter 18 "Time Synchronization with NTP"* for more information).

16.1 KVM: Using `kvm_clock`

KVM provides a paravirtualized clock which is supported via the `kvm_clock` driver. It is strongly recommended to use `kvm_clock`.

Use the following command inside a VM Guest running Linux to check whether the driver `kvm_clock` has been loaded:

```
tux > sudo dmesg | grep kvm-clock
[ 0.000000] kvm-clock: cpu 0, msr 0:7d3a81, boot clock
[ 0.000000] kvm-clock: cpu 0, msr 0:1206a81, primary cpu clock
[ 0.012000] kvm-clock: cpu 1, msr 0:1306a81, secondary cpu clock
[ 0.160082] Switching to clocksource kvm-clock
```

To check which clock source is currently used, run the following command in the VM Guest. It should output `kvm-clock`:

```
tux > cat /sys/devices/system/clocksource/clocksource0/current_clocksource
```



Important: `kvm-clock` and NTP

When using `kvm-clock`, it is recommended to use NTP in the VM Guest, as well. Using NTP on the VM Host Server is also recommended.

16.1.1 Other Timekeeping Methods

The paravirtualized `kvm-clock` is currently not for Windows* operating systems. For Windows*, use the [Windows Time Service Tools](#) for time synchronization (see <http://technet.microsoft.com/en-us/library/cc773263%28WS.10%29.aspx> for more information).

16.2 Xen Virtual Machine Clock Settings

With Xen 4, the independent wallclock setting `/proc/sys/xen/independent_wallclock` used for time synchronization between Xen host and guest was removed. A new configuration option `tsc_mode` was introduced. It specifies a method of utilizing the *timestamp counter* to synchronize the guest time with the Xen server. Its default value '0' handles the vast majority of hardware and software environments.

For more details on `tsc_mode`, see the `xen-tscmode` manual page (`man 7 xen-tscmode`).

17 libguestfs

Virtual Machines consist of disk images and definition files. Manually accessing and manipulating these guest components (outside of normal hypervisor processes) is possible, but inherently dangerous and risks compromising data integrity. libguestfs is a C library and a corresponding set of tools designed for safely accessing and modifying *Virtual Machine* disk images—outside of normal hypervisor processes, but without the risk normally associated with manual editing.

17.1 VM Guest Manipulation Overview

17.1.1 VM Guest Manipulation Risk

As disk images and definition files are simply another type of file in a Linux environment, it is possible to use many tools to access, edit and write to these files. When used correctly, such tools can be an important part of guest administration. However, even correct usage of these tools is not without risk. Risks that should be considered when manually manipulating guest disk images include:

- *Data Corruption*: Concurrently accessing images, by the host machine or another node in a cluster, can cause changes to be lost or data corruption to occur if virtualization protection layers are bypassed.
- *Security*: Mounting disk images as loop devices requires root access. While an image is loop mounted, other users and processes can potentially access the disk contents.
- *Administrator Error*: Bypassing virtualization layers correctly requires advanced understanding of virtual components and tools. Failing to isolate the images or failing to clean up properly after changes have been made can lead to further problems once back in virtualization control.

17.1.2 libguestfs Design

libguestfs C library has been designed to safely and securely create, access and modify virtual machine (VM Guest) disk images. It also provides additional language bindings: for Perl (<http://libguestfs.org/guestfs-perl.3.html>), Python (<http://libguestfs.org/guestfs-python.3.html>), PHP (only for 64-bit machines), and Ruby (<http://libguestfs.org/guestfs-ruby.3.html>). libguestfs can access VM Guest disk images without needing root and with multiple layers of defense against rogue disk images.

libguestfs provides many tools designed for accessing and modifying VM Guest disk images and contents. These tools provide such capabilities as: viewing and editing files inside guests, scripting changes to VM Guests, monitoring disk used/free statistics, creating guests, doing V2V or P2V migrations, performing backups, cloning VM Guests, formatting disks, and resizing disks.



Warning: Best Practices

You must not use libguestfs tools on live virtual machines. Doing so will probably result in disk corruption in the VM Guest. libguestfs tools try to stop you from doing this, but cannot catch all cases.

However most command have the `--ro` (read-only) option. With this option, you can attach a command to a live virtual machine. The results might be strange or inconsistent at times but you will not risk disk corruption.

17.2 Package Installation

libguestfs is shipped through 4 packages:

- `libguestfs0`: which provides the main C library
- `guestfs-data`: which contains the appliance files used when launching images (stored in `/usr/lib64/guestfs`)
- `guestfs-tools`: the core guestfs tools, man pages, and the `/etc/libguestfs-tools.conf` configuration file.
- `guestfs-winsupport`: provides support for Windows file guests in the guestfs tools. This package only needs to be installed to handle Windows guests, for example when converting a Windows guest to KVM.

To install guestfs tools on your system run:

```
tux > sudo zypper in guestfs-tools
```

17.3 Guestfs Tools

17.3.1 Modifying Virtual Machines

The set of tools found within the guestfs-tools package is used for accessing and modifying virtual machine disk images. This functionality is provided through a familiar shell interface with built-in safeguards which ensure image integrity. Guestfs tools shells expose all capabilities of the guestfs API, and create an appliance on the fly using the packages installed on the machine and the files found in [/usr/lib64/guestfs](#).

17.3.2 Supported File Systems and Disk Images

Guestfs tools support various file systems including:

- Ext2, Ext3, Ext4
- Xfs
- Btrfs

Multiple disk image formats are also supported:

- raw
- qcow2



Warning: Unsupported File System

Guestfs may also support Windows* file systems (VFAT, NTFS), BSD* and Apple* file systems, and other disk image formats (VMDK, VHDX...). However, these file systems and disk image formats are unsupported on openSUSE Leap.

17.3.3 virt-rescue

virt-rescue is similar to a rescue CD, but for virtual machines, and without the need for a CD. **virt-rescue** presents users with a rescue shell and some simple recovery tools which can be used to examine and correct problems within a virtual machine or disk image.

```
tux > virt-rescue -a sles.qcow2
Welcome to virt-rescue, the libguestfs rescue shell.

Note: The contents of / are the rescue appliance.
You need to mount the guest's partitions under /sysroot
before you can examine them. A helper script for that exists:
mount-rootfs-and-do-chroot.sh /dev/sda2

><rescue>
[ 67.194384] EXT4-fs (sda1): mounting ext3 file system
using the ext4 subsystem
[ 67.199292] EXT4-fs (sda1): mounted filesystem with ordered data
mode. Opts: (null)
mount: /dev/sda1 mounted on /sysroot.
mount: /dev bound on /sysroot/dev.
mount: /dev/pts bound on /sysroot/dev/pts.
mount: /proc bound on /sysroot/proc.
mount: /sys bound on /sysroot/sys.
Directory: /root
Thu Jun 5 13:20:51 UTC 2014
(none):~ #
```

You are now running the VM Guest in rescue mode:

```
(none):~ # cat /etc/fstab
devpts /dev/pts          devpts mode=0620,gid=5 0 0
proc   /proc                 proc   defaults                0 0
sysfs  /sys                  sysfs  noauto                  0 0
debugfs /sys/kernel/debug    debugfs noauto                  0 0
usbfs  /proc/bus/usb         usbfs  noauto                  0 0
tmpfs  /run                  tmpfs  noauto                  0 0
/dev/disk/by-id/ata-QEMU_HARDDISK_QM00001-part1 / ext3 defaults 1 1
```

17.3.4 virt-resize

virt-resize is used to resize a virtual machine disk, making it larger or smaller overall, and resizing or deleting any partitions contained within.

5. Bring up the VM Guest using the new disk image and confirm correct operation before deleting the old image.

17.3.5 Other virt-* Tools

There are guestfs tools to simplify administrative tasks—such as viewing and editing files, or obtaining information on the virtual machine.

17.3.5.1 virt-file systems

This tool is used to report information regarding file systems, partitions, and logical volumes in a disk image or virtual machine.

```
tux > virt-file systems -l -a sles.qcow2
Name      Type      VFS  Label  Size      Parent
/dev/sda1 filesystem ext3  -      17178820608 -
```

17.3.5.2 virt-ls

virt-ls lists file names, file sizes, checksums, extended attributes and more from a virtual machine or disk image. Multiple directory names can be given, in which case the output from each is concatenated. To list directories from a libvirt guest, use the **-d** option to specify the name of the guest. For a disk image, use the **-a** option.

```
tux > virt-ls -h -lR -a sles.qcow2 /var/log/
d 0755      776 /var/log
- 0640         0 /var/log/NetworkManager
- 0644      23K /var/log/Xorg.0.log
- 0644      23K /var/log/Xorg.0.log.old
d 0700      482 /var/log/YaST2
- 0644      512 /var/log/YaST2/_dev_vda
- 0644       59 /var/log/YaST2/arch.info
- 0644      473 /var/log/YaST2/config_diff_2017_05_03.log
- 0644     5.1K /var/log/YaST2/curl_log
- 0644     1.5K /var/log/YaST2/disk_vda.info
- 0644     1.4K /var/log/YaST2/disk_vda.info-1
[...]
```

17.3.5.3 **virt-cat**

virt-cat is a command line tool to display the contents of a file that exists in the named virtual machine (or disk image). Multiple file names can be given, in which case they are concatenated together. Each file name must be a full path, starting at the root directory (starting with '/').

```
tux > virt-cat -a sles.qcow2 /etc/fstab
devpts /dev/pts devpts mode=0620,gid=5 0 0
proc /proc proc defaults 0 0
```

17.3.5.4 **virt-df**

virt-df is a command line tool to display free space on virtual machine file systems. Unlike other tools, it not only displays the size of disk allocated to a virtual machine, but can look inside disk images to show how much space is actually being used.

```
tux > virt-df -a sles.qcow2
Filesystem                1K-blocks      Used Available  Use%
sles.qcow2:/dev/sda1      16381864      520564  15022492   4%
```

17.3.5.5 **virt-edit**

virt-edit is a command line tool capable of editing files that reside in the named virtual machine (or disk image).

17.3.5.6 **virt-tar-in/out**

virt-tar-in unpacks an uncompressed TAR archive into a virtual machine disk image or named libvirt domain. **virt-tar-out** packs a virtual machine disk image directory into a TAR archive.

```
tux > virt-tar-out -a sles.qcow2 /home homes.tar
```

17.3.5.7 **virt-copy-in/out**

virt-copy-in copies files and directories from the local disk into a virtual machine disk image or named libvirt domain. **virt-copy-out** copies files and directories out of a virtual machine disk image or named libvirt domain.

```
tux > virt-copy-in -a sles.qcow2 data.tar /tmp/
virt-ls -a sles.qcow2 /tmp/
.ICE-unix
.X11-unix
data.tar
```

17.3.5.8 **virt-log**

virt-log shows the log files of the named libvirt domain, virtual machine or disk image. If the package `guestfs-winsupport` is installed it can also show the event log of a Windows virtual machine disk image.

```
tux > virt-log -a windows8.qcow2
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
<Events>
<Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event"><System><Provider
  Name="EventLog"></Provider>
<EventID Qualifiers="32768">6011</EventID>
<Level>4</Level>
<Task>0</Task>
<Keywords>0x0080000000000000</Keywords>
<TimeCreated SystemTime="2014-09-12 05:47:21"></TimeCreated>
<EventRecordID>1</EventRecordID>
<Channel>System</Channel>
<Computer>windows-uj49s6b</Computer>
<Security UserID=""></Security>
</System>
<EventData><Data><string>WINDOWS-UJ49S6B</string>
<string>WIN-KG190623QG4</string>
</Data>
<Binary></Binary>
</EventData>
</Event>
...

```

17.3.6 **guestfish**

guestfish is a shell and command line tool for examining and modifying virtual machine file systems. It uses `libguestfs` and exposes all of the functionality of the `guestfs` API.

Examples of usage:

```
tux > guestfish -a disk.img <<EOF
run
list-fileSYSTEMS
EOF
```

```
guestfish
```

```
Welcome to guestfish, the guest filesystem shell for
editing virtual machine filesystems and disk images.
```

```
Type: 'help' for help on commands
      'man' to read the manual
      'quit' to quit the shell
```

```
><fs> add sles.qcow2
><fs> run
><fs> list-fileSYSTEMS
/dev/sda1: ext3
><fs> mount /dev/sda1 /
cat /etc/fstab
devpts /dev/pts          devpts mode=0620,gid=5 0 0
proc   /proc              proc   defaults                0 0
sysfs  /sys                 sysfs  noauto                  0 0
debugfs /sys/kernel/debug debugfs noauto                  0 0
usbfs  /proc/bus/usb         usbfs  noauto                  0 0
tmpfs  /run                 tmpfs  noauto                  0 0
/dev/disk/by-id/ata-QEMU_HARDDISK_QM00001-part1 / ext3 defaults 1 1
```

17.3.7 Converting a Physical Machine into a KVM Guest

Libguestfs provides tools to help converting Xen virtual machines or physical machines into KVM guests. The following section will cover a special use case: converting a bare metal machine into a KVM one.

Converting a physical machine into a KVM one is not yet supported in openSUSE Leap. This feature is released as a technology preview only.

Converting a physical machine requires collecting information about it and transmitting this to a conversion server. This is achieved by running a live system prepared with [virt-p2v](#) and [kiwi](#) tools on the machine.

1. Install the needed packages with the command:

```
tux > sudo zypper in virt-p2v kiwi-desc-isoboot
```



Note

These steps will document how to create an ISO image to create a bootable DVD. Alternatively, you can create a PXE boot image instead; for more information about building PXE images with KIWI, see [man virt-p2v-make-kiwi](#).

2. Create a KIWI configuration:

```
tux > virt-p2v-make-kiwi -o /tmp/p2v.kiwi
```

The `-o` defines where to create the KIWI configuration.

3. Edit the `config.xml` file in the generated configuration if needed. For example, in `config.xml` adjust the keyboard layout of the live system.
4. Build the ISO image with [kiwi](#):

```
tux > kiwi --build /tmp/p2v.kiwi ❶ \
-d /tmp/build ❷ \
--ignore-repos \
--add-repo http://URL/TO/SLE/REPOSITORIES ❸ \
--type iso
```

- ❶ The directory where the KIWI configuration was generated in the previous step.
 - ❷ The directory where KIWI will place the generated ISO image and other intermediary build results.
 - ❸ The URLs to the package repositories as found with `zypper lr -d`. Use one `--add-repo` parameter per repository.
5. Burn the ISO on a DVD or a USB stick. With such a medium, boot the machine to be converted.
 6. After the system is started, you will be asked for the connection details of the *conversion server*. This server is a machine with the `virt-v2v` package installed. If the network setup is more complex than a DHCP client, click the *Configure network* button to open the YaST network configuration dialog.

Click the *Test connection* button to allow moving to the next page of the wizard.

7. Select the disks and network interfaces to be converted and define the VM data like the amount of allocated CPUs, memory and the Virtual Machine name.



Note

If not defined, the created disk image format will be *raw* by default. This can be changed by entering the desired format in the *Output format* field.

There are two possibilities to generate the virtual machine: either using the *local* or the *libvirt* output. The first one will place the Virtual Machine disk image and configuration in the path defined in the *Output storage* field. These can then be used to define a new libvirt-handled guest using [virsh](#). The second method will create a new libvirt-handled guest with the disk image placed in the pool defined in the *Output storage* field.

Click *Start conversion* to start it.

17.4 Troubleshooting

17.4.1 Btrfs-related Problems

When using the `guestfs` tools on an image with Btrfs root partition (the default with openSUSE Leap) the following error message may be displayed:

```
tux > virt-ls -a /path/to/sles12sp2.qcow2 /
virt-ls: multi-boot operating systems are not supported
```

If using `guestfish` `-i` option, remove this option and instead use the commands `'run'` followed by `'list-file systems'`. You can then mount filesystems you want by hand using the `'mount'` or `'mount-ro'` command.

If using `guestmount` `-i`, remove this option and choose the filesystem(s) you want to see by manually adding `'-m'` option(s). Use `'virt-file systems'` to see what filesystems are available.

If using other `virt` tools, multi-boot operating systems won't work with these tools. Use the `guestfish` equivalent commands (see the `virt` tool manual page).

This is usually caused by the presence of snapshots in the guests. In this case guestfs does not know which snapshot to bootstrap. To force the use of a snapshot, use the `-m` parameter as follows:

```
tux > virt-ls -m /dev/sda2::subvol=@/.snapshots/2/snapshot -a /path/to/sles12sp2.qcow2 /
```

17.4.2 Environment

When troubleshooting problems within a libguestfs appliance, the environment variable `LIBGUESTFS_DEBUG=1` can be used to enable debug messages. To output each command/API call in a format that is similar to guestfish commands, use the environment variable `LIBGUESTFS_TRACE=1`.

17.4.3 `libguestfs-test-tool`

`libguestfs-test-tool` is a test program that checks if basic libguestfs functionality is working. It will print a large amount of diagnostic messages and details of the guestfs environment, then create a test image and try to start it. If it runs to completion successfully, the following message should be seen near the end:

```
===== TEST FINISHED OK =====
```

17.5 External References

- [libguestfs.org \(http://libguestfs.org\)](http://libguestfs.org) ↗
- [libguestfs FAQ \(http://libguestfs.org/guestfs-faq.1.html\)](http://libguestfs.org/guestfs-faq.1.html) ↗

IV Managing Virtual Machines with Xen

- 18 Setting Up a Virtual Machine Host **179**
- 19 Virtual Networking **190**
- 20 Managing a Virtualization Environment **199**
- 21 Block Devices in Xen **205**
- 22 Virtualization: Configuration Options and Settings **209**
- 23 Administrative Tasks **218**
- 24 XenStore: Configuration Database Shared between Domains **226**
- 25 Xen as a High-Availability Virtualization Host **231**

18 Setting Up a Virtual Machine Host

This section documents how to set up and use openSUSE Leap 15.2 as a virtual machine host. Usually, the hardware requirements for the Dom0 are the same as those for the openSUSE Leap operating system. Additional CPU, disk, memory, and network resources should be added to accommodate the resource demands of all planned VM Guest systems.



Tip: Resources

Remember that VM Guest systems, like physical machines, perform better when they run on faster processors and have access to more system memory.

The virtual machine host requires several software packages and their dependencies to be installed. To install all necessary packages, run YaST *Software Management*, select *View > Patterns* and choose *Xen Virtual Machine Host Server* for installation. The installation can also be performed with YaST using the module *Virtualization > Install Hypervisor and Tools*.

After the Xen software is installed, restart the computer and, on the boot screen, choose the newly added option with the Xen kernel.

Updates are available through your update channel. To be sure to have the latest updates installed, run YaST *Online Update* after the installation has finished.

18.1 Best Practices and Suggestions

When installing and configuring the openSUSE Leap operating system on the host, be aware of the following best practices and suggestions:

- If the host should always run as Xen host, run YaST *System > Boot Loader* and activate the Xen boot entry as default boot section.
 - In YaST, click *System > Boot Loader*.
 - Change the default boot to the *Xen* label, then click *Set as Default*.
 - Click *Finish*.
- For best performance, only the applications and processes required for virtualization should be installed on the virtual machine host.

- When using both iSCSI and OCFS2 to host Xen images, the latency required for OCFS2 default timeouts in openSUSE Leap may not be met. To reconfigure this timeout, run `systemctl configure o2cb` or edit `O2CB_HEARTBEAT_THRESHOLD` in the system configuration.
- If you intend to use a watchdog device attached to the Xen host, use only one at a time. It is recommended to use a driver with actual hardware integration over a generic software one.



Note: Hardware Monitoring

The Dom0 kernel is running virtualized, so tools like `irqbalance` or `lscpu` will not reflect the real hardware characteristics.

18.2 Managing Dom0 Memory

In previous versions of openSUSE Leap, the default memory allocation scheme of a Xen host was to allocate all host physical memory to Dom0 and enable auto-ballooning. Memory was automatically ballooned from Dom0 when additional domains were started. This behavior has always been error prone and disabling it was strongly encouraged. Starting with openSUSE Leap 15.1, auto-ballooning has been disabled by default and Dom0 is given 10% of host physical memory + 1GB. For example, on a host with 32GB of physical memory, 4.2GB of memory is allocated for Dom0.

The use of `dom0_mem` Xen command line option in `/etc/default/grub` is still supported and encouraged. You can restore the old behavior by setting `dom0_mem` to the host physical memory size and enabling the `autoballoon` setting in `/etc/xen/xl.conf`.



Warning: Insufficient Memory for Dom0

The amount of memory reserved for Dom0 is a function of the number of VM Guests running on the host since Dom0 provides back-end network and disk I/O services for each VM Guest. Other workloads running in Dom0 should also be considered when calculating Dom0 memory allocation. In general, memory sizing of Dom0 should be determined like any other virtual machine.

18.2.1 Setting Dom0 Memory Allocation

1. Determine memory allocation required for Dom0.
2. At Dom0, type `xl info` to view the amount of memory that is available on the machine. Dom0's current memory allocation can be determined with the `xl list` command.
3. Run `YaST > Boot Loader`.
4. Select the Xen section.
5. In *Additional Xen Hypervisor Parameters*, add `dom0_mem=MEM_AMOUNT` where `MEM_AMOUNT` is the maximum amount of memory to allocate to Dom0. Add `K`, `M`, or `G`, to specify the size, for example, `dom0_mem=2G`.
6. Restart the computer to apply the changes.



Warning: Xen Dom0 Memory

When using the XL tool stack and the `dom0_mem=` option for the Xen hypervisor in GRUB 2 you need to disable `xl autoballoon` in `etc/xen/xl.conf`. Otherwise launching VMs will fail with errors about not being able to balloon down Dom0. So add `autoballoon=0` to `xl.conf` if you have the `dom0_mem=` option specified for Xen. Also see [Xen dom0 memory \(http://wiki.xen.org/wiki/Xen_Best_Practices#Xen_dom0_dedicat-ed_memory_and_preventing_dom0_memory_balloonning\)](http://wiki.xen.org/wiki/Xen_Best_Practices#Xen_dom0_dedicat-ed_memory_and_preventing_dom0_memory_balloonning) ↗

18.3 Network Card in Fully Virtualized Guests

In a fully virtualized guest, the default network card is an emulated Realtek network card. However, it is also possible to use the split network driver to run the communication between Dom0 and a VM Guest. By default, both interfaces are presented to the VM Guest, because the drivers of some operating systems require both to be present.

When using openSUSE Leap, only the paravirtualized network cards are available for the VM Guest by default. The following network options are available:

emulated

To use an emulated network interface like an emulated Realtek card, specify `type=ioemu` in the `vif` device section of the domain `xl` configuration. An example configuration would look like:

```
vif = [ 'type=ioemu,mac=00:16:3e:5f:48:e4,bridge=br0' ]
```

Find more details about the `xl` configuration in the `xl.conf` manual page [man 5 xl.conf](#).

paravirtualized

When you specify `type=vif` and do not specify a model or type, the paravirtualized network interface is used:

```
vif = [ 'type=vif,mac=00:16:3e:5f:48:e4,bridge=br0,backen=0' ]
```

emulated and paravirtualized

If the administrator should be offered both options, simply specify both type and model. The `xl` configuration would look like:

```
vif = [ 'type=ioemu,mac=00:16:3e:5f:48:e4,model=rtl8139,bridge=br0' ]
```

In this case, one of the network interfaces should be disabled on the VM Guest.

18.4 Starting the Virtual Machine Host

If virtualization software is correctly installed, the computer boots to display the GRUB 2 boot loader with a *Xen* option on the menu. Select this option to start the virtual machine host.



Note: Xen and Kdump

In Xen, the hypervisor manages the memory resource. If you need to reserve system memory for a recovery kernel in Dom0, this memory need to be reserved by the hypervisor. Thus, it is necessary to add the parameter `crashkernel=size` to the `kernel` line instead of using the line with the other boot parameters.

For more information on the `crashkernel` parameter, see *Book "System Analysis and Tuning Guide", Chapter 17 "Kexec and Kdump", Section 17.4 "Calculating crashkernel Allocation Size"*.

If the *Xen* option is not on the GRUB 2 menu, review the steps for installation and verify that the GRUB 2 boot loader has been updated. If the installation has been done without selecting the Xen pattern, run the YaST *Software Management*, select the filter *Patterns* and choose *Xen Virtual Machine Host Server* for installation.

After booting the hypervisor, the Dom0 virtual machine starts and displays its graphical desktop environment. If you did not install a graphical desktop, the command line environment appears.



Tip: Graphics Problems

Sometimes it may happen that the graphics system does not work properly. In this case, add `vga=ask` to the boot parameters. To activate permanent settings, use `vga=mode-0x???` where `???` is calculated as `0x100` + VESA mode from http://en.wikipedia.org/wiki/VESA_BIOS_Extensions, for example `vga=mode-0x361`.

Before starting to install virtual guests, make sure that the system time is correct. To do this, configure NTP (Network Time Protocol) on the controlling domain:

1. In YaST select *Network Services* > *NTP Configuration*.
2. Select the option to automatically start the NTP daemon during boot. Provide the IP address of an existing NTP time server, then click *Finish*.



Note: Time Services on Virtual Guests

Hardware clocks commonly are not very precise. All modern operating systems try to correct the system time compared to the hardware time by means of an additional time source. To get the correct time on all VM Guest systems, also activate the network time services on each respective guest or make sure that the guest uses the system time of the host. For more about Independent Wallclocks in openSUSE Leap see [Section 16.2, "Xen Virtual Machine Clock Settings"](#).

For more information about managing virtual machines, see [Chapter 20, Managing a Virtualization Environment](#).

18.5 PCI Pass-Through

To take full advantage of VM Guest systems, it is sometimes necessary to assign specific PCI devices to a dedicated domain. When using fully virtualized guests, this functionality is only available if the chipset of the system supports this feature, and if it is activated from the BIOS.

This feature is available from both AMD* and Intel*. For AMD machines, the feature is called *IOMMU*; in Intel speak, this is *VT-d*. Note that Intel-VT technology is not sufficient to use this feature for fully virtualized guests. To make sure that your computer supports this feature, ask your supplier specifically to deliver a system that supports PCI Pass-Through.

LIMITATIONS

- Some graphics drivers use highly optimized ways to access DMA. This is not supported, and thus using graphics cards may be difficult.
- When accessing PCI devices behind a *PCIe* bridge, all of the PCI devices must be assigned to a single guest. This limitation does not apply to *PCIe* devices.
- Guests with dedicated PCI devices cannot be migrated live to a different host.

The configuration of PCI Pass-Through is twofold. First, the hypervisor must be informed at boot time that a PCI device should be available for reassigning. Second, the PCI device must be assigned to the VM Guest.

18.5.1 Configuring the Hypervisor for PCI Pass-Through

1. Select a device to reassign to a VM Guest. To do this, run `lspci -k`, and read the device number and the name of the original module that is assigned to the device:

```
06:01.0 Ethernet controller: Intel Corporation Ethernet Connection I217-LM (rev 05)
Subsystem: Dell Device 0617
Kernel driver in use: e1000e
Kernel modules: e1000e
```

In this case, the PCI number is (06:01.0) and the dependent kernel module is e1000e.

2. Specify a module dependency to ensure that `xen_pciback` is the first module to control the device. Add the file named `/etc/modprobe.d/50-e1000e.conf` with the following content:

```
install e1000e /sbin/modprobe xen_pciback ; /sbin/modprobe \
--first-time --ignore-install e1000e
```

3. Instruct the `xen_pciback` module to control the device using the 'hide' option. Edit or create `/etc/modprobe.d/50-xen-pciback.conf` with the following content:

```
options xen_pciback hide=(06:01.0)
```

4. Reboot the system.
5. Check if the device is in the list of assignable devices with the command

```
xl pci-assignable-list
```

18.5.1.1 Dynamic Assignment with xl

To avoid restarting the host system, you can use dynamic assignment with `xl` to use PCI Pass-Through.

Begin by making sure that `dom0` has the `pciback` module loaded:

```
tux > sudo modprobe pciback
```

Then make a device assignable by using `xl pci-assignable-add`. For example, to make the device `06:01.0` available for guests, run the command:

```
tux > sudo xl pci-assignable-add 06:01.0
```

18.5.2 Assigning PCI Devices to VM Guest Systems

There are several possibilities to dedicate a PCI device to a VM Guest:

Adding the device while installing:

During installation, add the `pci` line to the configuration file:

```
pci=['06:01.0']
```

Hotplugging PCI devices to VM Guest systems

The command `xl` can be used to add or remove PCI devices on the fly. To add the device with number `06:01.0` to a guest with name `sles12` use:

```
xl pci-attach sles12 06:01.0
```

Adding the PCI device to Xend

To add the device to the guest permanently, add the following snippet to the guest configuration file:

```
pci = [ '06:01.0,power_mgmt=1,permissive=1' ]
```

After assigning the PCI device to the VM Guest, the guest system must care for the configuration and device drivers for this device.

18.5.3 VGA Pass-Through

Xen 4.0 and newer supports VGA graphics adapter pass-through on fully virtualized VM Guests. The guest can take full control of the graphics adapter with high-performance full 3D and video acceleration.

LIMITATIONS

- VGA Pass-Through functionality is similar to PCI Pass-Through and as such also requires *IOMMU* (or Intel VT-d) support from the mainboard chipset and BIOS.
- Only the primary graphics adapter (the one that is used when you power on the computer) can be used with VGA Pass-Through.
- VGA Pass-Through is supported only for fully virtualized guests. Paravirtual guests (PV) are not supported.
- The graphics card cannot be shared between multiple VM Guests using VGA Pass-Through — you can dedicate it to one guest only.

To enable VGA Pass-Through, add the following settings to your fully virtualized guest configuration file:

```
gfx_passthru=1
pci=[ 'yy:zz.n' ]
```

where yy:zz.n is the PCI controller ID of the VGA graphics adapter as found with `lspci -v` on Dom0.

18.5.4 Troubleshooting

In some circumstances, problems may occur during the installation of the VM Guest. This section describes some known problems and their solutions.

During boot, the system hangs

The software I/O translation buffer allocates a large chunk of low memory early in the bootstrap process. If the requests for memory exceed the size of the buffer it usually results in a hung boot process. To check if this is the case, switch to console 10 and check the output there for a message similar to

```
kernel: PCI-DMA: Out of SW-IOMMU space for 32768 bytes at device 000:01:02.0
```

In this case you need to increase the size of the `swiotlb`. Add `swiotlb=VALUE` (where `VALUE` is specified as the number of slab entries) on the cmdline of Dom0. Note that the number can be adjusted up or down to find the optimal size for the machine.



Note: swiotlb a PV guest

The `swiotlb=force` kernel parameter is required for DMA access to work for PCI devices on a PV guest. For more information about IOMMU and the `swiotlb` option see the file `boot-options.txt` from the package `kernel-source`.

18.5.5 For More Information

There are several resources on the Internet that provide interesting information about PCI Pass-Through:

- https://wiki.xenproject.org/wiki/VTd_HowTo ↗
- <http://software.intel.com/en-us/articles/intel-virtualization-technology-for-directed-io-vt-d-enhancing-intel-platforms-for-efficient-virtualization-of-io-devices/> ↗
- http://support.amd.com/TechDocs/48882_IOMMU.pdf ↗

18.6 USB Pass-Through

There are two methods for passing through individual host USB devices to a guest. The first is via an emulated USB device controller, the second is using PVUSB.

18.6.1 Identify the USB Device

Before you can pass through a USB device to the VM Guest, you need to identify it on the VM Host Server. Use the `lsusb` command to list the USB devices on the host system:

```
root # lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 003: ID 0461:4d15 Primax Electronics, Ltd Dell Optical Mouse
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

To pass through the Dell mouse, for example, specify either the device tag in the form `vendor_id:device_id` (0461:4d15) or the bus address in the form `bus.device` (2.3). Remember to remove leading zeros, otherwise `x1` would interpret the numbers as octal values.

18.6.2 Emulated USB Device

In emulated USB, the device model (QEMU) presents an emulated USB controller to the guest. The USB device is then controlled from Dom0 while USB commands are translated between the VM Guest and the host USB device. This method is only available to fully virtualized domains (HVM).

Enable the emulated USB hub with the `usb=1` option. Then specify devices among the list of devices in the config file along with other emulated devices by using `host:USBID`. For example:

```
usb=1
usbdevice=['tablet','host:2.3','host:0424:460']
```

18.6.3 Paravirtualized PVUSB

PVUSB is a new high performance method for USB Pass-Through from dom0 to the virtualized guests. With PVUSB, there are two ways to add USB devices to a guest:

- via the configuration file at domain creation time
- via hotplug while the VM is running

PVUSB uses paravirtualized front- and back-end interfaces. PVUSB supports USB 1.1 and USB 2.0, and it works for both PV and HVM guests. To use PVUSB, you need `usbfront` in your guest OS, and `usbback` in dom0 or `usb back-end` in `qemu`. On openSUSE Leap, the USB back-end comes with `qemu`.

As of Xen 4.7, `x1` PVUSB support and hotplug support is introduced.

In the configuration file, specify USB controllers and USB host devices with `usbctrl` and `usbdev`. For example, in case of HVM guests:

```
usbctrl=['type=qusb,version=2,ports=4','type=qusb,version=1,ports=4', ]
usbdev=['hostbus=2, hostaddr=1, controller=0,port=1', ]
```



Note

It is important to specify `type=qusb` for the controller of HVM guests.

To manage hotpluggin PVUSB devices, use the `usbctrl-attach`, `usbctrl-detach`, `usb-list`, `usbdev-attach` and `usb-detach` subcommands. For example:

Create a USB controller which is version USB 1.1 and has 8 ports:

```
root # xl usbctrl-attach test_vm version=1 ports=8 type=qusb
```

Find the first available controller:port in the domain, and attach USB device whose busnum:devnum is 2:3 to it; you can also specify `controller` and `port`:

```
root # xl usbdev-attach test_vm hostbus=2 hostaddr=3
```

Show all USB controllers and USB devices in the domain:

```
root # xl usb-list test_vm
Devid Type  BE  state usb-ver ports
0      qusb  0  1     1       8
  Port 1: Bus 002 Device 003
  Port 2:
  Port 3:
  Port 4:
  Port 5:
  Port 6:
  Port 7:
  Port 8:
```

Detach the USB device under controller 0 port 1:

```
root # xl usbdev-detach test_vm 0 1
```

Remove the USB controller with the indicated `dev_id`, and all USB devices under it:

```
root # xl usbctrl-detach test_vm dev_id
```

For more information, see https://wiki.xenproject.org/wiki/Xen_USB_Passthrough.

19 Virtual Networking

A VM Guest system needs some means to communicate either with other VM Guest systems or with a local network. The network interface to the VM Guest system is made of a split device driver, which means that any virtual Ethernet device has a corresponding network interface in Dom0. This interface is set up to access a virtual network that is run in Dom0. The bridged virtual network is fully integrated into the system configuration of openSUSE Leap and can be configured with YaST.

When installing a Xen VM Host Server, a bridged network configuration will be proposed during normal network configuration. The user can choose to change the configuration during the installation and customize it to the local needs.

If desired, Xen VM Host Server can be installed after performing a default Physical Server installation using the Install Hypervisor and Tools module in YaST. This module will prepare the system for hosting virtual machines, including invocation of the default bridge networking proposal.

In case the necessary packages for a Xen VM Host Server are installed manually with rpm or zypper, the remaining system configuration needs to be done by the administrator manually or with YaST.

The network scripts that are provided by Xen are not used by default in openSUSE Leap. They are only delivered for reference but disabled. The network configuration that is used in openSUSE Leap is done by means of the YaST system configuration similar to the configuration of network interfaces in openSUSE Leap.

For more general information about managing network bridges, see [Section 12.1, “Network Bridge”](#).

19.1 Network Devices for Guest Systems

The Xen hypervisor can provide different types of network interfaces to the VM Guest systems. The preferred network device should be a paravirtualized network interface. This yields the highest transfer rates with the lowest system requirements. Up to eight network interfaces may be provided for each VM Guest.

Systems that are not aware of paravirtualized hardware may not have this option. To connect systems to a network that can only run fully virtualized, several emulated network interfaces are available. The following emulations are at your disposal:

- Realtek 8139 (PCI). This is the default emulated network card.
- AMD PCnet32 (PCI)
- NE2000 (PCI)
- NE2000 (ISA)
- Intel e100 (PCI)
- Intel e1000 and its variants e1000-82540em, e1000-82544gc, e1000-82545em (PCI)

All these network interfaces are software interfaces. Because every network interface must have a unique MAC address, an address range has been assigned to XenSource that can be used by these interfaces.



Tip: Virtual Network Interfaces and MAC Addresses

The default configuration of MAC addresses in virtualized environments creates a random MAC address that looks like 00:16:3E:xx:xx:xx. Normally, the amount of available MAC addresses should be big enough to get only unique addresses. However, if you have a very big installation, or to make sure that no problems arise from random MAC address assignment, you can also manually assign these addresses.

For debugging or system management purposes, it may be useful to know which virtual interface in Dom0 is connected to which Ethernet device in a running guest. This information may be read from the device naming in Dom0. All virtual devices follow the rule vif<domain number>.<interface_number>.

For example, if you want to know the device name for the third interface (eth2) of the VM Guest with id 5, the device in Dom0 would be vif5.2. To obtain a list of all available interfaces, run the command ip a.

The device naming does not contain any information about which bridge this interface is connected to. However, this information is available in Dom0. To get an overview about which interface is connected to which bridge, run the command **bridge link**. The output may look as follows:

```
tux > sudo bridge link
2: eth0 state DOWN : <NO-CARRIER,BROADCAST,MULTICAST,SLAVE,UP> mtu 1500 master br0
3: eth1 state UP : <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 master br1
```

In this example, there are three configured bridges: br0, br1 and br2. Currently, br0 and br1 each have a real Ethernet device added: eth0 and eth1, respectively.

19.2 Host-Based Routing in Xen

Xen can be set up to use host-based routing in the controlling Dom0. Unfortunately, this is not yet well supported from YaST and requires quite an amount of manual editing of configuration files. Thus, this is a task that requires an advanced administrator.

The following configuration will only work when using fixed IP addresses. Using DHCP is not practicable with this procedure, because the IP address must be known to both, the VM Guest and the VM Host Server system.

The easiest way to create a routed guest is to change the networking from a bridged to a routed network. As a requirement to the following procedures, a VM Guest with a bridged network setup must be installed. For example, the VM Host Server is named earth with the IP 192.168.1.20, and the VM Guest has the name alice with the IP 192.168.1.21.

PROCEDURE 19.1: CONFIGURING A ROUTED IPV4 VM GUEST

1. Make sure that alice is shut down. Use `xl` commands to shut down and check.
2. Prepare the network configuration on the VM Host Server earth:
 - a. Create a hotplug interface that will be used to route the traffic. To accomplish this, create a file named `/etc/sysconfig/network/ifcfg-alice.0` with the following content:

```
NAME="Xen guest alice"
BOOTPROTO="static"
STARTMODE="hotplug"
```

b. Ensure that IP forwarding is enabled:

- i. In YaST, go to *Network Settings* > *Routing*.
- ii. Enter the *Routing* tab and activate *Enable IPv4 Forwarding* and *Enable IPv6 Forwarding* options.
- iii. Confirm the setting and quit YaST.

c. Apply the following configuration to `firewalld`:

- Add `alice.0` to the devices in the public zone:

```
tux > sudo firewall-cmd --zone=public --add-interface=alice.0
```

- Tell the firewall which address should be forwarded:

```
tux > sudo firewall-cmd --zone=public \
--add-forward-
port=port=80:proto=tcp:toport=80:toaddr="192.168.1.21/32,0/0"
```

- Make the runtime configuration changes permanent:

```
tux > sudo firewall-cmd --runtime-to-permanent
```

d. Add a static route to the interface of alice. To accomplish this, add the following line to the end of `/etc/sysconfig/network/routes`:

```
192.168.1.21 - - alice.0
```

e. To make sure that the switches and routers that the VM Host Server is connected to know about the routed interface, activate `proxy_arp` on earth. Add the following lines to `/etc/sysctl.conf`:

```
net.ipv4.conf.default.proxy_arp = 1
net.ipv4.conf.all.proxy_arp = 1
```

f. Activate all changes with the commands:

```
tux > sudo systemctl restart systemd-sysctl wicked
```

3. Proceed with configuring the Xen configuration of the VM Guest by changing the vif interface configuration for alice as described in *Section 20.1, "XL—Xen Management Tool"*. Make the following changes to the text file you generate during the process:

- a. Remove the snippet

```
bridge=br0
```

- b. And add the following one:

```
vifname=vifalice.0
```

or

```
vifname=vifalice.0=emu
```

for a fully virtualized domain.

- c. Change the script that is used to set up the interface to the following:

```
script=/etc/xen/scripts/vif-route-ifup
```

- d. Activate the new configuration and start the VM Guest.

4. The remaining configuration tasks must be accomplished from inside the VM Guest.

- a. Open a console to the VM Guest with `xl console DOMAIN` and log in.
- b. Check that the guest IP is set to 192.168.1.21.
- c. Provide VM Guest with a host route and a default gateway to the VM Host Server. Do this by adding the following lines to `/etc/sysconfig/network/routes`:

```
192.168.1.20 - - eth0
default 192.168.1.20 - -
```

5. Finally, test the network connection from the VM Guest to the world outside and from the network to your VM Guest.

19.3 Creating a Masqueraded Network Setup

Creating a masqueraded network setup is quite similar to the routed setup. However, there is no `proxy_arp` needed, and some firewall rules are different. To create a masqueraded network to a guest dolly with the IP address 192.168.100.1 where the host has its external interface on `br0`, proceed as follows. For easier configuration, only the already installed guest is modified to use a masqueraded network:

PROCEDURE 19.2: CONFIGURING A MASQUERADED IPV4 VM GUEST

1. Shut down the VM Guest system with `xl shutdown DOMAIN`.
2. Prepare the network configuration on the VM Host Server:
 - a. Create a hotplug interface that will be used to route the traffic. To accomplish this, create a file named `/etc/sysconfig/network/ifcfg-dolly.0` with the following content:

```
NAME="Xen guest dolly"
BOOTPROTO="static"
STARTMODE="hotplug"
```

- b. Edit the file `/etc/sysconfig/SuSEfirewall2` and add the following configurations:

- Add dolly.0 to the devices in `FW_DEV_DMZ`:

```
FW_DEV_DMZ="dolly.0"
```

- Switch on the routing in the firewall:

```
FW_ROUTE="yes"
```

- Switch on masquerading in the firewall:

```
FW_MASQUERADE="yes"
```

- Tell the firewall which network should be masqueraded:

```
FW_MASQ_NETS="192.168.100.1/32"
```

- Remove the networks from the masquerading exceptions:

```
FW_NOMASQ_NETS=""
```

- Finally, restart the firewall with the command:

```
tux > sudo systemctl restart SuSEfirewall2
```

- c. Add a static route to the interface of dolly. To accomplish this, add the following line to the end of `/etc/sysconfig/network/routes`:

```
192.168.100.1 - - dolly.0
```

- d. Activate all changes with the command:

```
tux > sudo systemctl restart wicked
```

3. Proceed with configuring the Xen configuration of the VM Guest.

- a. Change the vif interface configuration for dolly as described in [Section 20.1, "XL—Xen Management Tool"](#).

- b. Remove the entry:

```
bridge=br0
```

- c. And add the following one:

```
vifname=vifdolly.0
```

- d. Change the script that is used to set up the interface to the following:

```
script=/etc/xen/scripts/vif-route-ifup
```

- e. Activate the new configuration and start the VM Guest.

4. The remaining configuration tasks need to be accomplished from inside the VM Guest.

- a. Open a console to the VM Guest with `xl console DOMAIN` and log in.

- b. Check whether the guest IP is set to 192.168.100.1.

- c. Provide VM Guest with a host route and a default gateway to the VM Host Server. Do this by adding the following lines to `/etc/sysconfig/network/routes`:

```
192.168.1.20 - - eth0
default 192.168.1.20 - -
```

5. Finally, test the network connection from the VM Guest to the outside world.

19.4 Special Configurations

There are many network configuration possibilities available to Xen. The following configurations are not activated by default:

19.4.1 Bandwidth Throttling in Virtual Networks

With Xen, you may limit the network transfer rate a virtual guest may use to access a bridge. To configure this, you need to modify the VM Guest configuration as described in [Section 20.1, “XL—Xen Management Tool”](#).

In the configuration file, first search for the device that is connected to the virtual bridge. The configuration looks like the following:

```
vif = [ 'mac=00:16:3e:4f:94:a9,bridge=br0' ]
```

To add a maximum transfer rate, add a parameter `rate` to this configuration as in:

```
vif = [ 'mac=00:16:3e:4f:94:a9,bridge=br0,rate=100Mb/s' ]
```

Note that the rate is either `Mb/s` (megabits per second) or `MB/s` (megabytes per second). In the above example, the maximum transfer rate of the virtual interface is 100 megabits. By default, there is no limitation to the bandwidth of a guest to the virtual bridge.

It is even possible to fine-tune the behavior by specifying the time window that is used to define the granularity of the credit replenishment:

```
vif = [ 'mac=00:16:3e:4f:94:a9,bridge=br0,rate=100Mb/s@20ms' ]
```

19.4.2 Monitoring the Network Traffic

To monitor the traffic on a specific interface, the little application `iftop` is a nice program that displays the current network traffic in a terminal.

When running a Xen VM Host Server, you need to define the interface that is monitored. The interface that Dom0 uses to get access to the physical network is the bridge device, for example `br0`. This, however, may vary on your system. To monitor all traffic to the physical interface, run a terminal as `root` and use the command:

```
iftop -i br0
```

To monitor the network traffic of a special network interface of a specific VM Guest, supply the correct virtual interface. For example, to monitor the first Ethernet device of the domain with id 5, use the command:

```
ftop -i vif5.0
```

To quit `iftop`, press the key `Q`. More options and possibilities are available in the manual page `man 8 iftop`.

20 Managing a Virtualization Environment

Apart from using the recommended `libvirt` library (*Part II, “Managing Virtual Machines with libvirt”*), you can manage Xen guest domains with the `xl` tool from the command line.

20.1 XL—Xen Management Tool

The `xl` program is a tool for managing Xen guest domains. It is part of the `xen-tools` package. `xl` is based on the LibXenlight library, and can be used for general domain management, such as domain creation, listing, pausing, or shutting down. Usually you need to be `root` to execute `xl` commands.



Note

`xl` can only manage running guest domains specified by their configuration file. If a guest domain is not running, you cannot manage it with `xl`.



Tip

To allow users to continue to have managed guest domains in the way the obsolete `xm` command allowed, we now recommend using `libvirt`'s `virsh` and `virt-manager` tools. For more information, see *Part II, “Managing Virtual Machines with libvirt”*.

`xl` operations rely upon `xenstored` and `xenconsoled` services. Make sure you start

```
tux > systemctl start xencommons
```

at boot time to initialize all the daemons required by `xl`.



Tip: Set up a `xenbr0` Network Bridge in the Host Domain

In the most common network configuration, you need to set up a bridge in the host domain named `xenbr0` to have a working network for the guest domains.

The basic structure of every `xl` command is:

```
xl <subcommand> [options] domain_id
```

where `<subcommand>` is the `xl` command to run, `domain_id` is the ID number assigned to a domain or the name of the virtual machine, and **OPTIONS** indicates subcommand-specific options.

For a complete list of the available `xl` subcommands, run `xl help`. For each command, there is a more detailed help available that is obtained with the extra parameter `--help`. More information about the respective subcommands is available in the manual page of `xl`.

For example, the `xl list --help` displays all options that are available to the list command. As an example, the `xl list` command displays the status of all virtual machines.

```
tux > sudo xl list
Name                ID    Mem VCPUs    State  Time(s)
Domain-0            0    457    2    r----- 2712.9
sles12              7    512    1    -b----- 16.3
opensuse            512    1    12.9
```

The *State* information indicates if a machine is running, and in which state it is. The most common flags are `r` (running) and `b` (blocked) where blocked means it is either waiting for IO, or sleeping because there is nothing to do. For more details about the state flags, see `man 1 xl`.

Other useful `xl` commands include:

- `xl create` creates a virtual machine from a given configuration file.
- `xl reboot` reboots a virtual machine.
- `xl destroy` immediately terminates a virtual machine.
- `xl block-list` displays all virtual block devices attached to a virtual machine.

20.1.1 Guest Domain Configuration File

When operating domains, `xl` requires a domain configuration file for each domain. The default directory to store such configuration files is `/etc/xen/`.

A domain configuration file is a plain text file. It consists of several `KEY = VALUE` pairs. Some keys are mandatory, some are general and apply to any guest, and some apply only to a specific guest type (para or fully virtualized). A value can either be a `"string"` surrounded by single or double quotes, a number, a boolean value, or a list of several values enclosed in brackets `[value1, value2, ...]`.

EXAMPLE 20.1: GUEST DOMAIN CONFIGURATION FILE FOR SLED 12: `/etc/xen/sled12.cfg`

```
name= "sled12"
```

```
builder = "hvm"
vncviewer = 1
memory = 512
disk = [ '/var/lib/xen/images/sled12.raw,,hda', '/dev/cdrom,,hdc,cdrom' ]
vif = [ 'mac=00:16:3e:5f:48:e4,model=rtl8139,bridge=br0' ]
boot = "n"
```

To start such domain, run `xl create /etc/xen/sled12.cfg`.

20.2 Automatic Start of Guest Domains

To make a guest domain start automatically after the host system boots, follow these steps:

1. Create the domain configuration file if it does not exist, and save it in the `/etc/xen/` directory, for example `/etc/xen/domain_name.cfg`.
2. Make a symbolic link of the guest domain configuration file in the `auto/` subdirectory.

```
tux > sudo ln -s /etc/xen/domain_name.cfg /etc/xen/auto/domain_name.cfg
```

3. On the next system boot, the guest domain defined in `domain_name.cfg` will be started.

20.3 Event Actions

In the guest domain configuration file, you can define actions to be performed on a predefined set of events. For example, to tell the domain to restart itself after it is powered off, include the following line in its configuration file:

```
on_poweroff="restart"
```

A list of predefined events for a guest domain follows:

LIST OF EVENTS

on_poweroff

Specifies what should be done with the domain if it shuts itself down.

on_reboot

Action to take if the domain shuts down with a reason code requesting a reboot.

on_watchdog

Action to take if the domain shuts down because of a Xen watchdog timeout.

on_crash

Action to take if the domain crashes.

For these events, you can define one of the following actions:

LIST OF RELATED ACTIONS

destroy

Destroy the domain.

restart

Destroy the domain and immediately create a new domain with the same configuration.

rename-restart

Rename the domain that terminated, and then immediately create a new domain with the same configuration as the original.

preserve

Keep the domain. It can be examined, and later destroyed with **xl destroy**.

coredump-destroy

Write a core dump of the domain to /var/xen/dump/NAME and then destroy the domain.

coredump-restart

Write a core dump of the domain to /var/xen/dump/NAME and then restart the domain.

20.4 Time Stamp Counter

The Time Stamp Counter (TSC) may be specified for each domain in the guest domain configuration file (for more information, see *Section 20.1.1, "Guest Domain Configuration File"*).

With the tsc_mode setting, you specify whether rdtsc instructions are executed “natively” (fast, but TSC-sensitive applications may sometimes run incorrectly) or emulated (always run correctly, but performance may suffer).

tsc_mode=0 (default)

Use this to ensure correctness while providing the best performance possible—for more information, see <https://xenbits.xen.org/docs/4.3-testing/misc/tscmode.txt>.

tsc_mode=1 (always emulate)

Use this when TSC-sensitive apps are running and worst-case performance degradation is known and acceptable.

tsc_mode=2 (never emulate)

Use this when all applications running in this VM are TSC-resilient and highest performance is required.

tsc_mode=3 (PVRDTSCP)

High-TSC-frequency applications may be paravirtualized (modified) to obtain both correctness and highest performance—any unmodified applications must be TSC-resilient.

For background information, see <https://xenbits.xen.org/docs/4.3-testing/misc/tscmode.txt>.

20.5 Saving Virtual Machines

PROCEDURE 20.1: SAVE A VIRTUAL MACHINE'S CURRENT STATE

1. Make sure the virtual machine to be saved is running.
2. In the host environment, enter

```
tux > sudo xl save ID STATE-FILE
```

where *ID* is the virtual machine ID you want to save, and *STATE-FILE* is the name you specify for the memory state file. By default, the domain will no longer be running after you create its snapshot. Use *-c* to keep it running even after you create the snapshot.

20.6 Restoring Virtual Machines

PROCEDURE 20.2: RESTORE A VIRTUAL MACHINE'S CURRENT STATE

1. Make sure the virtual machine to be restored has not been started since you ran the save operation.
2. In the host environment, enter

```
tux > sudo xl restore STATE-FILE
```

where *STATE-FILE* is the previously saved memory state file. By default, the domain will be running after it is restored. To pause it after the restore, use *-p*.

20.7 Virtual Machine States

A virtual machine's state can be displayed by viewing the results of the `xl list` command, which abbreviates the state using a single character.

- **r** - running - The virtual machine is currently running and consuming allocated resources.
- **b** - blocked - The virtual machine's processor is not running and not able to run. It is either waiting for I/O or has stopped working.
- **p** - paused - The virtual machine is paused. It does not interact with the hypervisor but still maintains its allocated resources, such as memory.
- **s** - shutdown - The guest operating system is in the process of being shut down, rebooted, or suspended, and the virtual machine is being stopped.
- **c** - crashed - The virtual machine has crashed and is not running.
- **d** - dying - The virtual machine is in the process of shutting down or crashing.

21 Block Devices in Xen

21.1 Mapping Physical Storage to Virtual Disks

The disk(s) specification for a Xen domain in the domain configuration file is as straightforward as the following example:

```
disk = [ 'format=raw,vdev=hdc,access=ro,devtype=cdrom,target=/root/image.iso' ]
```

It defines a disk block device based on the `/root/image.iso` disk image file. The disk will be seen as `hdc` by the guest, with read-only (`ro`) access. The type of the device is `cdrom` with `raw` format.

The following example defines an identical device, but using simplified positional syntax:

```
disk = [ '/root/image.iso,raw,hdc,ro,cdrom' ]
```

You can include more disk definitions in the same line, each one separated by a comma. If a parameter is not specified, then its default value is taken:

```
disk = [ '/root/image.iso,raw,hdc,ro,cdrom', '/dev/vg/guest-volume,,hda','...' ]
```

LIST OF PARAMETERS

target

Source block device or disk image file path.

format

The format of the image file. Default is `raw`.

vdev

Virtual device as seen by the guest. Supported values are `hd[x]`, `xvd[x]`, `sd[x]` etc. See </usr/share/doc/packages/xen/misc/vbd-interface.txt> for more details. This parameter is mandatory.

access

Whether the block device is provided to the guest in read-only or read-write mode. Supported values are `ro` or `r` for read-only, and `rw` or `w` for read/write access. Default is `ro` for `devtype=cdrom`, and `rw` for other device types.

devtype

Qualifies virtual device type. Supported value is `cdrom`.

backendtype

The back-end implementation to use. Supported values are `phy`, `tap`, and `qdisk`. Normally this option should not be specified as the back-end type is automatically determined.

script

Specifies that `target` is not a normal host path, but rather information to be interpreted by the executable program. The specified script file is looked for in `/etc/xen/scripts` if it does not point to an absolute path. These scripts are normally called `block-<script_name>`.

For more information about specifying virtual disks, see [/usr/share/doc/packages/xen/misc/xl-disk-configuration.txt](#).

21.2 Mapping Network Storage to Virtual Disk

Similar to mapping a local disk image (see [Section 21.1, “Mapping Physical Storage to Virtual Disks”](#)), you can map a network disk as a virtual disk as well.

The following example shows mapping of an RBD (RADOS Block Device) disk with multiple Ceph monitors and cephx authentication enabled:

```
disk = [ 'vdev=hdc, backendtype=qdisk, \
target=rbd:libvirt-pool/new-libvirt-image:\
id=libvirt:key=AQDsPwtW8JoXJBAAyLPQe7MhCC+JPKI3QuhaAw==:auth_supported=cephx;none:\
mon_host=137.65.135.205\\:6789;137.65.135.206\\:6789;137.65.135.207\\:6789' ]
```

Following is an example of an NBD (Network Block Device) disk mapping:

```
disk = [ 'vdev=hdc, backendtype=qdisk, target=nbd:151.155.144.82:5555' ]
```

21.3 File-Backed Virtual Disks and Loopback Devices

When a virtual machine is running, each of its file-backed virtual disks consumes a loopback device on the host. By default, the host allows up to 64 loopback devices to be consumed.

To simultaneously run more file-backed virtual disks on a host, you can increase the number of available loopback devices by adding the following option to the host's `/etc/modprobe.conf.local` file.


```
options loop max_loop=x
```

where x is the maximum number of loopback devices to create.

Changes take effect after the module is reloaded.



Tip

Enter `rmmod loop` and `modprobe loop` to unload and reload the module. In case `rmmod` does not work, unmount all existing loop devices or reboot the computer.

21.4 Resizing Block Devices

While it is always possible to add new block devices to a VM Guest system, it is sometimes more desirable to increase the size of an existing block device. In case such a system modification is already planned during deployment of the VM Guest, some basic considerations should be done:

- Use a block device that may be increased in size. LVM devices and file system images are commonly used.
- Do not partition the device inside the VM Guest, but use the main device directly to apply the file system. For example, use `/dev/xvdb` directly instead of adding partitions to `/dev/xvdb`.
- Make sure that the file system to be used can be resized. Sometimes, for example with Ext3, some features must be switched off to be able to resize the file system. A file system that can be resized online and mounted is `XFS`. Use the command `xfs_growfs` to resize that file system after the underlying block device has been increased in size. For more information about `XFS`, see `man 8 xfs_growfs`.

When resizing an LVM device that is assigned to a VM Guest, the new size is automatically known to the VM Guest. No further action is needed to inform the VM Guest about the new size of the block device.

When using file system images, a loop device is used to attach the image file to the guest. For more information about resizing that image and refreshing the size information for the VM Guest, see *Section 23.2, "Sparse Image Files and Disk Space"*.

21.5 Scripts for Managing Advanced Storage Scenarios

There are scripts that can help with managing advanced storage scenarios such as disk environments provided by **dmmd** (“device mapper—multi disk”) including LVM environments built upon a software RAID set, or a software RAID set built upon an LVM environment. These scripts are part of the `xen-tools` package. After installation, they can be found in `/etc/xen/scripts`:

- **block-dmmd**
- **block-drbd-probe**
- **block-npiv**

The scripts allow for external commands to perform some action, or series of actions of the block devices prior to serving them up to a guest.

These scripts could formerly only be used with **xl** or **libxl** using the disk configuration syntax `script=`. They can now be used with libvirt by specifying the base name of the block script in the `<source>` element of the disk. For example:

```
<source dev='dmmd:md;/dev/md0;lvm;/dev/vgxen/lv-vm01' />
```

22 Virtualization: Configuration Options and Settings

The documentation in this section, describes advanced management tasks and configuration options that might help technology innovators implement leading-edge virtualization solutions. It is provided as a courtesy and does not imply that all documented options and tasks are supported by Novell, Inc.

22.1 Virtual CD Readers

Virtual CD readers can be set up when a virtual machine is created or added to an existing virtual machine. A virtual CD reader can be based on a physical CD/DVD, or based on an ISO image. Virtual CD readers work differently depending on whether they are paravirtual or fully virtual.

22.1.1 Virtual CD Readers on Paravirtual Machines

A paravirtual machine can have up to 100 block devices composed of virtual CD readers and virtual disks. On paravirtual machines, virtual CD readers present the CD as a virtual disk with read-only access. Virtual CD readers cannot be used to write data to a CD.

After you have finished accessing a CD on a paravirtual machine, it is recommended that you remove the virtual CD reader from the virtual machine.

Paravirtualized guests can use the device type `devtype=cdrom`. This partly emulates the behavior of a real CD reader, and allows CDs to be changed. It is even possible to use the `eject` command to open the tray of the CD reader.

22.1.2 Virtual CD Readers on Fully Virtual Machines

A fully virtual machine can have up to four block devices composed of virtual CD readers and virtual disks. A virtual CD reader on a fully virtual machine interacts with an inserted CD in the way you would expect a physical CD reader to interact.

When a CD is inserted in the physical CD reader on the host computer, all virtual machines with virtual CD readers based on the physical CD reader, such as `/dev/cdrom/`, can read the inserted CD. Assuming the operating system has automount functionality, the CD should automatically appear in the file system. Virtual CD readers cannot be used to write data to a CD. They are configured as read-only devices.

22.1.3 Adding Virtual CD Readers

Virtual CD readers can be based on a CD inserted into the CD reader or on an ISO image file.

1. Make sure that the virtual machine is running and the operating system has finished booting.
2. Insert the desired CD into the physical CD reader or copy the desired ISO image to a location available to Dom0.
3. Select a new, unused block device in your VM Guest, such as `/dev/xvdb`.
4. Choose the CD reader or ISO image that you want to assign to the guest.
5. When using a real CD reader, use the following command to assign the CD reader to your VM Guest. In this example, the name of the guest is `alice`:

```
tux > sudo xl block-attach alice target=/dev/sr0,vdev=xvdb,access=ro
```

6. When assigning an image file, use the following command:

```
tux > sudo xl block-attach alice target=/path/to/file.iso,vdev=xvdb,access=ro
```

7. A new block device, such as `/dev/xvdb`, is added to the virtual machine.
8. If the virtual machine is running Linux, complete the following:

- a. Open a terminal in the virtual machine and enter `fdisk -l` to verify that the device was properly added. You can also enter `ls /sys/block` to see all disks available to the virtual machine.

The CD is recognized by the virtual machine as a virtual disk with a drive designation, for example:

```
/dev/xvdb
```

- b. Enter the command to mount the CD or ISO image using its drive designation. For example,

```
tux > sudo mount -o ro /dev/xvdb /mnt
```

mounts the CD to a mount point named `/mnt`.

The CD or ISO image file should be available to the virtual machine at the specified mount point.

9. If the virtual machine is running Windows, reboot the virtual machine. Verify that the virtual CD reader appears in its My Computer section.

22.1.4 Removing Virtual CD Readers

1. Make sure that the virtual machine is running and the operating system has finished booting.
2. If the virtual CD reader is mounted, unmount it from within the virtual machine.
3. Enter `xl block-list alice` on the host view of the guest block devices.
4. Enter `xl block-detach alice BLOCK_DEV_ID` to remove the virtual device from the guest. If that fails, try to add `-f` to force the removal.
5. Press the hardware eject button to eject the CD.

22.2 Remote Access Methods

Some configurations, such as those that include rack-mounted servers, require a computer to run without a video monitor, keyboard, or mouse. This type of configuration is often called headless and requires the use of remote administration technologies.

Typical configuration scenarios and technologies include:

Graphical Desktop with X Window Server

If a graphical desktop, such as GNOME, is installed on the virtual machine host, you can use a remote viewer, such as a VNC viewer. On a remote computer, log in and manage the remote guest environment by using graphical tools, such as tigervnc or virt-viewer.

Text Only

You can use the ssh command from a remote computer to log in to a virtual machine host and access its text-based console. You can then use the xl command to manage virtual machines, and the virt-install command to create new virtual machines.

22.3 VNC Viewer

VNC viewer is used to view the environment of the running guest system in a graphical way. You can use it from Dom0 (known as local access or on-box access), or from a remote computer.

You can use the IP address of a VM Host Server and a VNC viewer to view the display of this VM Guest. When a virtual machine is running, the VNC server on the host assigns the virtual machine a port number to be used for VNC viewer connections. The assigned port number is the lowest port number available when the virtual machine starts. The number is only available for the virtual machine while it is running. After shutting down, the port number might be assigned to other virtual machines.

For example, if ports 1 and 2 and 4 and 5 are assigned to the running virtual machines, the VNC viewer assigns the lowest available port number, 3. If port number 3 is still in use the next time the virtual machine starts, the VNC server assigns a different port number to the virtual machine.

To use the VNC viewer from a remote computer, the firewall must permit access to as many ports as VM Guest systems run from. This means from port 5900 and up. For example, to run 10 VM Guest systems, you need to open the TCP ports 5900:5910.

To access the virtual machine from the local console running a VNC viewer client, enter one of the following commands:

- `vncviewer ::590#`
- `vncviewer :#`

`#` is the VNC viewer port number assigned to the virtual machine.

When accessing the VM Guest from a machine other than Dom0, use the following syntax:

```
tux > vncviewer 192.168.1.20::590#
```

In this case, the IP address of Dom0 is 192.168.1.20.

22.3.1 Assigning VNC Viewer Port Numbers to Virtual Machines

Although the default behavior of VNC viewer is to assign the first available port number, you should assign a specific VNC viewer port number to a specific virtual machine.

To assign a specific port number on a VM Guest, edit the `xl` setting of the virtual machine and change the `vnclisten` to the desired value. Note that for example for port number 5902, specify 2 only, as 5900 is added automatically:

```
vfb = [ 'vnc=1,vnclisten="localhost:2" ' ]
```

For more information regarding editing the `xl` settings of a guest domain, see [Section 20.1, “XL —Xen Management Tool”](#).



Tip

Assign higher port numbers to avoid conflict with port numbers assigned by the VNC viewer, which uses the lowest available port number.

22.3.2 Using SDL instead of a VNC Viewer

If you access a virtual machine's display from the virtual machine host console (known as local or on-box access), you should use SDL instead of VNC viewer. VNC viewer is faster for viewing desktops over a network, but SDL is faster for viewing desktops from the same computer.

To set the default to use SDL instead of VNC, change the virtual machine's configuration information to the following. For instructions, see [Section 20.1, "XL—Xen Management Tool"](#).

```
vfb = [ 'sdl=1' ]
```

Remember that, unlike a VNC viewer window, closing an SDL window terminates the virtual machine.

22.4 Virtual Keyboards

When a virtual machine is started, the host creates a virtual keyboard that matches the keymap entry according to the virtual machine's settings. If there is no keymap entry specified, the virtual machine's keyboard defaults to English (US).

To view a virtual machine's current keymap entry, enter the following command on the Dom0:

```
tux > xl list -l VM_NAME | grep keymap
```

To configure a virtual keyboard for a guest, use the following snippet:

```
vfb = [ 'keymap="de"' ]
```

For a complete list of supported keyboard layouts, see the Keymaps section of the xl.cfg manual page man 5 xl.cfg.

22.5 Dedicating CPU Resources

In Xen it is possible to specify how many and which CPU cores the Dom0 or VM Guest should use to retain its performance. The performance of Dom0 is important for the overall system, as the disk and network drivers are running on it. Also I/O intensive guests' workloads may consume lots of Dom0s' CPU cycles. On the other hand, the performance of VM Guests is also important, to be able to accomplish the task they were set up for.

22.5.1 Dom0

Dedicating CPU resources to Dom0 results in a better overall performance of the virtualized environment because Dom0 has free CPU time to process I/O requests from VM Guests. Failing to dedicate exclusive CPU resources to Dom0 usually results in a poor performance and can cause the VM Guests to function incorrectly.

Dedicating CPU resources involves three basic steps: modifying Xen boot line, binding Dom0's VCPUs to a physical processor, and configuring CPU-related options on VM Guests:

1. First you need to append the `dom0_max_vcpus=X` to the Xen boot line. Do so by adding the following line to `/etc/default/grub`:

```
GRUB_CMDLINE_XEN="dom0_max_vcpus=X"
```

If `/etc/default/grub` already contains a line setting `GRUB_CMDLINE_XEN`, rather append `dom0_max_vcpus=X` to this line.

`X` needs to be replaced by the number of VCPUs dedicated to Dom0.

2. Update the GRUB 2 configuration file by running the following command:

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

3. Reboot for the change to take effect.
4. The next step is to bind (or “pin”) each Dom0's VCPU to a physical processor.

```
tux > sudo xl vcpu-pin Domain-0 0 0
xl vcpu-pin Domain-0 1 1
```

The first line binds Dom0's VCPU number 0 to the physical processor number 0, while the second line binds Dom0's VCPU number 1 to the physical processor number 1.

5. Lastly, you need to make sure no VM Guest uses the physical processors dedicated to VCPUs of Dom0. Assuming you are running an 8-CPU system, you need to add

```
cpus="2-8"
```

to the configuration file of the relevant VM Guest.

22.5.2 VM Guests

It is often necessary to dedicate specific CPU resources to a virtual machine. By default, a virtual machine uses any available CPU core. Its performance can be improved by assigning a reasonable number of physical processors to it as other VM Guests are not allowed to use them after that. Assuming a machine with 8 CPU cores while a virtual machine needs to use 2 of them, change its configuration file as follows:

```
vcpus=2  
cpus="2,3"
```

The above example dedicates 2 processors to the VM Guest, and these being the 3rd and 4th one, (2 and 3 counted from zero). If you need to assign more physical processors, use the `cpus="2-8"` syntax.

If you need to change the CPU assignment for a guest named “alice” in a hotplug manner, do the following on the related Dom0:

```
tux > sudo xl vcpu-set alice 2  
tux > sudo xl vcpu-pin alice 0 2  
tux > sudo xl vcpu-pin alice 1 3
```

The example will dedicate 2 physical processors to the guest, and bind its VCPU 0 to physical processor 2 and VCPU 1 to physical processor 3. Now check the assignment:

```
tux > sudo xl vcpu-list alice
```

Name	ID	VCPUs	CPU	State	Time(s)	CPU Affinity
alice	4	0	2	-b-	1.9	2-3
alice	4	1	3	-b-	2.8	2-3

22.6 HVM Features

In Xen some features are only available for fully virtualized domains. They are not very often used, but still may be interesting in some environments.

22.6.1 Specify Boot Device on Boot

Just as with physical hardware, it is sometimes desirable to boot a VM Guest from a different device than its own boot device. For fully virtual machines, it is possible to select a boot device with the `boot` parameter in a domain xl configuration file:

```
boot = BOOT_DEVICE
```

`BOOT_DEVICE` can be one of `c` for hard disk, `d` for CD-ROM, or `n` for Network/PXE. You can specify multiple options, and they will be attempted in the given order. For example,

```
boot = dc
```

boots from CD-ROM, and falls back to the hard disk if CD-ROM is not bootable.

22.6.2 Changing CPUIDs for Guests

To be able to migrate a VM Guest from one VM Host Server to a different VM Host Server, the VM Guest system may only use CPU features that are available on both VM Host Server systems. If the actual CPUs are different on both hosts, it may be necessary to hide some features before the VM Guest is started. This maintains the possibility to migrate the VM Guest between both hosts. For fully virtualized guests, this can be achieved by configuring the `cpuid` that is available to the guest.

To gain an overview of the current CPU, have a look at `/proc/cpuinfo`. This contains all the important information that defines the current CPU.

To redefine a CPU, first have a look at the respective cpuid definitions of the CPU vendor. These are available from:

Intel

<http://www.intel.com/Assets/PDF/appnote/241618.pdf>

```
cpuid = "host,tm=0,sse3=0"
```

The syntax is a comma-separated list of key = value pairs, preceded by the word "host". A few keys take a numerical value, while all others take a single character which describes what to do with the feature bit. See `man 5 xl.cfg` for a complete list of cpuid keys. The respective bits may be changed by using the following values:

1

- Force the corresponding bit to 1
- 0 Force the corresponding bit to 0
- x Use the values of the default policy
- k Use the values defined by the host
- s Like k, but preserve the value over migrations

Note that counting bits is done from right to left, starting with bit 0.

22.6.3 Increasing the Number of PCI-IRQs

In case you need to increase the default number of PCI-IRQs available to Dom0 and/or VM Guest, you can do so by modifying the Xen kernel command line. Use the command **extra_guest_irqs=** DOMU_IRGS,DOM0_IRGS. The optional first number DOMU_IRGS is common for all VM Guests, while the optional second number DOM0_IRGS (preceded by a comma) is for Dom0. Changing the setting for VM Guest has no impact on Dom0 and vice versa. For example to change Dom0 without changing VM Guest, use

```
extra_guest_irqs=,512
```

23 Administrative Tasks

23.1 The Boot Loader Program

The boot loader controls how the virtualization software boots and runs. You can modify the boot loader properties by using YaST, or by directly editing the boot loader configuration file.

The YaST boot loader program is located at *YaST > System > Boot Loader*. Click the *Bootloader Options* tab and select the line containing the Xen kernel as the *Default Boot Section*.

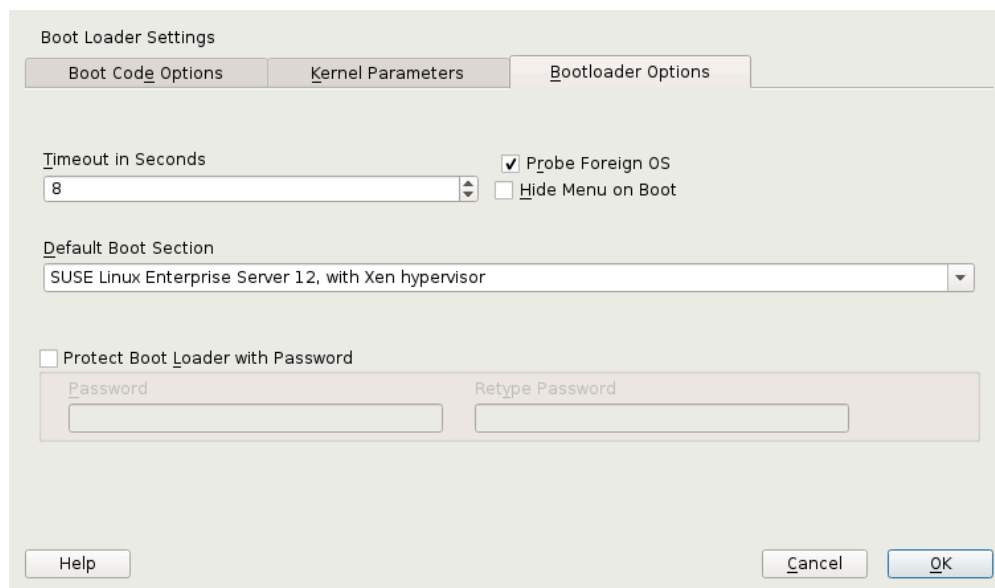


FIGURE 23.1: BOOT LOADER SETTINGS

Confirm with *OK*. Next time you boot the host, it will be ready to provide the Xen virtualization environment.

You can use the Boot Loader program to specify functionality, such as:

- Pass kernel command line parameters.
- Specify the kernel image and initial RAM disk.
- Select a specific hypervisor.
- Pass additional parameters to the hypervisor. See <http://xenbits.xen.org/docs/unstable/misc/xen-command-line.html> for their complete list.

You can customize your virtualization environment by editing the `/etc/default/grub` file. Add the following line to this file: `GRUB_CMDLINE_XEN="<boot_parameters>"`. Do not forget to run `grub2-mkconfig -o /boot/grub2/grub.cfg` after editing the file.

23.2 Sparse Image Files and Disk Space

If the host's physical disk reaches a state where it has no available space, a virtual machine using a virtual disk based on a sparse image file cannot write to its disk. Consequently, it reports I/O errors.

If this situation occurs, you should free up available space on the physical disk, remount the virtual machine's file system, and set the file system back to read-write.

To check the actual disk requirements of a sparse image file, use the command `du -h <image file>`.

To increase the available space of a sparse image file, first increase the file size and then the file system.



Warning: Back Up Before Resizing

Touching the sizes of partitions or sparse files always bears the risk of data failure. Do not work without a backup.

The resizing of the image file can be done online, while the VM Guest is running. Increase the size of a sparse image file with:

```
tux > sudo dd if=/dev/zero of=<image file> count=0 bs=1M seek=<new size in MB>
```

For example, to increase the file `/var/lib/xen/images/sles/disk0` to a size of 16GB, use the command:

```
tux > sudo dd if=/dev/zero of=/var/lib/xen/images/sles/disk0 count=0 bs=1M seek=16000
```



Note: Increasing Non-Sparse Images

It is also possible to increase the image files of devices that are not sparse files. However, you must know exactly where the previous image ends. Use the seek parameter to point to the end of the image file and use a command similar to the following:

```
tux > sudo dd if=/dev/zero of=/var/lib/xen/images/sles/disk0 seek=8000 bs=1M
count=2000
```

Be sure to use the right seek, else data loss may happen.

If the VM Guest is running during the resize operation, also resize the loop device that provides the image file to the VM Guest. First detect the correct loop device with the command:

```
tux > sudo losetup -j /var/lib/xen/images/sles/disk0
```

Then resize the loop device, for example `/dev/loop0`, with the following command:

```
tux > sudo losetup -c /dev/loop0
```

Finally check the size of the block device inside the guest system with the command `fdisk -l /dev/xvdb`. The device name depends on the actually increased device.

The resizing of the file system inside the sparse file involves tools that are depending on the actual file system.

23.3 Migrating Xen VM Guest Systems

With Xen it is possible to migrate a VM Guest system from one VM Host Server to another with almost no service interruption. This could be used for example to move a busy VM Guest to a VM Host Server that has stronger hardware or is not yet loaded. Or, if a service of a VM Host Server is required, all VM Guest systems running on this machine can be migrated to other machines to avoid interruption of service. These are only two examples—many more reasons may apply to your personal situation.

Before starting, some preliminary considerations regarding the VM Host Server should be taken into account:

- All VM Host Server systems should use a similar CPU. The frequency is not so important, but they should be using the same CPU family. To get more information about the used CPU, see `cat /proc/cpuinfo`.
- All resources that are used by a specific guest system must be available on all involved VM Host Server systems—for example all used block devices must exist on both VM Host Server systems.
- If the hosts included in the migration process run in different subnets, make sure that either DHCP relay is available to the guests, or for guests with static network configuration, set up the network manually.
- Using special features like PCI Pass-Through may be problematic. Do not implement these when deploying for an environment that should migrate VM Guest systems between different VM Host Server systems.
- For fast migrations, a fast network is mandatory. If possible, use GB Ethernet and fast switches. Deploying VLAN might also help avoid collisions.

23.3.1 Preparing Block Devices for Migrations

The block devices needed by the VM Guest system must be available on all involved VM Host Server systems. This is done by implementing some kind of shared storage that serves as container for the root file system of the migrated VM Guest system. Common possibilities include:

- iSCSI can be set up to give access to the same block devices from different systems at the same time.
- NFS is a widely used root file system that can easily be accessed from different locations. For more information, see *Book "Reference", Chapter 22 "Sharing File Systems with NFS"*.
- DRBD can be used if only two VM Host Server systems are involved. This gives some extra data security, because the used data is mirrored over the network. .

- [SCSI](#) can also be used if the available hardware permits shared access to the same disks.
- [NPIV](#) is a special mode to use Fibre channel disks. However, in this case all migration hosts must be attached to the same Fibre channel switch. For more information about NPIV, see [Section 21.1, “Mapping Physical Storage to Virtual Disks”](#). Commonly, this works if the Fibre channel environment supports 4 Gbit or faster connections.

23.3.2 Migrating VM Guest Systems

The actual migration of the VM Guest system is done with the command:

```
tux > sudo xl migrate <domain_name> <host>
```

The speed of the migration depends on how fast the memory print can be saved to disk, sent to the new VM Host Server and loaded there. This means that small VM Guest systems can be migrated faster than big systems with a lot of memory.

23.4 Monitoring Xen

For a regular operation of many virtual guests, having a possibility to check the sanity of all the different VM Guest systems is indispensable. Xen offers several tools besides the system tools to gather information about the system.



Tip: Monitoring the VM Host Server

Basic monitoring of the VM Host Server (I/O and CPU) is available via the Virtual Machine Manager. Refer to [Section 9.8.1, “Monitoring with Virtual Machine Manager”](#) for details.

23.4.1 Monitor Xen with **xentop**

The preferred terminal application to gather information about Xen virtual environment is **xentop**. Unfortunately, this tool needs a rather broad terminal, else it inserts line breaks into the display.

xentop has several command keys that can give you more information about the system that is monitored. Some of the more important are:

D

Change the delay between the refreshes of the screen.

N

Also display network statistics. Note, that only standard configurations will be displayed. If you use a special configuration like a routed network, no network will be displayed.

B

Display the respective block devices and their cumulated usage count.

For more information about xentop see the manual page man 1 xentop.



Tip: **virt-top**

libvirt offers the hypervisor-agnostic tool **virt-top**, which is recommended for monitoring VM Guests. See *Section 9.8.2, "Monitoring with virt-top"* for details.

23.4.2 Additional Tools

There are many system tools that also help monitoring or debugging a running openSUSE system. Many of these are covered in *Book "System Analysis and Tuning Guide", Chapter 2 "System Monitoring Utilities"*. Especially useful for monitoring a virtualization environment are the following tools:

ip

The command line utility **ip** may be used to monitor arbitrary network interfaces. This is especially useful if you have set up a network that is routed or applied a masqueraded network. To monitor a network interface with the name alice.0, run the following command:

```
tux > watch ip -s link show alice.0
```

bridge

In a standard setup, all the Xen VM Guest systems are attached to a virtual network bridge. **bridge** allows you to determine the connection between the bridge and the virtual network adapter in the VM Guest system. For example, the output of **bridge link** may look like the following:

```
2: eth0 state DOWN : <NO-CARRIER, ...,UP> mtu 1500 master br0
8: vnet0 state UNKNOWN : <BROADCAST, ...,LOWER_UP> mtu 1500 master virbr0 \
state forwarding priority 32 cost 100
```

This shows that there are two virtual bridges defined on the system. One is connected to the physical Ethernet device `eth0`, the other one is connected to a VLAN interface `vnet0`.

iptables-save

Especially when using masquerade networks, or if several Ethernet interfaces are set up together with a firewall setup, it may be helpful to check the current firewall rules.

The command `iptables` may be used to check all the different firewall settings. To list all the rules of a chain, or even of the complete setup, you may use the commands `iptables-save` or `iptables -S`.

23.5 Providing Host Information for VM Guest Systems

In a standard Xen environment, the VM Guest systems have only very limited information about the VM Host Server system they are running on. If a guest should know more about the VM Host Server it runs on, `vhostmd` can provide more information to selected guests. To set up your system to run `vhostmd`, proceed as follows:

1. Install the package `vhostmd` on the VM Host Server.
2. To add or remove `metric` sections from the configuration, edit the file `/etc/vhostmd/vhostmd.conf`. However, the default works well.
3. Check the validity of the `vhostmd.conf` configuration file with the command:

```
tux > cd /etc/vhostmd
tux > xmllint --postvalid --noout vhostmd.conf
```

4. Start the `vhostmd` daemon with the command `sudo systemctl start vhostmd`. If `vhostmd` should be started automatically during start-up of the system, run the command:

```
tux > sudo systemctl enable vhostmd
```

5. Attach the image file `/dev/shm/vhostmd0` to the VM Guest system named `alice` with the command:

```
tux > xl block-attach opensuse /dev/shm/vhostmd0,,xvdb,ro
```

6. Log on the VM Guest system.

7. Install the client package vm-dump-metrics.
8. Run the command **vm-dump-metrics**. To save the result to a file, use the option **-d** <filename>.

The result of the vm-dump-metrics is an XML output. The respective metric entries follow the DTD /etc/vhostmd/metric.dtd.

For more information, see the manual pages man 8 vhostmd and /usr/share/doc/vhostmd/README on the VM Host Server system. On the guest, see the manual page man 1 vm-dump-metrics.

24 XenStore: Configuration Database Shared between Domains

This section introduces basic information about XenStore, its role in the Xen environment, the directory structure of files used by XenStore, and the description of XenStore's commands.

24.1 Introduction

XenStore is a database of configuration and status information shared between VM Guests and the management tools running in Dom0. VM Guests and the management tools read and write to XenStore to convey configuration information, status updates, and state changes. The XenStore database is managed by Dom0 and supports simple operations such as reading and writing a key. VM Guests and management tools can be notified of any changes in XenStore by watching entries of interest. Note that the `xenstored` daemon is managed by the `xencommons` service. XenStore is located on Dom0 in a single database file `/var/lib/xenstored/tdb` (`tdb` represents *tree database*).

24.2 File System Interface

XenStore database content is represented by a virtual file system similar to `/proc` (for more information on `/proc`, see *Book "System Analysis and Tuning Guide", Chapter 2 "System Monitoring Utilities", Section 2.6 "The /proc File System"*). The tree has three main paths: `/vm`, `/local/domain`, and `/tool`.

- `/vm` - stores information about the VM Guest configuration.
- `/local/domain` - stores information about VM Guest on the local node.
- `/tool` - stores general information about various tools.



Tip

Each VM Guest has two different ID numbers. The *universal unique identifier* (UUID) remains the same even if the VM Guest is migrated to another machine. The *domain identifier* (DOMID) is an identification number that represents a particular running instance. It typically changes when the VM Guest is migrated to another machine.

24.2.1 XenStore Commands

The file system structure of the XenStore database can be operated with the following commands:

xenstore-ls

Displays the full dump of the XenStore database.

xenstore-read path_to_xenstore_entry

Displays the value of the specified XenStore entry.

xenstore-exists xenstore_path

Reports whether the specified XenStore path exists.

xenstore-list xenstore_path

Displays all the children entries of the specified XenStore path.

xenstore-write path_to_xenstore_entry

Updates the value of the specified XenStore entry.

xenstore-rm xenstore_path

Removes the specified XenStore entry or directory.

xenstore-chmod xenstore_path mode

Updates the read/write permission on the specified XenStore path.

xenstore-control

Sends a command to the xenstored back-end, such as triggering an integrity check.

24.2.2 /vm

The /vm path is indexed by the UUID of each VM Guest, and stores configuration information such as the number of virtual CPUs and the amount of allocated memory. There is a /vm/<uuid> directory for each VM Guest. To list the directory content, use **xenstore-list**.

```
tux > sudo xenstore-list /vm
00000000-0000-0000-0000-000000000000
9b30841b-43bc-2af9-2ed3-5a649f466d79-1
```

The first line of the output belongs to Dom0, and the second one to a running VM Guest. The following command lists all the entries related to the VM Guest:

```
tux > sudo xenstore-list /vm/9b30841b-43bc-2af9-2ed3-5a649f466d79-1
```

```
image
rtc
device
pool_name
shadow_memory
uuid
on_reboot
start_time
on_poweroff
bootloader_args
on_crash
vcpus
vcpu_avail
bootloader
name
```

To read a value of an entry, for example the number of virtual CPUs dedicated to the VM Guest, use **xenstore-read**:

```
tux > sudo xenstore-read /vm/9b30841b-43bc-2af9-2ed3-5a649f466d79-1/vcpus
1
```

A list of selected /vm/<uuid> entries follows:

uuid

UUID of the VM Guest. It does not change during the migration process.

on_reboot

Specifies whether to destroy or restart the VM Guest in response to a reboot request.

on_poweroff

Specifies whether to destroy or restart the VM Guest in response to a halt request.

on_crash

Specifies whether to destroy or restart the VM Guest in response to a crash.

vcpus

Number of virtual CPUs allocated to the VM Guest.

vcpu_avail

Bitmask of active virtual CPUs for the VM Guest. The bitmask has several bits equal to the value of vcpus, with a bit set for each online virtual CPU.

name

The name of the VM Guest.

Regular VM Guests (not Dom0) use the /vm/<uuid>/image path:

```
tux > sudo xenstore-list /vm/9b30841b-43bc-2af9-2ed3-5a649f466d79-1/image
ostype
kernel
cmdline
ramdisk
dmargs
device-model
display
```

An explanation of the used entries follows:

ostype

The OS type of the VM Guest.

kernel

The path on Dom0 to the kernel for the VM Guest.

cmdline

The kernel command line for the VM Guest used when booting.

ramdisk

The path on Dom0 to the RAM disk for the VM Guest.

dmargs

Shows arguments passed to the QEMU process. If you look at the QEMU process with **ps**, you should see the same arguments as in /vm/<uuid>/image/dmargs.

24.2.3 /local/domain/<domid>

This path is indexed by the running domain (VM Guest) ID, and contains information about the running VM Guest. Remember that the domain ID changes during VM Guest migration. The following entries are available:

vm

The path of the /vm directory for this VM Guest.

on_reboot, on_poweroff, on_crash, name

See identical options in *Section 24.2.2, "/vm"*

domid

Domain identifier for the VM Guest.

cpu

The current CPU to which the VM Guest is pinned.

cpu_weight

The weight assigned to the VM Guest for scheduling purposes. Higher weights use the physical CPUs more often.

Apart from the individual entries described above, there are also several subdirectories under /local/domain/<domid>, containing specific entries. To see all entries available, refer to [XenStore Reference \(http://wiki.xen.org/wiki/XenStore_Reference\)](http://wiki.xen.org/wiki/XenStore_Reference).

/local/domain/<domid>/memory

Contains memory information. /local/domain/<domid>/memory/target contains target memory size for the VM Guest (in kilobytes).

/local/domain/<domid>/console

Contains information about a console used by the VM Guest.

/local/domain/<domid>/backend

Contains information about all back-end devices used by the VM Guest. The path has subdirectories of its own.

/local/domain/<domid>/device

Contains information about the front-end devices for the VM Guest.

/local/domain/<domid>/device-misc

Contains miscellaneous information about devices.

/local/domain/<domid>/store

Contains information about the VM Guest's store.

25 Xen as a High-Availability Virtualization Host

Setting up two Xen hosts as a failover system has several advantages compared to a setup where every server runs on dedicated hardware.

- Failure of a single server does not cause major interruption of the service.
- A single big machine is normally way cheaper than multiple smaller machines.
- Adding new servers as needed is a trivial task.
- The usage of the server is improved, which has positive effects on the power consumption of the system.

The setup of migration for Xen hosts is described in [Section 23.3, “Migrating Xen VM Guest Systems”](#). In the following, several typical scenarios are described.

25.1 Xen HA with Remote Storage

Xen can directly provide several remote block devices to the respective Xen guest systems. These include iSCSI, NPIV, and NBD. All of these may be used to do live migrations. When a storage system is already in place, first try to use the same device type you already used in the network. If the storage system cannot be used directly but provides a possibility to offer the needed space over NFS, it is also possible to create image files on NFS. If the NFS file system is available on all Xen host systems, this method also allows live migrations of Xen guests.

When setting up a new system, one of the main considerations is whether a dedicated storage area network should be implemented. The following possibilities are available:

TABLE 25.1: XEN REMOTE STORAGE

Method	Complexity	Comments
Ethernet	low	Note that all block device traffic goes over the same Ethernet interface as the network traffic. This may be limiting the performance of the guest.

Method	Complexity	Comments
Ethernet dedicated to storage.	medium	Running the storage traffic over a dedicated Ethernet interface may eliminate a bottleneck on the server side. However, planning your own network with your own IP address range and possibly a VLAN dedicated to storage requires numerous considerations.
NPIV	high	NPIV is a method to virtualize Fibre channel connections. This is available with adapters that support a data rate of at least 4 Gbit/s and allows the setup of complex storage systems.

Typically, a 1 Gbit/s Ethernet device can fully use a typical hard disk or storage system. When using very fast storage systems, such an Ethernet device will probably limit the speed of the system.

25.2 Xen HA with Local Storage

For space or budget reasons, it may be necessary to rely on storage that is local to the Xen host systems. To still maintain the possibility of live migrations, it is necessary to build block devices that are mirrored to both Xen hosts. The software that allows this is called Distributed Replicated Block Device (DRBD).

If a system that uses DRBD to mirror the block devices or files between two Xen hosts should be set up, both hosts should use the identical hardware. If one of the hosts has slower hard disks, both hosts will suffer from this limitation.

During the setup, each of the required block devices should use its own DRBD device. The setup of such a system is quite a complex task.

25.3 Xen HA and Private Bridges

When using several guest systems that need to communicate between each other, it is possible to do this over the regular interface. However, for security reasons it may be advisable to create a bridge that is only connected to guest systems.

In an HA environment that also should support live migrations, such a private bridge must be connected to the other Xen hosts. This is possible by using dedicated physical Ethernet devices and a dedicated network.

A different implementation method is using VLAN interfaces. In that case, all the traffic goes over the regular Ethernet interface. However, the VLAN interface does not get the regular traffic, because only the VLAN packets that are tagged for the correct VLAN are forwarded.

For more information about the setup of a VLAN interface see [Section 12.1.3, “Using VLAN Interfaces”](#).

V Managing Virtual Machines with QEMU

- 26 QEMU Overview [235](#)
- 27 Setting Up a KVM VM Host Server [236](#)
- 28 Guest Installation [246](#)
- 29 Running Virtual Machines with qemu-system-ARCH [262](#)
- 30 Virtual Machine Administration Using QEMU Monitor [289](#)

26 QEMU Overview

QEMU is a fast, cross-platform open source machine emulator which can emulate a huge number of hardware architectures for you. QEMU lets you run a complete unmodified operating system (VM Guest) on top of your existing system (VM Host Server).

You can also use QEMU for debugging purposes—you can easily stop your running virtual machine, inspect its state and save and restore it later.

QEMU consists of the following parts:

- processor emulator (x86, IBM Z, PowerPC, Sparc)
- emulated devices (graphic card, network card, hard disks, mice)
- generic devices used to connect the emulated devices to the related host devices
- descriptions of the emulated machines (PC, Power Mac)
- debugger
- user interface used to interact with the emulator

QEMU is central to KVM and Xen Virtualization, where it provides the general machine emulation. Xen's usage of QEMU is somewhat hidden from the user, while KVM's usage exposes most QEMU features transparently. If the VM Guest hardware architecture is the same as the VM Host Server's architecture, QEMU can take advantage of the KVM acceleration (SUSE only supports QEMU with the KVM acceleration loaded).

Apart from providing a core virtualization infrastructure and processor-specific drivers, QEMU also provides an architecture-specific user space program for managing VM Guests. Depending on the architecture this program is one of:

- qemu-system-i386
- qemu-system-s390x
- qemu-system-x86_64

In the following this command is called qemu-system-ARCH; in examples the qemu-system-x86_64 command is used.

27 Setting Up a KVM VM Host Server

This section documents how to set up and use openSUSE Leap 15.2 as a QEMU-KVM based virtual machine host.



Tip: Resources

In general, the virtual guest system needs the same hardware resources as if it were installed on a physical machine. The more guests you plan to run on the host system, the more hardware resources—CPU, disk, memory, and network—you need to add to the VM Host Server.

27.1 CPU Support for Virtualization

To run KVM, your CPU must support virtualization, and virtualization needs to be enabled in BIOS. The file [/proc/cpuinfo](#) includes information about your CPU features.

27.2 Required Software

The KVM host requires several packages to be installed. To install all necessary packages, do the following:

1. Verify that the [yast2-vm](#) package is installed. This package is YaST's configuration tool that simplifies the installation of virtualization hypervisors.
2. Run *YaST > Virtualization > Install Hypervisor and Tools*.

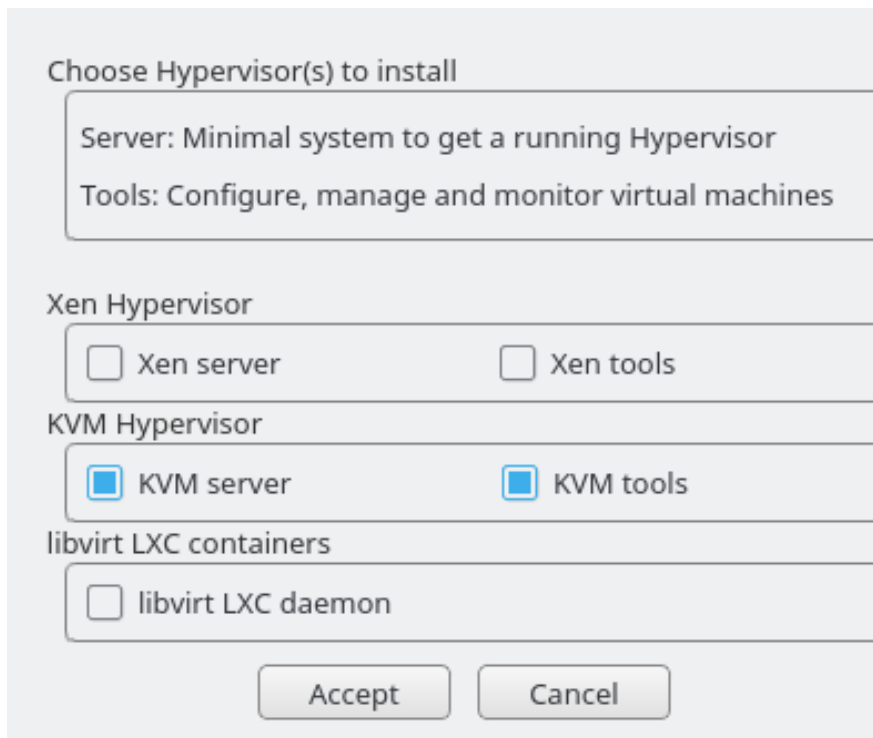


FIGURE 27.1: INSTALLING THE KVM HYPERVISOR AND TOOLS

3. Select *KVM server* and preferably also *KVM tools*, and confirm with *Accept*.
4. During the installation process, you can optionally let YaST create a *Network Bridge* for you automatically. If you do not plan to dedicate an additional physical network card to your virtual guests, network bridge is a standard way to connect the guest machines to the network.

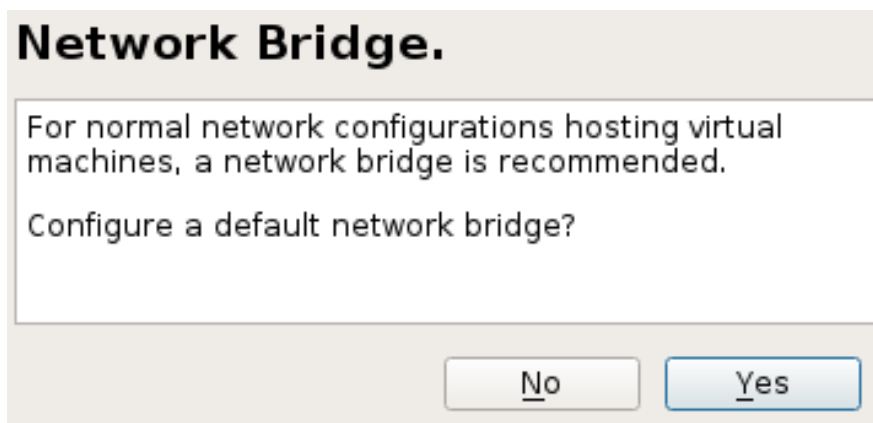


FIGURE 27.2: NETWORK BRIDGE

5. After all the required packages are installed (and new network setup activated), try to load the KVM kernel module relevant for your CPU type—`kvm-intel` or `kvm-amd`:

```
root # modprobe kvm-intel
```

Check if the module is loaded into memory:

```
tux > lsmod | grep kvm
kvm_intel          64835  6
kvm                411041  1 kvm_intel
```

Now the KVM host is ready to serve KVM VM Guests. For more information, see [Chapter 29, Running Virtual Machines with `qemu-system-ARCH`](#).

27.3 KVM Host-Specific Features

You can improve the performance of KVM-based VM Guests by letting them fully use specific features of the VM Host Server's hardware (*paravirtualization*). This section introduces techniques to make the guests access the physical host's hardware directly—without the emulation layer—to make the most use of it.



Tip

Examples included in this section assume basic knowledge of the `qemu-system-ARCH` command line options. For more information, see [Chapter 29, Running Virtual Machines with `qemu-system-ARCH`](#).

27.3.1 Using the Host Storage with `virtio-scsi`

`virtio-scsi` is an advanced storage stack for KVM. It replaces the former `virtio-blk` stack for SCSI devices pass-through. It has several advantages over `virtio-blk`:

Improved scalability

KVM guests have a limited number of PCI controllers, which results in a limited number of possibly attached devices. `virtio-scsi` solves this limitation by grouping multiple storage devices on a single controller. Each device on a `virtio-scsi` controller is represented as a logical unit, or *LUN*.

Standard command set

`virtio-blk` uses a small set of commands that need to be known to both the `virtio-blk` driver and the virtual machine monitor, and so introducing a new command requires updating both the driver and the monitor.

By comparison, `virtio-scsi` does not define commands, but rather a transport protocol for these commands following the industry-standard SCSI specification. This approach is shared with other technologies, such as Fibre Channel, ATAPI, and USB devices.

Device naming

`virtio-blk` devices are presented inside the guest as `/dev/vdX`, which is different from device names in physical systems and may cause migration problems.

`virtio-scsi` keeps the device names identical to those on physical systems, making the virtual machines easily relocatable.

SCSI device pass-through

For virtual disks backed by a whole LUN on the host, it is preferable for the guest to send SCSI commands directly to the LUN (pass-through). This is limited in `virtio-blk`, as guests need to use the `virtio-blk` protocol instead of SCSI command pass-through, and, moreover, it is not available for Windows guests. `virtio-scsi` natively removes these limitations.

27.3.1.1 `virtio-scsi` Usage

KVM supports the SCSI pass-through feature with the `virtio-scsi-pci` device:

```
root # qemu-system-x86_64 [...] \  
-device virtio-scsi-pci,id=scsi
```

27.3.2 Accelerated Networking with `vhost-net`

The `vhost-net` module is used to accelerate KVM's paravirtualized network drivers. It provides better latency and greater network throughput. Use the `vhost-net` driver by starting the guest with the following example command line:

```
root # qemu-system-x86_64 [...] \  
-netdev tap,id=guest0,vhost=on,script=no \  
-net nic,model=virtio,netdev=guest0,macaddr=00:16:35:AF:94:4B
```

Note that `guest0` is an identification string of the `vhost-driven` device.

27.3.3 Scaling Network Performance with Multiqueue virtio-net

As the number of virtual CPUs increases in VM Guests, QEMU offers a way of improving the network performance using *multiqueue*. Multiqueue virtio-net scales the network performance by allowing VM Guest virtual CPUs to transfer packets in parallel. Multiqueue support is required on both the VM Host Server and VM Guest sides.



Tip: Performance Benefit

The multiqueue virtio-net solution is most beneficial in the following cases:

- Network traffic packets are large.
- VM Guest has many connections active at the same time, mainly between the guest systems, or between the guest and the host, or between the guest and an external system.
- The number of active queues is equal to the number of virtual CPUs in the VM Guest.



Note

While multiqueue virtio-net increases the total network throughput, it increases CPU consumption as it uses of the virtual CPU's power.

PROCEDURE 27.1: HOW TO ENABLE MULTIQUEUE VIRTIO-NET

The following procedure lists important steps to enable the multiqueue feature with **qemu-system-ARCH**. It assumes that a tap network device with multiqueue capability (supported since kernel version 3.8) is set up on the VM Host Server.

1. In **qemu-system-ARCH**, enable multiqueue for the tap device:

```
-netdev tap,vhost=on,queues=2*N
```

where N stands for the number of queue pairs.

2. In **qemu-system-ARCH**, enable multiqueue and specify MSI-X (Message Signaled Interrupt) vectors for the virtio-net-pci device:

```
-device virtio-net-pci,mq=on,vectors=2*N+2
```

where the formula for the number of MSI-X vectors results from: N vectors for TX (transmit) queues, N for RX (receive) queues, one for configuration purposes, and one for possible VQ (vector quantization) control.

3. In VM Guest, enable multiqueue on the relevant network interface (`eth0` in this example):

```
tux > sudo ethtool -L eth0 combined 2*N
```

The resulting `qemu-system-ARCH` command line will look similar to the following example:

```
qemu-system-x86_64 [...] -netdev tap,id=guest0,queues=8,vhost=on \  
-device virtio-net-pci,netdev=guest0,mq=on,vectors=10
```

Note that the `id` of the network device (`guest0`) needs to be identical for both options.

Inside the running VM Guest, specify the following command with `root` privileges:

```
tux > sudo ethtool -L eth0 combined 8
```

Now the guest system networking uses the multiqueue support from the `qemu-system-ARCH` hypervisor.

27.3.4 VFIO: Secure Direct Access to Devices

Directly assigning a PCI device to a VM Guest (PCI pass-through) avoids performance issues caused by avoiding any emulation in performance-critical paths. VFIO replaces the traditional KVM PCI Pass-Through device assignment. A prerequisite for this feature is a VM Host Server configuration as described in *Important: Requirements for VFIO and SR-IOV*.

To be able to assign a PCI device via VFIO to a VM Guest, you need to find out which IOMMU Group it belongs to. The *IOMMU* (input/output memory management unit that connects a direct memory access-capable I/O bus to the main memory) API supports the notion of groups. A group is a set of devices that can be isolated from all other devices in the system. Groups are therefore the unit of ownership used by *VFIO*.

PROCEDURE 27.2: ASSIGNING A PCI DEVICE TO A VM GUEST VIA VFIO

1. Identify the host PCI device to assign to the guest.

```
tux > sudo lspci -nn  
[...]
```

```
00:10.0 Ethernet controller [0200]: Intel Corporation 82576 \
Virtual Function [8086:10ca] (rev 01)
[...]
```

Note down the device ID (00:10.0 in this case) and the vendor ID (8086:10ca).

2. Find the IOMMU group of this device:

```
tux > sudo readlink /sys/bus/pci/devices/0000\:00\:10.0/iommu_group
../../../../kernel/iommu_groups/20
```

The IOMMU group for this device is 20. Now you can check the devices belonging to the same IOMMU group:

```
tux > sudo ls -l /sys/bus/pci/devices/0000\:01\:10.0/iommu_group/devices/
[...] 0000:00:1e.0 -> ../../../../devices/pci0000:00/0000:00:1e.0
[...] 0000:01:10.0 -> ../../../../devices/pci0000:00/0000:00:1e.0/0000:01:10.0
[...] 0000:01:10.1 -> ../../../../devices/pci0000:00/0000:00:1e.0/0000:01:10.1
```

3. Unbind the device from the device driver:

```
tux > sudo echo "0000:01:10.0" > /sys/bus/pci/devices/0000\:01\:10.0/driver/unbind
```

4. Bind the device to the vfio-pci driver using the vendor ID from step 1:

```
tux > sudo echo "8086 153a" > /sys/bus/pci/drivers/vfio-pci/new_id
```

A new device /dev/vfio/IOMMU_GROUP will be created as a result, /dev/vfio/20 in this case.

5. Change the ownership of the newly created device:

```
tux > sudo chown qemu.qemu /dev/vfio/DEVICE
```

6. Now run the VM Guest with the PCI device assigned.

```
tux > sudo qemu-system-ARCH [...] -device
vfio-pci,host=00:10.0,id=ID
```

Important: No Hotplugging

As of openSUSE Leap 15.2 hotplugging of PCI devices passed to a VM Guest via VFIO is not supported.

You can find more detailed information on the *VFIO* driver in the `/usr/src/linux/Documentation/vfio.txt` file (package `kernel-source` needs to be installed).

27.3.5 VirtFS: Sharing Directories between Host and Guests

VM Guests usually run in a separate computing space—they are provided their own memory range, dedicated CPUs, and file system space. The ability to share parts of the VM Host Server's file system makes the virtualization environment more flexible by simplifying mutual data exchange. Network file systems, such as CIFS and NFS, have been the traditional way of sharing directories. But as they are not specifically designed for virtualization purposes, they suffer from major performance and feature issues.

KVM introduces a new optimized method called *VirtFS* (sometimes called “file system pass-through”). VirtFS uses a paravirtual file system driver, which avoids converting the guest application file system operations into block device operations, and then again into host file system operations.

You typically use VirtFS for the following situations:

- To access a shared directory from several guests, or to provide guest-to-guest file system access.
- To replace the virtual disk as the root file system to which the guest's RAM disk connects during the guest boot process.
- To provide storage services to different customers from a single host file system in a cloud environment.

27.3.5.1 Implementation

In QEMU, the implementation of VirtFS is simplified by defining two types of devices:

- `virtio-9p-pci` device which transports protocol messages and data between the host and the guest.
- `fsdev` device which defines the export file system properties, such as file system type and security model.

EXAMPLE 27.1: EXPORTING HOST'S FILE SYSTEM WITH VIRTFS

```
tux > sudo qemu-system-x86_64 [...] \
```

```
-fsdev local,id=expl①,path=/tmp/②,security_model=mapped③ \  
-device virtio-9p-pci,fsdev=expl④,mount_tag=v_tmp⑤
```

- ① Identification of the file system to be exported.
- ② File system path on the host to be exported.
- ③ Security model to be used—mapped keeps the guest file system modes and permissions isolated from the host, while none invokes a “pass-through” security model in which permission changes on the guest's files are reflected on the host as well.
- ④ The exported file system ID defined before with -fsdev id= .
- ⑤ Mount tag used later on the guest to mount the exported file system.

Such an exported file system can be mounted on the guest as follows:

```
tux > sudo mount -t 9p -o trans=virtio v_tmp /mnt
```

where v_tmp is the mount tag defined earlier with -device mount_tag= and /mnt is the mount point where you want to mount the exported file system.

27.3.6 KSM: Sharing Memory Pages between Guests

Kernel Same Page Merging (*KSM*) is a Linux kernel feature that merges identical memory pages from multiple running processes into one memory region. Because KVM guests run as processes under Linux, *KSM* provides the memory overcommit feature to hypervisors for more efficient use of memory. Therefore, if you need to run multiple virtual machines on a host with limited memory, *KSM* may be helpful to you.

KSM stores its status information in the files under the /sys/kernel/mm/ksm directory:

```
tux > ls -l /sys/kernel/mm/ksm  
full_scans  
merge_across_nodes  
pages_shared  
pages_sharing  
pages_to_scan  
pages_unshared  
pages_volatile  
run  
sleep_millisecs
```

For more information on the meaning of the /sys/kernel/mm/ksm/* files, see /usr/src/linux/Documentation/vm/ksm.txt (package kernel-source).

To use *KSM*, do the following.

1. Although openSUSE Leap includes *KSM* support in the kernel, it is disabled by default. To enable it, run the following command:

```
root # echo 1 > /sys/kernel/mm/ksm/run
```

2. Now run several VM Guests under KVM and inspect the content of files pages_sharing and pages_shared, for example:

```
tux > while [ 1 ]; do cat /sys/kernel/mm/ksm/pages_shared; sleep 1; done
13522
13523
13519
13518
13520
13520
13528
```

28 Guest Installation

The `libvirt`-based tools such as `virt-manager` and `virt-install` offer convenient interfaces to set up and manage virtual machines. They act as a kind of wrapper for the `qemu-system-ARCH` command. However, it is also possible to use `qemu-system-ARCH` directly without using `libvirt`-based tools.



Warning: `qemu-system-ARCH` and `libvirt`

Virtual Machines created with `qemu-system-ARCH` are not "visible" for the `libvirt`-based tools.

28.1 Basic Installation with `qemu-system-ARCH`

In the following example, a virtual machine for a SUSE Linux Enterprise Server 11 installation is created. For detailed information on the commands, refer to the respective man pages.

If you do not already have an image of a system that you want to run in a virtualized environment, you need to create one from the installation media. In such case, you need to prepare a hard disk image, and obtain an image of the installation media or the media itself.

Create a hard disk with `qemu-img`.

```
tux > qemu-img create ① -f raw ② /images/sles/hda ③ 8G ④
```

- ① The subcommand `create` tells `qemu-img` to create a new image.
- ② Specify the disk's format with the `-f` parameter.
- ③ The full path to the image file.
- ④ The size of the image—8 GB in this case. The image is created as a *Sparse image file* that grows when the disk is filled with data. The specified size defines the maximum size to which the image file can grow.

After at least one hard disk image is created, you can set up a virtual machine with `qemu-system-ARCH` that will boot into the installation system:

```
root # qemu-system-x86_64 -name "sles" ① -machine accel=kvm -M pc ② -m 768 ③ \  
-smp 2 ④ -boot d ⑤ \  
-drive file=/images/sles/hda,if=virtio,index=0,media=disk,format=raw ⑥ \  
-drive file=/isos/SLE-15-SP2-Online-ARCH-GM-media1.iso,index=1,media=cdrom ⑦ \  
-
```



```
-net nic,model=virtio,macaddr=52:54:00:05:11:11 ⑧ -net user \  
-vga cirrus ⑨ -balloon virtio ⑩
```

- ① Name of the virtual machine that will be displayed in the window caption and be used for the VNC server. This name must be unique.
- ② Specifies the machine type. Use `qemu-system-ARCH -M ?` to display a list of valid parameters. `pc` is the default *Standard PC*.
- ③ Maximum amount of memory for the virtual machine.
- ④ Defines an SMP system with two processors.
- ⑤ Specifies the boot order. Valid values are `a`, `b` (floppy 1 and 2), `c` (first hard disk), `d` (first CD-ROM), or `n` to `p` (Ether-boot from network adapter 1-3). Defaults to `c`.
- ⑥ Defines the first (`index=0`) hard disk. It will be accessed as a paravirtualized (`if=virtio`) drive in `raw` format.
- ⑦ The second (`index=1`) image drive will act as a CD-ROM.
- ⑧ Defines a paravirtualized (`model=virtio`) network adapter with the MAC address `52:54:00:05:11:11`. Be sure to specify a unique MAC address, otherwise a network conflict may occur.
- ⑨ Specifies the graphic card. If you specify `none`, the graphic card will be disabled.
- ⑩ Defines the paravirtualized balloon device that allows to dynamically change the amount of memory (up to the maximum value specified with the parameter `-m`).

After the installation of the guest operating system finishes, you can start the related virtual machine without the need to specify the CD-ROM device:

```
root # qemu-system-x86_64 -name "sles" -machine type=pc,accel=kvm -m 768 \  
-smp 2 -boot c \  
-drive file=/images/sles/hda,if=virtio,index=0,media=disk,format=raw \  
-net nic,model=virtio,macaddr=52:54:00:05:11:11 \  
-vga cirrus -balloon virtio
```

28.2 Managing Disk Images with `qemu-img`

In the previous section (see [Section 28.1, "Basic Installation with `qemu-system-ARCH`"](#)), we used the `qemu-img` command to create an image of a hard disk. You can, however, use `qemu-img` for general disk image manipulation. This section introduces `qemu-img` subcommands to help manage the disk images flexibly.

28.2.1 General Information on qemu-img Invocation

qemu-img uses subcommands (like **zypper** does) to do specific tasks. Each subcommand understands a different set of options. Some options are general and used by more of these subcommands, while some are unique to the related subcommand. See the qemu-img manual page (**man 1 qemu-img**) for a list of all supported options. **qemu-img** uses the following general syntax:

```
tux > qemu-img subcommand [options]
```

and supports the following subcommands:

create

Creates a new disk image on the file system.

check

Checks an existing disk image for errors.

compare

Check if two images have the same content.

map

Dumps the metadata of the image file name and its backing file chain.

amend

Amends the image format specific options for the image file name.

convert

Converts an existing disk image to a new one in a different format.

info

Displays information about the relevant disk image.

snapshot

Manages snapshots of existing disk images.

commit

Applies changes made to an existing disk image.

rebase

Creates a new base image based on an existing image.

resize

Increases or decreases the size of an existing image.

28.2.2 Creating, Converting and Checking Disk Images

This section describes how to create disk images, check their condition, convert a disk image from one format to another, and get detailed information about a particular disk image.

28.2.2.1 `qemu-img create`

Use **`qemu-img create`** to create a new disk image for your VM Guest operating system. The command uses the following syntax:

```
tux > qemu-img create -f fmt❶ -o options❷ fname❸ size❹
```

- ❶ The format of the target image. Supported formats are `raw` and `qcow2`.
- ❷ Some image formats support additional options to be passed on the command line. You can specify them here with the `-o` option. The `raw` image format supports only the `size` option, so it is possible to insert `-o size=8G` instead of adding the size option at the end of the command.
- ❸ Path to the target disk image to be created.
- ❹ Size of the target disk image (if not already specified with the `-o size=<image_size>` option. Optional suffixes for the image size are `K` (kilobyte), `M` (megabyte), `G` (gigabyte), or `T` (terabyte).

To create a new disk image `sles.raw` in the directory `/images` growing up to a maximum size of 4 GB, run the following command:

```
tux > qemu-img create -f raw -o size=4G /images/sles.raw
Formatting '/images/sles.raw', fmt=raw size=4294967296

tux > ls -l /images/sles.raw
-rw-r--r-- 1 tux users 4294967296 Nov 15 15:56 /images/sles.raw

tux > qemu-img info /images/sles.raw
image: /images/sles11.raw
file format: raw
virtual size: 4.0G (4294967296 bytes)
disk size: 0
```

As you can see, the *virtual* size of the newly created image is 4 GB, but the actual reported disk size is 0 as no data has been written to the image yet.



Tip: VM Guest Images on the Btrfs File System

If you need to create a disk image on the Btrfs file system, you can use `nocow=on` to reduce the performance overhead created by the copy-on-write feature of Btrfs:

```
tux > qemu-img create -o nocow=on test.img 8G
```

If you, however, want to use copy-on-write (for example for creating snapshots or sharing them across virtual machines), then leave the command line without the `nocow` option.

28.2.2.2 `qemu-img convert`

Use `qemu-img convert` to convert disk images to another format. To get a complete list of image formats supported by QEMU, run `qemu-img -h` and look at the last line of the output. The command uses the following syntax:

```
tux > qemu-img convert -c ① -f fmt ② -O out_fmt ③ -o options ④ fname ⑤ out_fname ⑥
```

- ① Applies compression on the target disk image. Only `qcow` and `qcow2` formats support compression.
- ② The format of the source disk image. It is usually autodetected and can therefore be omitted.
- ③ The format of the target disk image.
- ④ Specify additional options relevant for the target image format. Use `-o ?` to view the list of options supported by the target image format.
- ⑤ Path to the source disk image to be converted.
- ⑥ Path to the converted target disk image.

```
tux > qemu-img convert -O vmdk /images/sles.raw \  
/images/sles.vmdk  
  
tux > ls -l /images/  
-rw-r--r-- 1 tux users 4294967296 16. lis 10.50 sles.raw  
-rw-r--r-- 1 tux users 2574450688 16. lis 14.18 sles.vmdk
```

To see a list of options relevant for the selected target image format, run the following command (replace `vmdk` with your image format):

```
tux > qemu-img convert -O vmdk /images/sles.raw \
/images/sles.vmdk -o ?
Supported options:
size                Virtual disk size
backing_file        File name of a base image
compat6            VMDK version 6 image
subformat          VMDK flat extent format, can be one of {monolithicSparse \
                  (default) | monolithicFlat | twoGbMaxExtentSparse | twoGbMaxExtentFlat}
scsi               SCSI image
```

28.2.2.3 `qemu-img check`

Use `qemu-img check` to check the existing disk image for errors. Not all disk image formats support this feature. The command uses the following syntax:

```
tux > qemu-img check -f fmt ❶ fname ❷
```

- ❶ The format of the source disk image. It is usually autodetected and can therefore be omitted.
- ❷ Path to the source disk image to be checked.

If no error is found, the command returns no output. Otherwise, the type and number of errors found is shown.

```
tux > qemu-img check -f qcow2 /images/sles.qcow2
ERROR: invalid cluster offset=0x2af0000
[...]
ERROR: invalid cluster offset=0x34ab0000
378 errors were found on the image.
```

28.2.2.4 Increasing the Size of an Existing Disk Image

When creating a new image, you must specify its maximum size before the image is created (see [Section 28.2.2.1, "qemu-img create"](#)). After you have installed the VM Guest and have been using it for some time, the initial size of the image may no longer be sufficient. In that case, add more space to it.

To increase the size of an existing disk image by 2 gigabytes, use:

```
tux > qemu-img resize /images/sles.raw +2GB
```



Note

You can resize the disk image using the formats raw and qcow2. To resize an image in another format, convert it to a supported format with qemu-img convert first.

The image now contains an empty space of 2 GB after the final partition. You can resize the existing partitions or add new ones.

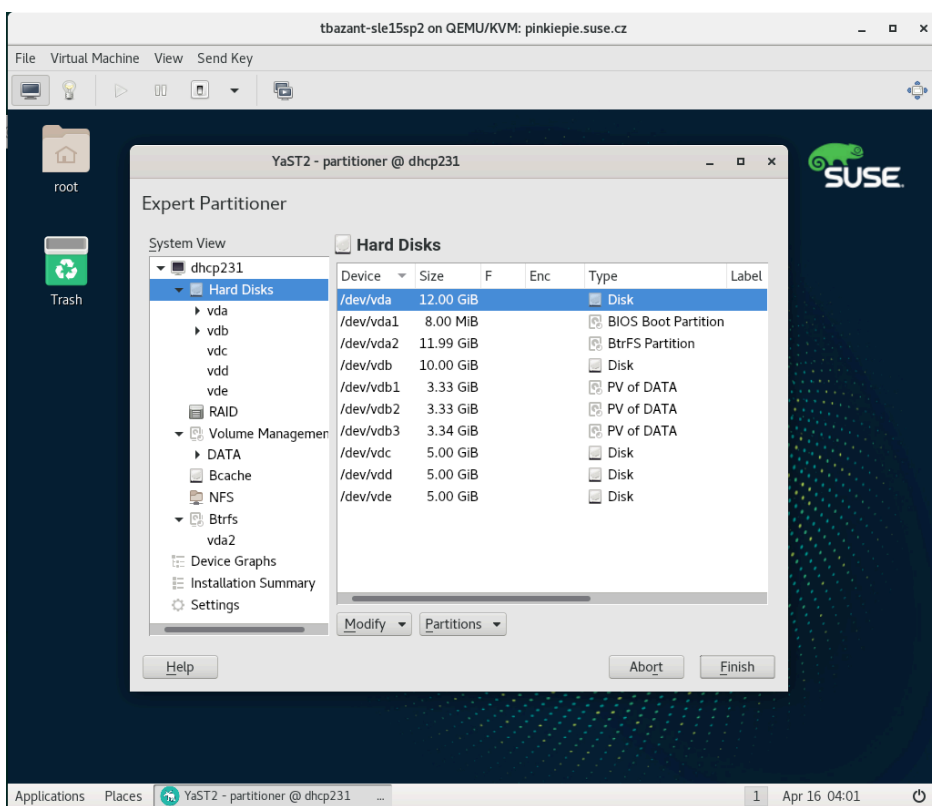


FIGURE 28.1: NEW 2 GB PARTITION IN GUEST YAST PARTITIONER

28.2.2.5 Advanced Options for the qcow2 File Format

qcow2 is the main disk image format used by QEMU. Its size grows on demand, and the disk space is only allocated when it is actually needed by the virtual machine.

A qcow2 formatted file is organized in units of constant size. These units are called *clusters*. Viewed from the guest side, the virtual disk is also divided into clusters of the same size. QEMU defaults to 64 kB clusters, but you can specify a different value when creating a new image:

```
tux > qemu-img create -f qcow2 -o cluster_size=128K virt_disk.qcow2 4G
```

A qcow2 image contains a set of tables organized in two levels that are called the L1 and L2 tables. There is just one L1 table per disk image, while there can be many L2 tables depending on how big the image is.

To read or write data to the virtual disk, QEMU needs to read its corresponding L2 table to find out the relevant data location. Because reading the table for each I/O operation consumes system resources, QEMU keeps a cache of L2 tables in memory to speed up disk access.

28.2.2.5.1 Choosing the Right Cache Size

The cache size relates to the amount of allocated space. L2 cache can map the following amount of virtual disk:

```
disk_size = l2_cache_size * cluster_size / 8
```

With the default 64 kB of cluster size, that is

```
disk_size = l2_cache_size * 8192
```

Therefore, to have a cache that maps n gigabytes of disk space with the default cluster size, you need

```
l2_cache_size = disk_size_GB * 131072
```

QEMU uses 1 MB (1048576 bytes) of L2 cache by default. Following the above formulas, 1 MB of L2 cache covers 8 GB (1048576 / 131072) of virtual disk. This means that the performance is fine with the default L2 cache size if your virtual disk size is up to 8 GB. For larger disks, you can speed up the disk access by increasing the L2 cache size.

28.2.2.5.2 Configuring the Cache Size

You can use the `-drive` option on the QEMU command line to specify the cache sizes. Alternatively when communicating via QMP, use the `blockdev-add` command. For more information on QMP, see [Section 30.11, "QMP - QEMU Machine Protocol"](#).

The following options configure the cache size for the virtual guest:

l2-cache-size

The maximum size of the L2 table cache.

refcount-cache-size

The maximum size of the *refcount* block cache. For more information on *refcount*, see <https://raw.githubusercontent.com/qemu/qemu/master/docs/qcow2-cache.txt>.

cache-size

The maximum size of both caches combined.

When specifying values for the options above, be aware of the following:

- The size of both the L2 and refcount block caches needs to be a multiple of the cluster size.
- If you only set one of the options, QEMU will automatically adjust the other options so that the L2 cache is 4 times bigger than the refcount cache.

The refcount cache is used much less often than the L2 cache, therefore you can keep it relatively small:

```
root # qemu-system-ARCH [...] \  
-drive file=disk_image.qcow2,l2-cache-size=4194304,refcount-cache-size=262144
```

28.2.2.5.3 Reducing the Memory Usage

The larger the cache, the more memory it consumes. There is a separate L2 cache for each qcow2 file. When using a lot of big disk images, you will probably need a considerably large amount of memory. Memory consumption is even worse if you add backing files (*Section 28.2.4, “Manipulate Disk Images Effectively”*) and snapshots (see *Section 28.2.3, “Managing Snapshots of Virtual Machines with qemu-img”*) to the guest's setup chain.

That is why QEMU introduced the `cache-clean-interval` setting. It defines an interval in seconds after which all cache entries that have not been accessed are removed from memory.

The following example removes all unused cache entries every 10 minutes:

```
root # qemu-system-ARCH [...] -drive file=hd.qcow2,cache-clean-interval=600
```

If this option is not set, the default value is 0 and it disables this feature.

28.2.3 Managing Snapshots of Virtual Machines with qemu-img

Virtual Machine snapshots are snapshots of the complete environment in which a VM Guest is running. The snapshot includes the state of the processor (CPU), memory (RAM), devices, and all writable disks.

Snapshots are helpful when you need to save your virtual machine in a particular state. For example, after you configured network services on a virtualized server and want to quickly start the virtual machine in the same state you last saved it. Or you can create a snapshot after the virtual machine has been powered off to create a backup state before you try something experimental and possibly make VM Guest unstable. This section introduces the latter case, while the former is described in *Chapter 30, Virtual Machine Administration Using QEMU Monitor*.

To use snapshots, your VM Guest must contain at least one writable hard disk image in `qcow2` format. This device is usually the first virtual hard disk.

Virtual Machine snapshots are created with the `savevm` command in the interactive QEMU monitor. To make identifying a particular snapshot easier, you can assign it a *tag*. For more information on QEMU monitor, see *Chapter 30, Virtual Machine Administration Using QEMU Monitor*.

Once your `qcow2` disk image contains saved snapshots, you can inspect them with the `qemu-img snapshot` command.



Warning: Shut Down the VM Guest

Do not create or delete virtual machine snapshots with the `qemu-img snapshot` command while the virtual machine is running. Otherwise, you may damage the disk image with the state of the virtual machine saved.

28.2.3.1 Listing Existing Snapshots

Use `qemu-img snapshot -l DISK_IMAGE` to view a list of all existing snapshots saved in the `disk_image` image. You can get the list even while the VM Guest is running.

```
tux > qemu-img snapshot -l /images/sles.qcow2
Snapshot list:
ID ①      TAG ②      VM SIZE ③      DATE ④      VM CLOCK ⑤
1      booting      4.4M 2013-11-22 10:51:10  00:00:20.476
2      booted      184M 2013-11-22 10:53:03  00:02:05.394
3      logged_in   273M 2013-11-22 11:00:25  00:04:34.843
4      ff_and_term_running 372M 2013-11-22 11:12:27  00:08:44.965
```

- ① Unique identification number of the snapshot. Usually auto-incremented.
- ② Unique description string of the snapshot. It is meant as a human-readable version of the ID.
- ③ The disk space occupied by the snapshot. Note that the more memory is consumed by running applications, the bigger the snapshot is.
- ④ Time and date the snapshot was created.
- ⑤ The current state of the virtual machine's clock.

28.2.3.2 Creating Snapshots of a Powered-Off Virtual Machine

Use `qemu-img snapshot -c SNAPSHOT_TITLE DISK_IMAGE` to create a snapshot of the current state of a virtual machine that was previously powered off.

```
tux > qemu-img snapshot -c backup_snapshot /images/sles.qcow2
```

```
tux > qemu-img snapshot -l /images/sles.qcow2
Snapshot list:
ID      TAG                VM SIZE          DATE            VM CLOCK
1       booting            4.4M 2013-11-22 10:51:10    00:00:20.476
2       booted             184M 2013-11-22 10:53:03    00:02:05.394
3       logged_in          273M 2013-11-22 11:00:25    00:04:34.843
4       ff_and_term_running 372M 2013-11-22 11:12:27    00:08:44.965
5       backup_snapshot    0 2013-11-22 14:14:00    00:00:00.000
```

If something breaks in your VM Guest and you need to restore the state of the saved snapshot (ID 5 in our example), power off your VM Guest and execute the following command:

```
tux > qemu-img snapshot -a 5 /images/sles.qcow2
```

The next time you run the virtual machine with `qemu-system-ARCH`, it will be in the state of snapshot number 5.



Note

The `qemu-img snapshot -c` command is not related to the `savevm` command of QEMU monitor (see *Chapter 30, Virtual Machine Administration Using QEMU Monitor*). For example, you cannot apply a snapshot with `qemu-img snapshot -a` on a snapshot created with `savevm` in QEMU's monitor.

28.2.3.3 Deleting Snapshots

Use `qemu-img snapshot -d SNAPSHOT_ID DISK_IMAGE` to delete old or unneeded snapshots of a virtual machine. This saves some disk space inside the `qcow2` disk image as the space occupied by the snapshot data is restored:

```
tux > qemu-img snapshot -d 2 /images/sles.qcow2
```

28.2.4 Manipulate Disk Images Effectively

Imagine the following real-life situation: you are a server administrator who runs and manages several virtualized operating systems. One group of these systems is based on one specific distribution, while another group (or groups) is based on different versions of the distribution or even on a different (and maybe non-Unix) platform. To make the case even more complex, individual virtual guest systems based on the same distribution usually differ according to the department and deployment. A file server typically uses a different setup and services than a Web server does, while both may still be based on openSUSE.

With QEMU it is possible to create “base” disk images. You can use them as template virtual machines. These base images will save you plenty of time because you will never need to install the same operating system more than once.

28.2.4.1 Base and Derived Images

First, build a disk image as usual and install the target system on it. For more information, see [Section 28.1, “Basic Installation with `qemu-system-ARCH`”](#) and [Section 28.2.2, “Creating, Converting and Checking Disk Images”](#). Then build a new image while using the first one as a base image. The base image is also called a *backing* file. After your new *derived* image is built, never boot the base image again, but boot the derived image instead. Several derived images may depend on one base image at the same time. Therefore, changing the base image can damage the dependencies. While using your derived image, QEMU writes changes to it and uses the base image only for reading.

It is a good practice to create a base image from a freshly installed (and, if needed, registered) operating system with no patches applied and no additional applications installed or removed. Later on, you can create another base image with the latest patches applied and based on the original base image.

28.2.4.2 Creating Derived Images



Note

While you can use the `raw` format for base images, you cannot use it for derived images because the `raw` format does not support the `backing_file` option. Use for example the `qcow2` format for the derived images.

For example, `/images/sles_base.raw` is the base image holding a freshly installed system.

```
tux > qemu-img info /images/sles_base.raw
image: /images/sles_base.raw
file format: raw
virtual size: 4.0G (4294967296 bytes)
disk size: 2.4G
```

The image's reserved size is 4 GB, the actual size is 2.4 GB, and its format is `raw`. Create an image derived from the `/images/sles_base.raw` base image with:

```
tux > qemu-img create -f qcow2 /images/sles_derived.qcow2 \
-o backing_file=/images/sles_base.raw
Formatting '/images/sles_derived.qcow2', fmt=qcow2 size=4294967296 \
backing_file='/images/sles_base.raw' encryption=off cluster_size=0
```

Look at the derived image details:

```
tux > qemu-img info /images/sles_derived.qcow2
image: /images/sles_derived.qcow2
file format: qcow2
virtual size: 4.0G (4294967296 bytes)
disk size: 140K
cluster_size: 65536
backing file: /images/sles_base.raw \
(actual path: /images/sles_base.raw)
```

Although the reserved size of the derived image is the same as the size of the base image (4 GB), the actual size is 140 KB only. The reason is that only changes made to the system inside the derived image are saved. Run the derived virtual machine, register it, if needed, and apply the latest patches. Do any other changes in the system such as removing unneeded or installing new software packages. Then shut the VM Guest down and examine its details once more:

```
tux > qemu-img info /images/sles_derived.qcow2
image: /images/sles_derived.qcow2
file format: qcow2
```

```
virtual size: 4.0G (4294967296 bytes)
disk size: 1.1G
cluster_size: 65536
backing file: /images/sles_base.raw \
(actual path: /images/sles_base.raw)
```

The `disk size` value has grown to 1.1 GB, which is the disk space occupied by the changes on the file system compared to the base image.

28.2.4.3 Rebasing Derived Images

After you have modified the derived image (applied patches, installed specific applications, changed environment settings, etc.), it reaches the desired state. At that point, you can merge the original base image and the derived image to create a new base image.

Your original base image (`/images/sles_base.raw`) holds a freshly installed system. It can be a template for new modified base images, while the new one can contain the same system as the first one plus all security and update patches applied, for example. After you have created this new base image, you can use it as a template for more specialized derived images as well. The new base image becomes independent of the original one. The process of creating base images from derived ones is called *rebasing*:

```
tux > qemu-img convert /images/sles_derived.qcow2 \
-O raw /images/sles_base2.raw
```

This command created the new base image `/images/sles_base2.raw` using the `raw` format.

```
tux > qemu-img info /images/sles_base2.raw
image: /images/sles11_base2.raw
file format: raw
virtual size: 4.0G (4294967296 bytes)
disk size: 2.8G
```

The new image is 0.4 gigabytes bigger than the original base image. It uses no backing file, and you can easily create new derived images based upon it. This lets you create a sophisticated hierarchy of virtual disk images for your organization, saving a lot of time and work.

28.2.4.4 Mounting an Image on a VM Host Server

It can be useful to mount a virtual disk image under the host system. It is strongly recommended to read [Chapter 17, *libguestfs*](#) and use dedicated tools to access a virtual machine image. However, if you need to do this manually, follow this guide.

Linux systems can mount an internal partition of a raw disk image using a loopback device. The first example procedure is more complex but more illustrative, while the second one is straightforward:

PROCEDURE 28.1: MOUNTING DISK IMAGE BY CALCULATING PARTITION OFFSET

1. Set a *loop* device on the disk image whose partition you want to mount.

```
tux > losetup /dev/loop0 /images/sles_base.raw
```

2. Find the *sector size* and the starting *sector number* of the partition you want to mount.

```
tux > fdisk -lu /dev/loop0

Disk /dev/loop0: 4294 MB, 4294967296 bytes
255 heads, 63 sectors/track, 522 cylinders, total 8388608 sectors
Units = sectors of 1 * 512 = 512 ① bytes
Disk identifier: 0x000ceca8

   Device Boot      Start         End      Blocks   Id  System
/dev/loop0p1                63      1542239     771088+  82  Linux swap
/dev/loop0p2    *    1542240 ②      8385929     3421845  83  Linux
```

- ① The disk sector size.
- ② The starting sector of the partition.

3. Calculate the partition start offset:

```
sector_size * sector_start = 512 * 1542240 = 789626880
```

4. Delete the loop and mount the partition inside the disk image with the calculated offset on a prepared directory.

```
tux > losetup -d /dev/loop0
tux > mount -o loop,offset=789626880 \
/images/sles_base.raw /mnt/sles/
tux > ls -l /mnt/sles/
total 112
drwxr-xr-x  2 root root  4096 Nov 16 10:02 bin
drwxr-xr-x  3 root root  4096 Nov 16 10:27 boot
drwxr-xr-x  5 root root  4096 Nov 16 09:11 dev
[...]
drwxrwxrwt 14 root root  4096 Nov 24 09:50 tmp
drwxr-xr-x 12 root root  4096 Nov 16 09:16 usr
drwxr-xr-x 15 root root  4096 Nov 16 09:22 var
```

5. Copy one or more files onto the mounted partition and unmount it when finished.

```
tux > cp /etc/X11/xorg.conf /mnt/sles/root/tmp
tux > ls -l /mnt/sles/root/tmp
tux > umount /mnt/sles/
```



Warning: Do not Write to Images Currently in Use

Never mount a partition of an image of a running virtual machine in a read-write mode. This could corrupt the partition and break the whole VM Guest.

29 Running Virtual Machines with `qemu-system-ARCH`

Once you have a virtual disk image ready (for more information on disk images, see [Section 28.2, “Managing Disk Images with `qemu-img`”](#)), it is time to start the related virtual machine. [Section 28.1, “Basic Installation with `qemu-system-ARCH`”](#) introduced simple commands to install and run a VM Guest. This chapter focuses on a more detailed explanation of `qemu-system-ARCH` usage, and shows solutions for more specific tasks. For a complete list of `qemu-system-ARCH`'s options, see its manual page (`man 1 qemu`).

29.1 Basic `qemu-system-ARCH` Invocation

The `qemu-system-ARCH` command uses the following syntax:

```
qemu-system-ARCH options ❶ disk_img ❷
```

- ❶ `qemu-system-ARCH` understands many options. Most of them define parameters of the emulated hardware, while others affect more general emulator behavior. If you do not supply any options, default values are used, and you need to supply the path to a disk image to be run.
- ❷ Path to the disk image holding the guest system you want to virtualize. `qemu-system-ARCH` supports many image formats. Use `qemu-img --help` to list them. If you do not supply the path to a disk image as a separate argument, you need to use the `-drive file=` option.

29.2 General `qemu-system-ARCH` Options

This section introduces general `qemu-system-ARCH` options and options related to the basic emulated hardware, such as the virtual machine's processor, memory, model type, or time processing methods.

`-name NAME_OF_GUEST`

Specifies the name of the running guest system. The name is displayed in the window caption and used for the VNC server.

`-boot OPTIONS`

Specifies the order in which the defined drives will be booted. Drives are represented by letters, where a and b stand for the floppy drives 1 and 2, c stands for the first hard disk, d stands for the first CD-ROM drive, and n to p stand for Ether-boot network adapters. For example, `qemu-system-ARCH [...] -boot order=ndc` first tries to boot from network, then from the first CD-ROM drive, and finally from the first hard disk.

-pidfile FILENAME

Stores the QEMU's process identification number (PID) in a file. This is useful if you run QEMU from a script.

-nodefaults

By default QEMU creates basic virtual devices even if you do not specify them on the command line. This option turns this feature off, and you must specify every single device manually, including graphical and network cards, parallel or serial ports, or virtual consoles. Even QEMU monitor is not attached by default.

-daemonize

“Daemonizes” the QEMU process after it is started. QEMU will detach from the standard input and standard output after it is ready to receive connections on any of its devices.



Note: SeaBIOS BIOS Implementation

SeaBIOS is the default BIOS used. You can boot USB devices, any drive (CD-ROM, Floppy, or a hard disk). It has USB mouse and keyboard support and supports multiple VGA cards. For more information about SeaBIOS, refer to the [SeaBIOS Website \(https://www.seabios.org/SeaBIOS\)](https://www.seabios.org/SeaBIOS).

29.2.1 Basic Virtual Hardware

29.2.1.1 Machine Type

You can specify the type of the emulated machine. Run `qemu-system-ARCH -M help` to view a list of supported machine types.



Note: ISA-PC

The machine type `isapc: ISA-only-PC` is unsupported.

29.2.1.2 CPU Model

To specify the type of the processor (CPU) model, run `qemu-system-ARCH -cpu MODEL`. Use `qemu-system-ARCH -cpu help` to view a list of supported CPU models.

CPU flags information can be found at [CPUID Wikipedia \(http://en.wikipedia.org/wiki/CPUID\)](http://en.wikipedia.org/wiki/CPUID).

29.2.1.3 Other Basic Options

The following is a list of most commonly used options while launching *qemu* from command line. To see all options available refer to *qemu-doc* man page.

-m MEGABYTES

Specifies how many megabytes are used for the virtual RAM size.

-balloon virtio

Specifies a paravirtualized device to dynamically change the amount of virtual RAM memory assigned to VM Guest. The top limit is the amount of memory specified with -m.

-smp NUMBER_OF_CPUS

Specifies how many CPUs will be emulated. QEMU supports up to 255 CPUs on the PC platform (up to 64 with KVM acceleration used). This option also takes other CPU-related parameters, such as number of *sockets*, number of *cores* per socket, or number of *threads* per core.

The following is an example of a working `qemu-system-ARCH` command line:

```
tux > qemu-system-x86_64 -name "SLES 12 SP2" -M pc-i440fx-2.7 -m 512 \  
-machine accel=kvm -cpu kvm64 -smp 2 -drive format=raw,file=/images/sles.raw
```

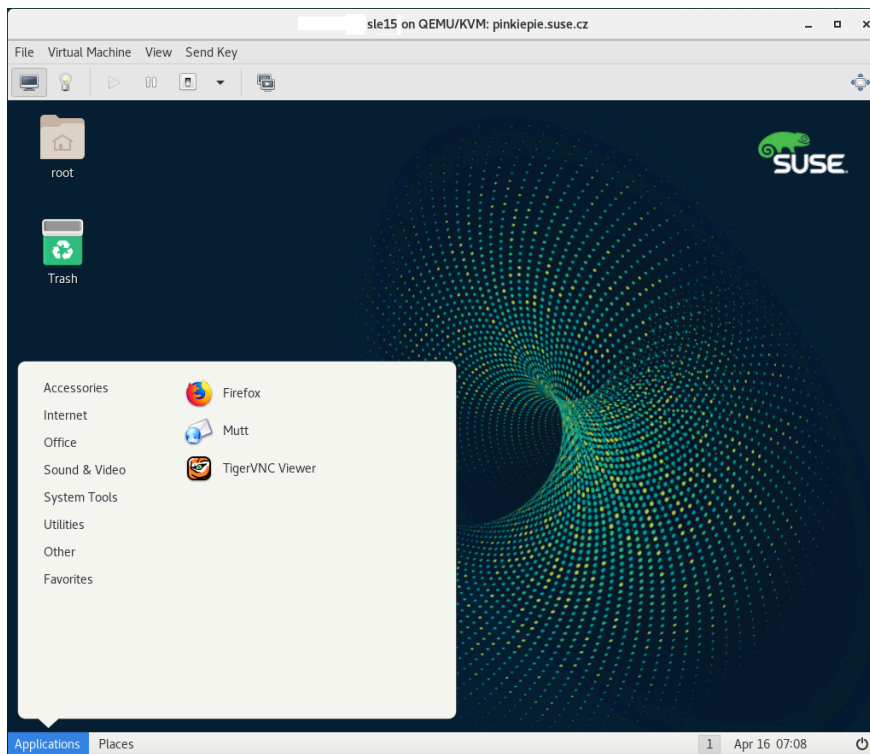


FIGURE 29.1: QEMU WINDOW WITH SLES AS VM GUEST

-no-acpi

Disables *ACPI* support.

-S

QEMU starts with CPU stopped. To start CPU, enter c in QEMU monitor. For more information, see *Chapter 30, Virtual Machine Administration Using QEMU Monitor*.

29.2.2 Storing and Reading Configuration of Virtual Devices

-readconfig CFG_FILE

Instead of entering the devices configuration options on the command line each time you want to run VM Guest, **qemu-system-ARCH** can read it from a file that was either previously saved with -writeconfig or edited manually.

-writeconfig CFG_FILE

Dumps the current virtual machine's devices configuration to a text file. It can be consequently re-used with the -readconfig option.

```
tux > qemu-system-x86_64 -name "SLES 12 SP2" -machine accel=kvm -M pc-i440fx-2.7 -m 512 -cpu kvm64 \
```

```
-smp 2 /images/sles.raw -writeconfig /images/sles.cfg
(exited)
tux > cat /images/sles.cfg
# qemu config file

[drive]
  index = "0"
  media = "disk"
  file = "/images/sles_base.raw"
```

This way you can effectively manage the configuration of your virtual machines' devices in a well-arranged way.

29.2.3 Guest Real-Time Clock

-rtc *OPTIONS*

Specifies the way the RTC is handled inside a VM Guest. By default, the clock of the guest is derived from that of the host system. Therefore, it is recommended that the host system clock is synchronized with an accurate external clock (for example, via NTP service).

If you need to isolate the VM Guest clock from the host one, specify clock=vm instead of the default clock=host.

You can also specify the initial time of the VM Guest's clock with the base option:

```
tux > qemu-system-x86_64 [...] -rtc clock=vm,base=2010-12-03T01:02:00
```

Instead of a time stamp, you can specify utc or localtime. The former instructs VM Guest to start at the current UTC value (Coordinated Universal Time, see <http://en.wikipedia.org/wiki/UTC>), while the latter applies the local time setting.

29.3 Using Devices in QEMU

QEMU virtual machines emulate all devices needed to run a VM Guest. QEMU supports, for example, several types of network cards, block devices (hard and removable drives), USB devices, character devices (serial and parallel ports), or multimedia devices (graphic and sound cards). This section introduces options to configure various types of supported devices.



Tip

If your device, such as `-drive`, needs a special driver and driver properties to be set, specify them with the `-device` option, and identify with `drive=` suboption. For example:

```
tux > sudo qemu-system-x86_64 [...] -drive if=none,id=drive0,format=raw \  
-device virtio-blk-pci,drive=drive0,scsi=off ...
```

To get help on available drivers and their properties, use `-device ?` and `-device DRIVER,?`.

29.3.1 Block Devices

Block devices are vital for virtual machines. In general, these are fixed or removable storage media usually called *drives*. One of the connected hard disks typically holds the guest operating system to be virtualized.

Virtual Machine drives are defined with `-drive`. This option has many sub-options, some of which are described in this section. For the complete list, see the manual page ([man 1 qemu](#)).

SUB-OPTIONS FOR THE `-drive` OPTION

`file=image_fname`

Specifies the path to the disk image that will be used with this drive. If not specified, an empty (removable) drive is assumed.

`if=drive_interface`

Specifies the type of interface to which the drive is connected. Currently only `floppy`, `scsi`, `ide`, or `virtio` are supported by SUSE. `virtio` defines a paravirtualized disk driver. Default is `ide`.

`index=index_of_connector`

Specifies the index number of a connector on the disk interface (see the `if` option) where the drive is connected. If not specified, the index is automatically incremented.

`media=type`

Specifies the type of media. Can be `disk` for hard disks, or `cdrom` for removable CD-ROM drives.

`format=img_fmt`

Specifies the format of the connected disk image. If not specified, the format is autodetected. Currently, SUSE supports raw and qcow2 formats.

cache=method

Specifies the caching method for the drive. Possible values are unsafe, writethrough, writeback, directsync, or none. To improve performance when using the qcow2 image format, select writeback. none disables the host page cache and, therefore, is the safest option. Default for image files is writeback. For more information, see [Chapter 15, Disk Cache Modes](#).



Tip

To simplify defining block devices, QEMU understands several shortcuts which you may find handy when entering the qemu-system-ARCH command line.

You can use

```
tux > sudo qemu-system-x86_64 -cdrom /images/cdrom.iso
```

instead of

```
tux > sudo qemu-system-x86_64 -drive format=raw,file=/images/cdrom.iso,index=2,media=cdrom
```

and

```
tux > sudo qemu-system-x86_64 -hda /images/image1.raw -hdb /images/image2.raw -hdc \
\
/images/image3.raw -hdd /images/image4.raw
```

instead of

```
tux > sudo qemu-system-x86_64 -drive format=raw,file=/images/image1.raw,index=0,media=disk \
-drive format=raw,file=/images/image2.raw,index=1,media=disk \
-drive format=raw,file=/images/image3.raw,index=2,media=disk \
-drive format=raw,file=/images/image4.raw,index=3,media=disk
```



Tip: Using Host Drives Instead of Images

As an alternative to using disk images (see [Section 28.2, “Managing Disk Images with `qemu-img`”](#)) you can also use existing VM Host Server disks, connect them as drives, and access them from VM Guest. Use the host disk device directly instead of disk image file names.

To access the host CD-ROM drive, use

```
tux > sudo qemu-system-x86_64 [...] -drive file=/dev/cdrom,media=cdrom
```

To access the host hard disk, use

```
tux > sudo qemu-system-x86_64 [...] -drive file=/dev/hdb,media=disk
```

A host drive used by a VM Guest must not be accessed concurrently by the VM Host Server or another VM Guest.

29.3.1.1 Freeing Unused Guest Disk Space

A *Sparse image file* is a type of disk image file that grows in size as the user adds data to it, taking up only as much disk space as is stored in it. For example, if you copy 1 GB of data inside the sparse disk image, its size grows by 1 GB. If you then delete for example 500 MB of the data, the image size does not by default decrease as expected.

That is why the `discard=on` option is introduced on the KVM command line. It tells the hypervisor to automatically free the “holes” after deleting data from the sparse guest image. Note that this option is valid only for the `if=scsi` drive interface:

```
tux > sudo qemu-system-x86_64 [...] -drive format=img_format,file=/path/to/  
file.img,if=scsi,discard=on
```



Important: Support Status

`if=scsi` is not supported. This interface does not map to *virtio-scsi*, but rather to the *lsi SCSI adapter*.

29.3.1.2 IOThreads

IOThreads are dedicated event loop threads for virtio devices to perform I/O requests in order to improve scalability, especially on an SMP VM Host Server with SMP VM Guests using many disk devices. Instead of using QEMU's main event loop for I/O processing, IOThreads allow spreading I/O work across multiple CPUs and can improve latency when properly configured.

IOThreads are enabled by defining IOThread objects. virtio devices can then use the objects for their I/O event loops. Many virtio devices can use a single IOThread object, or virtio devices and IOThread objects can be configured in a 1:1 mapping. The following example creates a single IOThread with ID `iothread0` which is then used as the event loop for two virtio-blk devices.

```
tux > qemu-system-x86_64 [...] -object iothread,id=iothread0\  
-drive if=none,id=drive0,cache=none,aio=native,\  
format=raw,file=filename -device virtio-blk-pci,drive=drive0,scsi=off,\  
iothread=iothread0 -drive if=none,id=drive1,cache=none,aio=native,\  
format=raw,file=filename -device virtio-blk-pci,drive=drive1,scsi=off,\  
iothread=iothread0 [...]
```

The following qemu command line example illustrates a 1:1 virtio device to IOThread mapping:

```
tux > qemu-system-x86_64 [...] -object iothread,id=iothread0\  
-object iothread,id=iothread1 -drive if=none,id=drive0,cache=none,aio=native,\  
format=raw,file=filename -device virtio-blk-pci,drive=drive0,scsi=off,\  
iothread=iothread0 -drive if=none,id=drive1,cache=none,aio=native,\  
format=raw,file=filename -device virtio-blk-pci,drive=drive1,scsi=off,\  
iothread=iothread1 [...]
```

29.3.1.3 Bio-Based I/O Path for virtio-blk

For better performance of I/O-intensive applications, a new I/O path was introduced for the virtio-blk interface in kernel version 3.7. This bio-based block device driver skips the I/O scheduler, and thus shortens the I/O path in guest and has lower latency. It is especially useful for high-speed storage devices, such as SSD disks.

The driver is disabled by default. To use it, do the following:

1. Append `virtio_blk.use_bio=1` to the kernel command line on the guest. You can do so via *YaST* > *System* > *Boot Loader*.

You can do it also by editing `/etc/default/grub`, searching for the line that contains `GRUB_CMDLINE_LINUX_DEFAULT=`, and adding the kernel parameter at the end. Then run `grub2-mkconfig >/boot/grub2/grub.cfg` to update the grub2 boot menu.

2. Reboot the guest with the new kernel command line active.



Tip: Bio-Based Driver on Slow Devices

The bio-based virtio-blk driver does not help on slow devices such as spin hard disks. The reason is that the benefit of scheduling is larger than what the shortened bio path offers. Do not use the bio-based driver on slow devices.

29.3.1.4 Accessing iSCSI Resources Directly

QEMU now integrates with `libiscsi`. This allows QEMU to access iSCSI resources directly and use them as virtual machine block devices. This feature does not require any host iSCSI initiator configuration, as is needed for a libvirt iSCSI target based storage pool setup. Instead it directly connects guest storage interfaces to an iSCSI target LUN by means of the user space library `libiscsi`. iSCSI-based disk devices can also be specified in the libvirt XML configuration.



Note: RAW Image Format

This feature is only available using the RAW image format, as the iSCSI protocol has some technical limitations.

The following is the QEMU command line interface for iSCSI connectivity.



Note: virt-manager Limitation

The use of `libiscsi` based storage provisioning is not yet exposed by the `virt-manager` interface, but instead it would be configured by directly editing the guest xml. This new way of accessing iSCSI based storage is to be done at the command line.

```
tux > sudo qemu-system-x86_64 -machine accel=kvm \  
-drive file=iscsi://192.168.100.1:3260/iqn.2016-08.com.example:314605ab-a88e-49af-  
b4eb-664808a3443b/0,\  
format=raw,if=none,id=mydrive,cache=none \  
-device ide-hd,bus=ide.0,unit=0,drive=mydrive ...
```

Here is an example snippet of guest domain xml which uses the protocol based iSCSI:

```
<devices>
```

```

...
<disk type='network' device='disk'>
  <driver name='qemu' type='raw'/>
  <source protocol='iscsi' name='iqn.2013-07.com.example:iscsi-nopool/2'>
    <host name='example.com' port='3260'/>
  </source>
  <auth username='myuser'>
    <secret type='iscsi' usage='libvirtiscsi'/>
  </auth>
  <target dev='vda' bus='virtio'/>
</disk>
</devices>

```

Contrast that with an example which uses the host based iSCSI initiator which virt-manager sets up:

```

<devices>
...
<disk type='block' device='disk'>
  <driver name='qemu' type='raw' cache='none' io='native'/>
  <source dev='/dev/disk/by-path/scsi-0:0:0:0'/>
  <target dev='hda' bus='ide'/>
  <address type='drive' controller='0' bus='0' target='0' unit='0'/>
</disk>
<controller type='ide' index='0'>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x01'
    function='0x1'/>
</controller>
</devices>

```

29.3.1.5 Using RADOS Block Devices with QEMU

RADOS Block Devices (RBD) store data in a Ceph cluster. They allow snapshotting, replication, and data consistency. You can use an RBD from your KVM-managed VM Guests similarly to how you use other block devices.

29.3.2 Graphic Devices and Display Options

This section describes QEMU options affecting the type of the emulated video card and the way VM Guest graphical output is displayed.

29.3.2.1 Defining Video Cards

QEMU uses `-vga` to define a video card used to display VM Guest graphical output. The `-vga` option understands the following values:

none

Disables video cards on VM Guest (no video card is emulated). You can still access the running VM Guest via the serial console.

std

Emulates a standard VESA 2.0 VBE video card. Use it if you intend to use high display resolution on VM Guest.

cirrus

Emulates Cirrus Logic GD5446 video card. Good choice if you insist on high compatibility of the emulated video hardware. Most operating systems (even Windows 95) recognize this type of card.



Tip

For best video performance with the `cirrus` type, use 16-bit color depth both on VM Guest and VM Host Server.

29.3.2.2 Display Options

The following options affect the way VM Guest graphical output is displayed.

-display gtk

Display video output in a GTK window. This interface provides UI elements to configure and control the VM during runtime.

-display sdl

Display video output via SDL, usually in a separate graphics window. For more information, see the SDL documentation.

-spice option[,option[,...]]

Enables the spice remote desktop protocol.

-display vnc

Refer to [Section 29.5, “Viewing a VM Guest with VNC”](#) for more information.

-nographic

Disables QEMU's graphical output. The emulated serial port is redirected to the console. After starting the virtual machine with -nographic, press `Ctrl-A H` in the virtual console to view the list of other useful shortcuts, for example, to toggle between the console and the QEMU monitor.

```
tux > qemu-system-x86_64 -hda /images/sles_base.raw -nographic

C-a h   print this help
C-a x   exit emulator
C-a s   save disk data back to file (if -snapshot)
C-a t   toggle console timestamps
C-a b   send break (magic sysrq)
C-a c   switch between console and monitor
C-a C-a sends C-a
        (pressed C-a c)

QEMU 2.3.1 monitor - type 'help' for more information
(qemu)
```

-no-frame

Disables decorations for the QEMU window. Convenient for dedicated desktop work space.

-full-screen

Starts QEMU graphical output in full screen mode.

-no-quit

Disables the close button of the QEMU window and prevents it from being closed by force.

-alt-grab, -ctrl-grab

By default, the QEMU window releases the “captured” mouse after pressing `Ctrl-Alt`. You can change the key combination to either `Ctrl-Alt-Shift` (-alt-grab), or the right `Ctrl` key (-ctrl-grab).

29.3.3 USB Devices

There are two ways to create USB devices usable by the VM Guest in KVM: you can either emulate new USB devices inside a VM Guest, or assign an existing host USB device to a VM Guest. To use USB devices in QEMU you first need to enable the generic USB driver with the -usb option. Then you can specify individual devices with the -usbdevice option.

29.3.3.1 Emulating USB Devices in VM Guest

SUSE currently supports the following types of USB devices: disk, host, serial, braille, net, mouse, and tablet.

TYPES OF USB DEVICES FOR THE `-usbdevice` OPTION

disk

Emulates a mass storage device based on file. The optional format option is used rather than detecting the format.

```
tux > qemu-system-x86_64 [...] -usbdevice
      disk:format=raw:/virt/usb_disk.raw
```

host

Pass through the host device (identified by bus.addr).

serial

Serial converter to a host character device.

braille

Emulates a braille device using BrlAPI to display the braille output.

net

Emulates a network adapter that supports CDC Ethernet and RNDIS protocols.

mouse

Emulates a virtual USB mouse. This option overrides the default PS/2 mouse emulation. The following example shows the hardware status of a mouse on VM Guest started with qemu-system-ARCH [...] -usbdevice mouse:

```
tux > sudo hwinfg --mouse
20: USB 00.0: 10503 USB Mouse
[Created at usb.122]
UDI: /org/freedesktop/Hal/devices/usb_device_627_1_1_if0
[...]
Hardware Class: mouse
Model: "Adomax QEMU USB Mouse"
Hotplug: USB
Vendor: usb 0x0627 "Adomax Technology Co., Ltd"
Device: usb 0x0001 "QEMU USB Mouse"
[...]
```

tablet

Emulates a pointer device that uses absolute coordinates (such as touchscreen). This option overrides the default PS/2 mouse emulation. The tablet device is useful if you are viewing VM Guest via the VNC protocol. See [Section 29.5, “Viewing a VM Guest with VNC”](#) for more information.

29.3.4 Character Devices

Use `-chardev` to create a new character device. The option uses the following general syntax:

```
qemu-system-x86_64 [...] -chardev BACKEND_TYPE,id=ID_STRING
```

where `BACKEND_TYPE` can be one of `null`, `socket`, `udp`, `msmouse`, `vc`, `file`, `pipe`, `console`, `serial`, `pty`, `stdio`, `braille`, `tty`, or `parport`. All character devices must have a unique identification string up to 127 characters long. It is used to identify the device in other related directives. For the complete description of all back-end's sub-options, see the manual page ([man 1 qemu](#)). A brief description of the available `back-ends` follows:

`null`

Creates an empty device that outputs no data and drops any data it receives.

`stdio`

Connects to QEMU's process standard input and standard output.

`socket`

Creates a two-way stream socket. If `PATH` is specified, a Unix socket is created:

```
tux > sudo qemu-system-x86_64 [...] -chardev \  
socket,id=unix_socket1,path=/tmp/unix_socket1,server
```

The `SERVER` suboption specifies that the socket is a listening socket.

If `PORT` is specified, a TCP socket is created:

```
tux > sudo qemu-system-x86_64 [...] -chardev \  
socket,id=tcp_socket1,host=localhost,port=7777,server,nowait
```

The command creates a local listening (`server`) TCP socket on port 7777. QEMU will not block waiting for a client to connect to the listening port (`nowait`).

`udp`

Sends all network traffic from VM Guest to a remote host over the UDP protocol.

```
tux > sudo qemu-system-x86_64 [...] \  
udp,host=192.168.1.100
```

```
-chardev udp,id=udp_fwd,host=mercury.example.com,port=7777
```

The command binds port 7777 on the remote host `mercury.example.com` and sends VM Guest network traffic there.

vc

Creates a new QEMU text console. You can optionally specify the dimensions of the virtual console:

```
tux > sudo qemu-system-x86_64 [...] -chardev vc,id=vc1,width=640,height=480 \  
-mon chardev=vc1
```

The command creates a new virtual console called `vc1` of the specified size, and connects the QEMU monitor to it.

file

Logs all traffic from VM Guest to a file on VM Host Server. The `path` is required and will be created if it does not exist.

```
tux > sudo qemu-system-x86_64 [...] \  
-chardev file,id=qemu_log1,path=/var/log/qemu/guest1.log
```

By default QEMU creates a set of character devices for serial and parallel ports, and a special console for QEMU monitor. However, you can create your own character devices and use them for the mentioned purposes. The following options will help you:

-serial CHAR_DEV

Redirects the VM Guest's virtual serial port to a character device `CHAR_DEV` on VM Host Server. By default, it is a virtual console (`vc`) in graphical mode, and `stdio` in non-graphical mode. The `-serial` understands many sub-options. See the manual page [man 1 qemu](#) for a complete list of them.

You can emulate up to four serial ports. Use `-serial none` to disable all serial ports.

-parallel DEVICE

Redirects the VM Guest's parallel port to a `DEVICE`. This option supports the same devices as `-serial`.



Tip

With openSUSE Leap as a VM Host Server, you can directly use the hardware parallel port devices `/dev/parportN` where `N` is the number of the port.

You can emulate up to three parallel ports. Use `-parallel none` to disable all parallel ports.

`-monitor CHAR_DEV`

Redirects the QEMU monitor to a character device `CHAR_DEV` on VM Host Server. This option supports the same devices as `-serial`. By default, it is a virtual console (`vc`) in a graphical mode, and `stdio` in non-graphical mode.

For a complete list of available character devices back-ends, see the man page (`man 1 qemu`).

29.4 Networking in QEMU

Use the `-netdev` option in combination with `-device` to define a specific type of networking and a network interface card for your VM Guest. The syntax for the `-netdev` option is

```
-netdev type[,prop[=value][,...]]
```

Currently, SUSE supports the following network types: `user`, `bridge`, and `tap`. For a complete list of `-netdev` sub-options, see the manual page (`man 1 qemu`).

SUPPORTED `-netdev` SUB-OPTIONS

`bridge`

Uses a specified network helper to configure the TAP interface and attach it to a specified bridge. For more information, see [Section 29.4.3, “Bridged Networking”](#).

`user`

Specifies user-mode networking. For more information, see [Section 29.4.2, “User-Mode Networking”](#).

`tap`

Specifies bridged or routed networking. For more information, see [Section 29.4.3, “Bridged Networking”](#).

29.4.1 Defining a Network Interface Card

Use `-netdev` together with the related `-device` option to add a new emulated network card:

```
tux > sudo qemu-system-x86_64 [...] \
```



```
-netdev tap①,id=hostnet0 \  
-device virtio-net-pci②,netdev=hostnet0,vlan=1③,\  
macaddr=00:16:35:AF:94:4B④,name=ncard1
```

- ① Specifies the network device type.
- ② Specifies the model of the network card. Use `qemu-system-ARCH -device help` and search for the `Network devices:` section to get the list of all network card models supported by QEMU on your platform.
Currently, SUSE supports the models `rtl8139`, `e1000` and its variants `e1000-82540em`, `e1000-82544gc` and `e1000-82545em`, and `virtio-net-pci`. To view a list of options for a specific driver, add `help` as a driver option:

```
tux > sudo qemu-system-x86_64 -device e1000,help  
e1000.mac=macaddr  
e1000.vlan=vlan  
e1000.netdev=netdev  
e1000.bootindex=int32  
e1000.autonegotiation=on/off  
e1000.mitigation=on/off  
e1000.addr=pci-devfn  
e1000.romfile=str  
e1000.rombar=uint32  
e1000.multifunction=on/off  
e1000.command_serr_enable=on/off
```

- ③ Connects the network interface to VLAN number 1. You can specify your own number—it is mainly useful for identification purpose. If you omit this suboption, QEMU uses the default 0.
- ④ Specifies the Media Access Control (MAC) address for the network card. It is a unique identifier and you are advised to always specify it. If not, QEMU supplies its own default MAC address and creates a possible MAC address conflict within the related VLAN.

29.4.2 User-Mode Networking

The `-netdev user` option instructs QEMU to use user-mode networking. This is the default if no networking mode is selected. Therefore, these command lines are equivalent:

```
tux > sudo qemu-system-x86_64 -hda /images/sles_base.raw
```

```
tux > sudo qemu-system-x86_64 -hda /images/sles_base.raw -netdev user,id=hostnet0
```

This mode is useful if you want to allow the VM Guest to access the external network resources, such as the Internet. By default, no incoming traffic is permitted and therefore, the VM Guest is not visible to other machines on the network. No administrator privileges are required in this networking mode. The user-mode is also useful for doing a network boot on your VM Guest from a local directory on VM Host Server.

The VM Guest allocates an IP address from a virtual DHCP server. VM Host Server (the DHCP server) is reachable at 10.0.2.2, while the IP address range for allocation starts from 10.0.2.15. You can use `ssh` to connect to VM Host Server at 10.0.2.2, and `scp` to copy files back and forth.

29.4.2.1 Command Line Examples

This section shows several examples on how to set up user-mode networking with QEMU.

EXAMPLE 29.1: RESTRICTED USER-MODE NETWORKING

```
tux > sudo qemu-system-x86_64 [...] \  
-netdev user ❶,id=hostnet0 \  
-device virtio-net-pci,netdev=hostnet0,vlan=1 ❷,name=user_net1 ❸,restrict=yes ❹
```

- ❶ Specifies user-mode networking.
- ❷ Connects to VLAN number 1. If omitted, defaults to 0.
- ❸ Specifies a human-readable name of the network stack. Useful when identifying it in the QEMU monitor.
- ❹ Isolates VM Guest. It then cannot communicate with VM Host Server and no network packets will be routed to the external network.

EXAMPLE 29.2: USER-MODE NETWORKING WITH CUSTOM IP RANGE

```
tux > sudo qemu-system-x86_64 [...] \  
-netdev user,id=hostnet0 \  
-device virtio-net-pci,netdev=hostnet0,net=10.2.0.0/8 ❶,host=10.2.0.6 ❷,\  
dhcpstart=10.2.0.20 ❸,hostname=tux_kvm_guest ❹
```

- ❶ Specifies the IP address of the network that VM Guest sees and optionally the netmask. Default is 10.0.2.0/8.
- ❷ Specifies the VM Host Server IP address that VM Guest sees. Default is 10.0.2.2.
- ❸ Specifies the first of the 16 IP addresses that the built-in DHCP server can assign to VM Guest. Default is 10.0.2.15.

- 4 Specifies the host name that the built-in DHCP server will assign to VM Guest.

EXAMPLE 29.3: USER-MODE NETWORKING WITH NETWORK-BOOT AND TFTP

```
tux > sudo qemu-system-x86_64 [...] \  
-netdev user,id=hostnet0 \  
-device virtio-net-pci,netdev=hostnet0,tftp=/images/tftp_dir ❶,\  
bootfile=/images/boot/pxelinux.0 ❷
```

- ❶ Activates a built-in TFTP (a file transfer protocol with the functionality of a very basic FTP) server. The files in the specified directory will be visible to a VM Guest as the root of a TFTP server.
- ❷ Broadcasts the specified file as a BOOTP (a network protocol that offers an IP address and a network location of a boot image, often used in diskless workstations) file. When used together with `tftp`, the VM Guest can boot from network from the local directory on the host.

EXAMPLE 29.4: USER-MODE NETWORKING WITH HOST PORT FORWARDING

```
tux > sudo qemu-system-x86_64 [...] \  
-netdev user,id=hostnet0 \  
-device virtio-net-pci,netdev=hostnet0,hostfwd=tcp::2222-:22
```

Forwards incoming TCP connections to the port 2222 on the host to the port 22 (SSH) on VM Guest. If `sshd` is running on VM Guest, enter

```
tux > ssh qemu_host -p 2222
```

where `qemu_host` is the host name or IP address of the host system, to get a SSH prompt from VM Guest.

29.4.3 Bridged Networking

With the `-netdev tap` option, QEMU creates a network bridge by connecting the host TAP network device to a specified VLAN of VM Guest. Its network interface is then visible to the rest of the network. This method does not work by default and needs to be explicitly specified.

First, create a network bridge and add a VM Host Server physical network interface (usually `eth0`) to it:

1. Start *YaST Control Center* and select *System > Network Settings*.

2. Click *Add* and select *Bridge* from the *Device Type* drop-down box in the *Hardware Dialog* window. Click *Next*.
3. Choose whether you need a dynamically or statically assigned IP address, and fill the related network settings if applicable.
4. In the *Bridged Devices* pane, select the Ethernet device to add to the bridge. Click *Next*. When asked about adapting an already configured device, click *Continue*.
5. Click *OK* to apply the changes. Check if the bridge is created:

```
tux > bridge link
2: eth0 state UP : <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 master br0 \
state forwarding priority 32 cost 100
```

29.4.3.1 Connecting to a Bridge Manually

Use the following example script to connect VM Guest to the newly created bridge interface `br0`. Several commands in the script are run via the `sudo` mechanism because they require `root` privileges.



Tip: Required Software

To manage a network bridge, you need to have the `tunctl` package installed.

```
#!/bin/bash
bridge=br0 ①
tap=$(sudo tunctl -u $(whoami) -b) ②
sudo ip link set $tap up ③
sleep 1s ④
sudo ip link add name $bridge type bridge
sudo ip link set $bridge up
sudo ip link set $tap master $bridge ⑤
qemu-system-x86_64 -machine accel=kvm -m 512 -hda /images/sles_base.raw \
-netdev tap,id=hostnet0 \
-device virtio-net-pci,netdev=hostnet0,vlan=0,macaddr=00:16:35:AF:94:4B,\
ifname=$tap ⑥,script=no ⑦,downscript=no
sudo ip link set $tap nomaster ⑧
sudo ip link set $tap down ⑨
sudo tunctl -d $tap ⑩
```

- ① Name of the bridge device.

- ② Prepare a new TAP device and assign it to the user who runs the script. TAP devices are virtual network devices often used for virtualization and emulation setups.
- ③ Bring up the newly created TAP network interface.
- ④ Make a 1-second pause to make sure the new TAP network interface is really up.
- ⑤ Add the new TAP device to the network bridge br0.
- ⑥ The ifname= suboption specifies the name of the TAP network interface used for bridging.
- ⑦ Before **qemu-system-ARCH** connects to a network bridge, it checks the script and down-script values. If it finds the specified scripts on the VM Host Server file system, it runs the script before it connects to the network bridge and downscript after it exits the network environment. You can use these scripts to set up and tear down the bridged interfaces. By default, /etc/qemu-ifup and /etc/qemu-ifdown are examined. If script=no and downscript=no are specified, the script execution is disabled and you need to take care of it manually.
- ⑧ Deletes the TAP interface from a network bridge br0.
- ⑨ Sets the state of the TAP device to down.
- ⑩ Tear down the TAP device.

29.4.3.2 Connecting to a Bridge with `qemu-bridge-helper`

Another way to connect VM Guest to a network through a network bridge is by means of the `qemu-bridge-helper` helper program. It configures the TAP interface for you, and attaches it to the specified bridge. The default helper executable is `/usr/lib/qemu-bridge-helper`. The helper executable is setuid root, which is only executable by the members of the virtualization group (`kvm`). Therefore the **qemu-system-ARCH** command itself does not need to be run under root privileges.

The helper is automatically called when you specify a network bridge:

```
qemu-system-x86_64 [...] \
-netdev bridge,id=hostnet0,vlan=0,br=br0 \
-device virtio-net-pci,netdev=hostnet0
```

You can specify your own custom helper script that will take care of the TAP device (de)configuration, with the helper=/path/to/your/helper option:

```
qemu-system-x86_64 [...] \
-netdev bridge,id=hostnet0,vlan=0,br=br0,helper=/path/to/bridge-helper \
```

```
-device virtio-net-pci,netdev=hostnet0
```



Tip

To define access privileges to `qemu-bridge-helper`, inspect the `/etc/qemu/bridge.conf` file. For example the following directive

```
allow br0
```

allows the `qemu-system-ARCH` command to connect its VM Guest to the network bridge `br0`.

29.5 Viewing a VM Guest with VNC

By default QEMU uses a GTK (a cross-platform toolkit library) window to display the graphical output of a VM Guest. With the `-vnc` option specified, you can make QEMU listen on a specified VNC display and redirect its graphical output to the VNC session.



Tip

When working with QEMU's virtual machine via VNC session, it is useful to work with the `-usbdevice tablet` option.

Moreover, if you need to use another keyboard layout than the default `en-us`, specify it with the `-k` option.

The first suboption of `-vnc` must be a *display* value. The `-vnc` option understands the following display specifications:

host:display

Only connections from host on the display number display will be accepted. The TCP port on which the VNC session is then running is normally a 5900 + display number. If you do not specify host, connections will be accepted from any host.

unix:path

The VNC server listens for connections on Unix domain sockets. The path option specifies the location of the related Unix socket.

none

The VNC server functionality is initialized, but the server itself is not started. You can start the VNC server later with the QEMU monitor. For more information, see [Chapter 30, Virtual Machine Administration Using QEMU Monitor](#).

Following the display value there may be one or more option flags separated by commas. Valid options are:

reverse

Connect to a listening VNC client via a *reverse* connection.

websocket

Opens an additional TCP listening port dedicated to VNC Websocket connections. By definition the Websocket port is 5700 + display.

password

Require that password-based authentication is used for client connections.

tls

Require that clients use TLS when communicating with the VNC server.

x509=/path/to/certificate/dir

Valid if TLS is specified. Require that x509 credentials are used for negotiating the TLS session.

x509verify=/path/to/certificate/dir

Valid if TLS is specified. Require that x509 credentials are used for negotiating the TLS session.

sasl

Require that the client uses SASL to authenticate with the VNC server.

acl

Turn on access control lists for checking of the x509 client certificate and SASL party.

lossy

Enable lossy compression methods (gradient, JPEG, ...).

non-adaptive

Disable adaptive encodings. Adaptive encodings are enabled by default.

share=[allow-exclusive|force-shared|ignore]

Set display sharing policy.



Note

For more details about the display options, see the *qemu-doc* man page.

An example VNC usage:

```
tux > qemu-system-x86_64 [...] -vnc :5
# (on the client:)
wilber > vncviewer venus:5 &
```

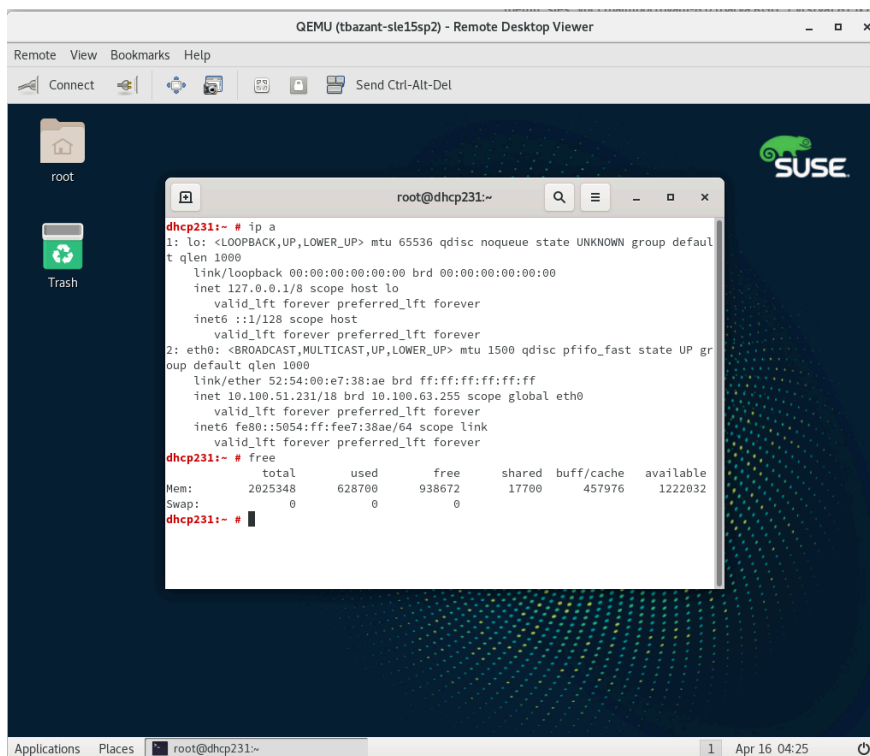


FIGURE 29.2: QEMU VNC SESSION

29.5.1 Secure VNC Connections

The default VNC server setup does not use any form of authentication. In the previous example, any user can connect and view the QEMU VNC session from any host on the network.

There are several levels of security that you can apply to your VNC client/server connection. You can either protect your connection with a password, use x509 certificates, use SASL authentication, or even combine some authentication methods in one QEMU command.

For more information about configuring x509 certificates on a VM Host Server and the client, see [Section 10.3.2, “Remote TLS/SSL Connection with x509 Certificate \(qemu+tls or xen+tls\)”](#) and [Section 10.3.2.3, “Configuring the Client and Testing the Setup”](#).

The Remmina VNC viewer supports advanced authentication mechanisms. Therefore, it will be used to view the graphical output of VM Guest in the following examples. For this example, let us assume that the server x509 certificates `ca-cert.pem`, `server-cert.pem`, and `server-key.pem` are located in the `/etc/pki/qemu` directory on the host. The client certificates can be placed in any custom directory, as Remmina asks for their path on the connection start-up.

EXAMPLE 29.5: PASSWORD AUTHENTICATION

```
qemu-system-x86_64 [...] -vnc :5,password -monitor stdio
```

Starts the VM Guest graphical output on VNC display number 5 (usually port 5905). The `password` suboption initializes a simple password-based authentication method. There is no password set by default and you need to set one with the `change vnc password` command in QEMU monitor:

```
QEMU 2.3.1 monitor - type 'help' for more information
(qemu) change vnc password
Password: ****
```

You need the `-monitor stdio` option here, because you would not be able to manage the QEMU monitor without redirecting its input/output.

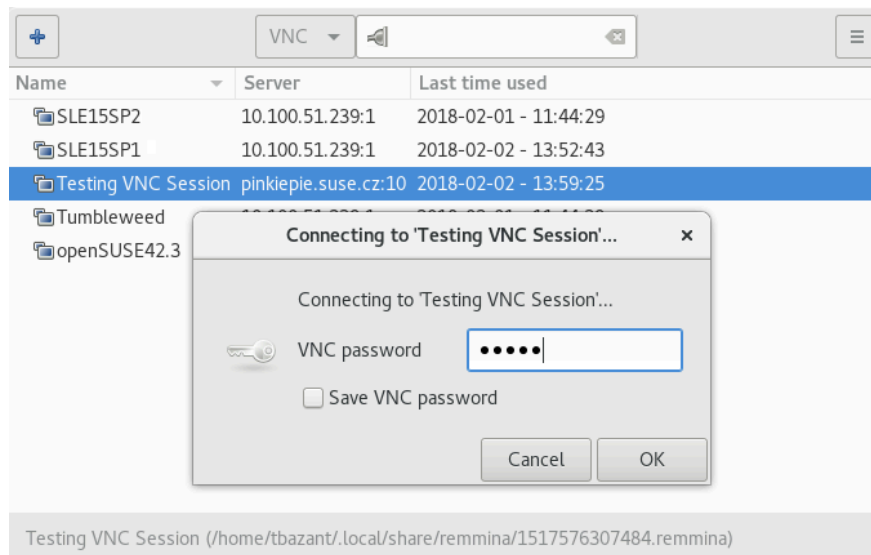


FIGURE 29.3: AUTHENTICATION DIALOG IN REMMINA

EXAMPLE 29.6: X509 CERTIFICATE AUTHENTICATION

The QEMU VNC server can use TLS encryption for the session and x509 certificates for authentication. The server asks the client for a certificate and validates it against the CA certificate. Use this authentication type if your company provides an internal certificate authority.

```
qemu-system-x86_64 [...] -vnc :5,tls,x509verify=/etc/pki/qemu
```

EXAMPLE 29.7: X509 CERTIFICATE AND PASSWORD AUTHENTICATION

You can combine the password authentication with TLS encryption and x509 certificate authentication to create a two-layer authentication model for clients. Remember to set the password in the QEMU monitor after you run the following command:

```
qemu-system-x86_64 [...] -vnc :5,password,tls,x509verify=/etc/pki/qemu \  
-monitor stdio
```

EXAMPLE 29.8: SASL AUTHENTICATION

Simple Authentication and Security Layer (SASL) is a framework for authentication and data security in Internet protocols. It integrates several authentication mechanisms, like PAM, Kerberos, LDAP and more. SASL keeps its own user database, so the connecting user accounts do not need to exist on VM Host Server.

For security reasons, you are advised to combine SASL authentication with TLS encryption and x509 certificates:

```
qemu-system-x86_64 [...] -vnc :5,tls,x509,sasl -monitor stdio
```

30 Virtual Machine Administration Using QEMU Monitor

When QEMU is running, a monitor console is provided for performing interaction with the user. Using the commands available in the monitor console, it is possible to inspect the running operating system, change removable media, take screenshots or audio grabs and control other aspects of the virtual machine.



Note

The following sections list selected useful QEMU monitor commands and their purpose. To get the full list, enter `help` in the QEMU monitor command line.

30.1 Accessing Monitor Console



Tip: No Monitor Console for libvirt

You can access the monitor console only if you started the virtual machine directly with the `qemu-system-ARCH` command and are viewing its graphical output in a native QEMU window.

If you started the virtual machine with `libvirt` (for example using `virt-manager`) and are viewing its output via VNC or Spice sessions, you cannot access the monitor console directly. You can, however, send the monitor command to the virtual machine via `virsh`:

```
root # virsh qemu-monitor-command COMMAND
```

The way you access the monitor console depends on which display device you use to view the output of a virtual machine. Find more details about displays in [Section 29.3.2.2, "Display Options"](#). For example, to view the monitor while the `-display gtk` option is in use, press `Ctrl-Alt-2`. Similarly, when the `-nographic` option is in use, you can switch to the monitor console by pressing `Ctrl-a c`.

To get help while using the console, use `help` or `?`. To get help for a specific command, use `help COMMAND`.

30.2 Getting Information about the Guest System

To get information about the guest system, use **info**. If used without any option, the list of possible options is printed. Options determine which part of the system will be analyzed:

info version

Shows the version of QEMU.

info commands

Lists available QMP commands.

info network

Shows the network state.

info chardev

Shows the character devices.

info block

Information about block devices, such as hard disks, floppy drives, or CD-ROMs.

info blockstats

Read and write statistics on block devices.

info registers

Shows the CPU registers.

info cpus

Shows information about available CPUs.

info history

Shows the command line history.

info irq

Shows the interrupt statistics.

info pic

Shows the i8259 (PIC) state.

info pci

Shows the PCI information.

info tlb

Shows virtual to physical memory mappings.

info mem

Shows the active virtual memory mappings.

info jit

Shows dynamic compiler information.

info kvm

Shows the KVM information.

info numa

Shows the NUMA information.

info usb

Shows the guest USB devices.

info usbhost

Shows the host USB devices.

info profile

Shows the profiling information.

info capture

Shows the capture (audio grab) information.

info snapshots

Shows the currently saved virtual machine snapshots.

info status

Shows the current virtual machine status.

info mice

Shows which guest mice are receiving events.

info vnc

Shows the VNC server status.

info name

Shows the current virtual machine name.

info uuid

Shows the current virtual machine UUID.

info usernet

Shows the user network stack connection states.

info migrate

Shows the migration status.

info balloon

Shows the balloon device information.

info qtree

Shows the device tree.

info qdm

Shows the qdev device model list.

info roms

Shows the ROMs.

info migrate_cache_size

Shows the current migration xbzrle (“Xor Based Zero Run Length Encoding”) cache size.

info migrate_capabilities

Shows the status of the various migration capabilities, such as xbzrle compression.

info mtree

Shows the VM Guest memory hierarchy.

info trace-events

Shows available trace-events and their status.

30.3 Changing VNC Password

To change the VNC password, use the **change vnc password** command and enter the new password:

```
(qemu) change vnc password
Password: *****
(qemu)
```

30.4 Managing Devices

To add a new disk while the guest is running (hotplug), use the `drive_add` and `device_add` commands. First define a new drive to be added as a device to bus 0:

```
(qemu) drive_add 0 if=none,file=/tmp/test.img,format=raw,id=disk1
OK
```

You can confirm your new device by querying the block subsystem:

```
(qemu) info block
[...]
disk1: removable=1 locked=0 tray-open=0 file=/tmp/test.img ro=0 drv=raw \
encrypted=0 bps=0 bps_rd=0 bps_wr=0 iops=0 iops_rd=0 iops_wr=0
```

After the new drive is defined, it needs to be connected to a device so that the guest can see it. The typical device would be a `virtio-blk-pci` or `scsi-disk`. To get the full list of available driver values, run:

```
(qemu) device_add ?
name "VGA", bus PCI
name "usb-storage", bus usb-bus
[...]
name "virtio-blk-pci", bus virtio-bus
```

Now add the device

```
(qemu) device_add virtio-blk-pci,drive=disk1,id=myvirtio1
```

and confirm with

```
(qemu) info pci
[...]
Bus 0, device 4, function 0:
  SCSI controller: PCI device 1af4:1001
  IRQ 0.
  BAR0: I/O at 0xffffffffffffffff [0x003e].
  BAR1: 32 bit memory at 0xffffffffffffffff [0x0000ffe].
  id "myvirtio1"
```



Tip

Devices added with the `device_add` command can be removed from the guest with `device_del`. Enter `help device_del` on the QEMU monitor command line for more information.

To release the device or file connected to the removable media device, use the **eject** *DEVICE* command. Use the optional **-f** to force ejection.

To change removable media (like CD-ROMs), use the **change** *DEVICE* command. The name of the removable media can be determined using the **info block** command:

```
(qemu) info block
ide1-cd0: type=cdrom removable=1 locked=0 file=/dev/sr0 ro=1 drv=host_device
(qemu) change ide1-cd0 /path/to/image
```

30.5 Controlling Keyboard and Mouse

It is possible to use the monitor console to emulate keyboard and mouse input if necessary. For example, if your graphical user interface intercepts some key combinations at low level (such as **Ctrl-Alt-F1** in X Window), you can still enter them using the **sendkey** *KEYS*:

```
sendkey ctrl-alt-f1
```

To list the key names used in the *KEYS* option, enter **sendkey** and press **-|**.

To control the mouse, the following commands can be used:

mouse_move *DX dy [DZ]*

Move the active mouse pointer to the specified coordinates dx, dy with the optional scroll axis dz.

mouse_button *VAL*

Change the state of the mouse buttons (1 = left, 2 = middle, 4 = right).

mouse_set *INDEX*

Set which mouse device receives events. Device index numbers can be obtained with the **info mice** command.

30.6 Changing Available Memory

If the virtual machine was started with the **-balloon virtio** option (the paravirtualized balloon device is therefore enabled), you can change the available memory dynamically. For more information about enabling the balloon device, see [Section 28.1, “Basic Installation with qemu-system-ARCH”](#).

To get information about the balloon device in the monitor console and to determine whether the device is enabled, use the **info balloon** command:

```
(qemu) info balloon
```

If the balloon device is enabled, use the **balloon MEMORY_IN_MB** command to set the requested amount of memory:

```
(qemu) balloon 400
```

30.7 Dumping Virtual Machine Memory

To save the content of the virtual machine memory to a disk or console output, use the following commands:

memsave *ADDR SIZE FILENAME*

Saves virtual memory dump starting at *ADDR* of size *SIZE* to file *FILENAME*

pmemsave *ADDR SIZE FILENAME*

Saves physical memory dump starting at *ADDR* of size *SIZE* to file *FILENAME* -

x / *FMT ADDR*

Makes a virtual memory dump starting at address *ADDR* and formatted according to the *FMT* string. The *FMT* string consists of three parameters *COUNTFORMATSIZE*:

The *COUNT* parameter is the number of items to be dumped.

The *FORMAT* can be *x* (hex), *d* (signed decimal), *u* (unsigned decimal), *o* (octal), *c* (char) or *i* (assembly instruction).

The *SIZE* parameter can be *b* (8 bits), *h* (16 bits), *w* (32 bits) or *g* (64 bits). On x86, *h* or *w* can be specified with the *i* format to respectively select 16 or 32-bit code instruction size.

xp / *FMT ADDR*

Makes a physical memory dump starting at address *ADDR* and formatted according to the *FMT* string. The *FMT* string consists of three parameters *COUNTFORMATSIZE*:

The *COUNT* parameter is the number of the items to be dumped.

The *FORMAT* can be *x* (hex), *d* (signed decimal), *u* (unsigned decimal), *o* (octal), *c* (char) or *i* (asm instruction).

The SIZE parameter can be b (8 bits), h (16 bits), w (32 bits) or g (64 bits). On x86, h or w can be specified with the i format to respectively select 16 or 32-bit code instruction size.

30.8 Managing Virtual Machine Snapshots

Managing snapshots in QEMU monitor is not officially supported by SUSE yet. The information found in this section may be helpful in specific cases.

Virtual Machine snapshots are snapshots of the complete virtual machine including the state of CPU, RAM, and the content of all writable disks. To use virtual machine snapshots, you must have at least one non-removable and writable block device using the qcow2 disk image format. Snapshots are helpful when you need to save your virtual machine in a particular state. For example, after you have configured network services on a virtualized server and want to quickly start the virtual machine in the same state that was saved last. You can also create a snapshot after the virtual machine has been powered off to create a backup state before you try something experimental and possibly make VM Guest unstable. This section introduces the former case, while the latter is described in *Section 28.2.3, "Managing Snapshots of Virtual Machines with qemu-img"*.

The following commands are available for managing snapshots in QEMU monitor:

savevm NAME

Creates a new virtual machine snapshot under the tag NAME or replaces an existing snapshot.

loadvm NAME

Loads a virtual machine snapshot tagged NAME.

delvm

Deletes a virtual machine snapshot.

info snapshots

Prints information about available snapshots.

```
(qemu) info snapshots
Snapshot list:
ID ①      TAG ②      VM SIZE ③    DATE ④      VM CLOCK ⑤
1        booting    4.4M 2013-11-22 10:51:10    00:00:20.476
2        booted     184M 2013-11-22 10:53:03    00:02:05.394
3        logged_in  273M 2013-11-22 11:00:25    00:04:34.843
```

- ① Unique identification number of the snapshot. Usually auto-incremented.
- ② Unique description string of the snapshot. It is meant as a human readable version of the ID.
- ③ The disk space occupied by the snapshot. Note that the more memory is consumed by running applications, the bigger the snapshot is.
- ④ Time and date the snapshot was created.
- ⑤ The current state of the virtual machine's clock.

30.9 Suspending and Resuming Virtual Machine Execution

The following commands are available for suspending and resuming virtual machines:

stop

Suspends the execution of the virtual machine.

cont

Resumes the execution of the virtual machine.

system_reset

Resets the virtual machine. The effect is similar to the reset button on a physical machine. This may leave the file system in an unclean state.

system_powerdown

Sends an *ACPI* shutdown request to the machine. The effect is similar to the power button on a physical machine.

q or quit

Terminates QEMU immediately.

30.10 Live Migration

The live migration process allows to transmit any virtual machine from one host system to another host system without any interruption in availability. It is possible to change hosts permanently or only during maintenance.

The requirements for live migration:

- All requirements from *Section 9.7.1, "Migration Requirements"* are applicable.
- Live migration is only possible between VM Host Servers with the same CPU features.
- *AHCI* interface, *VirtFS* feature, and the `-mem-path` command line option are not compatible with migration.
- The guest on the source and destination hosts must be started in the same way.
- `-snapshot` qemu command line option should not be used for migration (and this `qemu` command line option is not supported).

Important: Support Status

The `postcopy` mode is not yet supported in openSUSE Leap. It is released as a technology preview only. For more information about `postcopy`, see <http://wiki.qemu.org/Features/PostCopyLiveMigration>.

More recommendations can be found at the following Web site: <http://www.linux-kvm.org/page/Migration>

The live migration process has the following steps:

1. The virtual machine instance is running on the source host.
2. The virtual machine is started on the destination host in the frozen listening mode. The parameters used are the same as on the source host plus the `-incoming tcp:IP:PORT` parameter, where `IP` specifies the IP address and `PORT` specifies the port for listening to the incoming migration. If 0 is set as IP address, the virtual machine listens on all interfaces.
3. On the source host, switch to the monitor console and use the `migrate -d tcp: DESTINATION_IP:PORT` command to initiate the migration.
4. To determine the state of the migration, use the `info migrate` command in the monitor console on the source host.
5. To cancel the migration, use the `migrate_cancel` command in the monitor console on the source host.

6. To set the maximum tolerable downtime for migration in seconds, use the `migrate_set_downtime NUMBER_OF_SECONDS` command.
7. To set the maximum speed for migration in bytes per second, use the `migrate_set_speed BYTES_PER_SECOND` command.

30.11 QMP - QEMU Machine Protocol

QMP is a JSON-based protocol that allows applications—such as `libvirt`—to communicate with a running QEMU instance. There are several ways you can access the QEMU monitor using QMP commands.

30.11.1 Access QMP via Standard Input/Output

The most flexible way to use QMP is by specifying the `-mon` option. The following example creates a QMP instance using standard input/output. Note that in the following examples, `->` marks lines with commands sent from client to the running QEMU instance, while `<-` marks lines with the output returned from QEMU.

```
tux > sudo qemu-system-x86_64 [...] \  
-chardev stdio,id=mon0 \  
-mon chardev=mon0,mode=control,pretty=on  
  
<- {  
  "QMP": {  
    "version": {  
      "qemu": {  
        "micro": 0,  
        "minor": 0,  
        "major": 2  
      },  
      "package": ""  
    },  
    "capabilities": [  
    ]  
  }  
}
```

When a new QMP connection is established, QMP sends its greeting message and enters capabilities negotiation mode. In this mode, only the `qmp_capabilities` command works. To exit capabilities negotiation mode and enter command mode, the `qmp_capabilities` command must be issued first:

```
-> { "execute": "qmp_capabilities" }
<- {
  "return": {
  }
}
```

Note that `"return": {}` is a QMP's success response.

QMP's commands can have arguments. For example to eject a CD-ROM drive, enter the following:

```
->{ "execute": "eject", "arguments": { "device": "ide1-cd0" } }
<- {
  "timestamp": {
    "seconds": 1410353381,
    "microseconds": 763480
  },
  "event": "DEVICE_TRAY_MOVED",
  "data": {
    "device": "ide1-cd0",
    "tray-open": true
  }
}
{
  "return": {
  }
}
```

30.11.2 Access QMP via Telnet

Instead of the standard input/output, you can connect the QMP interface to a network socket and communicate with it via a specified port:

```
tux > sudo qemu-system-x86_64 [...] \
-chardev socket,id=mon0,host=localhost,port=4444,server,nowait \
-mon chardev=mon0,mode=control,pretty=on
```

And then run telnet to connect to port 4444:

```
tux > telnet localhost 4444
```

```

Trying ::1...
Connected to localhost.
Escape character is '^]'.
<- {
  "QMP": {
    "version": {
      "qemu": {
        "micro": 0,
        "minor": 0,
        "major": 2
      },
      "package": ""
    },
    "capabilities": [
    ]
  }
}

```

You can create several monitor interfaces at the same time. The following example creates one HMP instance—human monitor which understands 'normal' QEMU monitor's commands—on the standard input/output, and one QMP instance on local host port 4444:

```

tux > sudo qemu-system-x86_64 [...] \
-chardev stdio,id=mon0 -mon chardev=mon0,mode=readline \
-chardev socket,id=mon1,host=localhost,port=4444,server,nowait \
-mon chardev=mon1,mode=control,pretty=on

```

30.11.3 Access QMP via Unix Socket

Invoke QEMU using the `-qmp` option, and create a unix socket:

```

tux > sudo qemu-system-x86_64 [...] \
-qmp unix:/tmp/qmp-sock,server --monitor stdio

QEMU waiting for connection on: unix:./qmp-sock,server

```

To communicate with the QEMU instance via the `/tmp/qmp-sock` socket, use `nc` (see [man 1 nc](#) for more information) from another terminal on the same host:

```

tux > sudo nc -U /tmp/qmp-sock
<- {"QMP": {"version": {"qemu": {"micro": 0, "minor": 0, "major": 2} [...]

```

30.11.4 Access QMP via libvirt's **virsh** Command

If you run your virtual machines under **libvirt** (see *Part II, "Managing Virtual Machines with libvirt"*), you can communicate with its running guests by running the **virsh qemu-monitor-command**:

```
tux > sudo virsh qemu-monitor-command vm_guest1 \  
--pretty '{"execute":"query-kvm"}'  
<- {  
  "return": {  
    "enabled": true,  
    "present": true  
  },  
  "id": "libvirt-8"  
}
```

In the above example, we ran the simple command **query-kvm** which checks if the host is capable of running KVM and if KVM is enabled.



Tip: Generating Human-Readable Output

To use the standard human-readable output format of QEMU instead of the JSON format, use the **--hmp** option:

```
tux > sudo virsh qemu-monitor-command vm_guest1 --hmp "query-kvm"
```


VI Managing Virtual Machines with LXC

- 31 Linux Containers **304**
- 32 Migration from LXC to `libvirt-lxc` **311**

31 Linux Containers

31.1 Setting Up LXC Distribution Containers

A container is a self-contained software that includes the code and all its dependencies of an application. A containerized application can be deployed quickly and run reliably in a computing environment.

To set up an LXC container, you need to create a root file system containing the guest distribution.

PROCEDURE 31.1: CREATING A ROOT FILE SYSTEM

There is currently no GUI to create a root file system. You will thus need to open a terminal and use **zypper** as user **root** to populate the new root file system. In the following steps, the new **root** file system will be created in **/PATH/T0/ROOTFS**.

1. Add the openSUSE Leap repository and the corresponding update repository to the new **root** file system:

```
root # zypper --root /PATH/T0/ROOTFS ar http://download.opensuse.org/distribution/
      leap/42.3/repo/oss/ OSS
root # zypper --root /PATH/T0/ROOTFS ar http://download.opensuse.org/update/
      leap/42.3/oss/ Update-OSS
```

2. Refresh the repositories:

```
root # zypper --root /PATH/T0/ROOTFS ref
```

3. Install a minimal system:

```
root # zypper --root /PATH/T0/ROOTFS in -t pattern minimal_base
```

4. Set the **root** password:

```
root # echo "ttyS0" >>/PATH/T0/ROOTFS/etc/securetty
root # echo "root:YOURPASSWD" | chpasswd -R /PATH/T0/ROOTFS
```

PROCEDURE 31.2: DEFINING THE CONTAINER

1. Start Virtual Machine Manager.

2. (Optional) If not already present, add a local LXC connection by clicking *File > Add Connection*.
Select *LXC (Linux Containers)* as the hypervisor and click *Connect*.
3. Select the *localhost (LXC)* connection and click *File New Virtual Machine* menu.
4. Activate *Operating system container* and click *Forward*.
5. Type the path to the root file system from *Procedure 31.1, "Creating a Root File System"* and click the *Forward* button.
6. Choose the maximum amount of memory and CPUs to allocate to the container. Then click the *Forward* button.
7. Type in a name for the container. This name will be used for all **virsh** commands on the container.
Click *Advanced options*. Select the network to connect the container to and click the *Finish* button: the container will then be created and started. A console will also be automatically opened.

PROCEDURE 31.3: CONFIGURING IP ADDRESSES FOR NETWORK INTERFACES

Network devices and hostdev devices with network capabilities can be provided with one or more IP addresses to set on the network device in the guest. However, some hypervisors or network device types will simply ignore them or only use the first one.

1. Edit the container XML configuration using **virsh**:

```
tux > virsh -c lxc:/// edit MYCONTAINER
```

2. The following example shows how to set one or multiple IP addresses:

```
[...]
<devices>
  <interface type='network'>
    <source network='default' />
    <target dev='vnet0' />
    <ip address='192.168.122.5' prefix='24' />
    <ip address='192.168.122.5' prefix='24' peer ①='10.0.0.10' />
    <route family ②='ipv4' address ③='192.168.122.0' prefix ④='24'
      gateway ⑤='192.168.122.1' />
    <route family ②='ipv4' address ③='192.168.122.8' gateway ⑤='192.168.122.1' />
  </interface>
[...]
```

```

<hostdev mode='capabilities' type='net'>
  <source>
    <interface>eth0</interface>
  </source>
  <ip address='192.168.122.6' prefix='24' />
  <route family='ipv4' address='192.168.122.0' prefix='24' gateway='192.168.122.1' />
  <route family='ipv4' address='192.168.122.8' gateway='192.168.122.1' />
</hostdev>
</devices>
[...]
```

- ① Optional attribute. Holds the IP address of the other end of a point-to-point network device.
 - ② Can be set to either ipv4 or ipv6.
 - ③ Contains the IP address.
 - ④ Optional parameter (will be automatically set if not specified). Defines the number of 1 bits in the netmask. For IPv4, the default prefix is determined according to the network “class” (A, B, or C). For IPv6, the default prefix is 64.
 - ⑤ If you do not specify a default gateway in the XML file, none will be set.
3. You can also add route elements to define IP routes to add in the guest. These are used by the LXC driver.

```

[...]
```

```

<devices>
  <interface type①='ethernet'>
    <source>②
      <ip address③='192.168.123.1' prefix='24' />
      <ip address④='10.0.0.10' prefix='24' peer='192.168.122.5' />
      <route⑤ family='ipv4' address='192.168.42.0' prefix='24'
        gateway='192.168.123.4' />
    </source>
    [...]
  </interface>
  [...]
</devices>
[...]
```

- ① Network devices of type ethernet can optionally be provided with one or multiple IP addresses (③, ④) and with one or multiple routes (⑤) to set on the host side of the network device.

These are configured as subelements of the `source` element (2) of the interface. They have the same attributes as the similarly named elements used to configure the guest side of the interface (see the step above).

- 3 First IP address for the network device of type `ethernet`.
- 4 Second IP address for the network device of type `ethernet`.
- 5 Route to set on the host side of the network device.

Find further details about the attributes of this element at <http://libvirt.org/formatnetwork.html#elementsStaticroute>.

4. Save the changes and exit the editor.



Note: Container Network

To configure the container network, edit the `/etc/sysconfig/network/ifcfg-*` files.

31.2 Setting Up LXC Application Containers

Libvirt also allows to run single applications instead of full blown Linux distributions in containers. In this example, `bash` will be started in its own container.

PROCEDURE 31.4: DEFINING AN APPLICATION CONTAINER USING YAST

1. Start Virtual Machine Manager.
2. (Optional) If not already present, add a local LXC connection by clicking *File > Add Connection*.
Select *LXC (Linux Containers)* as the hypervisor and click *Connect*.
3. Select the *localhost (LXC)* connection and click *File New Virtual Machine* menu.
4. Activate *Application container* and click *Forward*.
Set the path to the application to be launched. As an example, the field is filled with `/bin/sh`, which is fine to create a first container. Click *Forward*.
5. Choose the maximum amount of memory and CPUs to allocate to the container. Click *Forward*.
6. Type in a name for the container. This name will be used for all `virsh` commands on the container.

Click *Advanced options*. Select the network to connect the container to and click *Finish*. The container will be created and started. A console will be opened automatically. Note that the container will be destroyed after the application has finished running.

31.3 Securing a Container Using AppArmor

By default, containers are not secured using AppArmor or SELinux. There is no graphical user interface to change the security model for a libvirt domain, but virsh will help.

1. Edit the container XML configuration using virsh:

```
tux > virsh -c lxc:/// edit MYCONTAINER
```

2. Add the following to the XML configuration, save it and exit the editor.

```
<domain>
...
  <seclabel type="dynamic" model="apparmor"/>
...
</domain>
```

3. With this configuration, an AppArmor profile for the container will be created in the /etc/apparmor.d/libvirt directory. The default profile only allows the minimum applications to run in the container. This can be changed by modifying the libvirt-CONTAINER-uuid file: this file is not overwritten by libvirt.

31.4 Differences between the libvirt LXC Driver and LXC

openSUSE versions prior to Leap were shipping LXC, while openSUSE Leap comes with the libvirt LXC driver, sometimes named libvirt-lxc to avoid confusion. The containers are not managed or configured in the same way in these tools. Here is a non-exhaustive list of differences.

The main difference is that domain configuration in libvirt is an XML file, while LXC configuration is a properties file. Most of the LXC properties can be mapped to the domain XML. The properties that cannot be migrated are:

- *lxc.network.script.up*: this script can be implemented using the `/etc/libvirt/hooks/network` libvirt hook, though the script will need to be adapted.
- *lxc.network.ipv**: libvirt cannot set the container network configuration from the domain configuration.
- *lxc.network.name*: libvirt cannot set the container network card name.
- *lxc.devtydir*: libvirt does not allow changing the location of the console devices.
- *lxc.console*: there is currently no way to log the output of the console into a file on the host for libvirt LXC containers.
- *lxc.pivotdir*: libvirt does not allow to fine-tune the directory used for the `pivot_root`. `/.olroot` is used.
- *lxc.rootfs.mount*: libvirt does not allow to fine-tune this.

LXC VLAN networks automatically create the VLAN interface on the host and then move it into the guest namespace. libvirt-lxc configuration can mention a VLAN tag ID only for Open vSwitch tap devices or PCI pass-through of SR-IOV VF. The conversion tool actually needs the user to manually create the VLAN interface on the host side.

LXC rootfs can also be an image file, but LXC brute-forces the mount to try to detect the proper file system format. libvirt-lxc can mount image files of several formats, but the 'auto' value for the format parameter is explicitly not supported. This means that the generated configuration will need to be tweaked by the user to get a proper match in that case.

LXC can support any cgroup configuration, even future ones, while libvirt domain configuration, needs to map each of them.

LXC can mount block devices in the rootfs, but it cannot mount raw partition files: the file needs to be manually attached to a loop device. On the other hand libvirt-lxc can mount block devices, but also partition files of any format.

31.5 Sharing Namespaces across Containers

Like Docker Open Source Engine, libvirt allows you to inherit the namespace from containers or processes to share the network namespace. The following example shows how to share required namespaces.

```
<domain type='lxc' xmlns:lxc='http://libvirt.org/schemas/domain/lxc/1.0'>
  [...]
  <lxc:namespace>
    <lxc:sharenet type='netns' value='red' />
    <lxc:shareuts type='name' value='CONTAINER_1' />
    <lxc:shareipc type='pid' value='12345' />
  </lxc:namespace>
</domain>
```

The `netns` option is specific to `sharenet`. Use it to use an existing network namespace (instead of creating a new network namespace for the container). In this case, the `privnet` option will be ignored.

31.6 For More Information

LXC Container Driver

<http://libvirt.org/drvlxc.html> 

32 Migration from LXC to `libvirt-lxc`

Since openSUSE Leap, LXC is integrated into `libvirt` library. This decision has several advantages over using LXC as a separate solution—such as a unified approach with other virtualization solutions or independence on the kernel used. This chapter describes steps needed to migrate an existing LXC environment for use with the `libvirt` library.

32.1 Host Migration

The migration itself has two phases. You first need to migrate the host, then the LXC containers. After that, you can run the original containers as VM Guests in the `libvirt` environment.

PROCEDURE 32.1: HOST MIGRATION

1. Upgrade the host to openSUSE Leap 15 using the official DVD media.
2. After the upgrade, install the `libvirt-daemon-lxc` and `libvirt-daemon-config-network` packages.
3. Create a `libvirt` XML configuration `lxc_container.xml` from the existing container `lxc_container`:

```
tux > sudo virt-lxc-convert /etc/lxc/lxc_container/config > lxc_container.xml
```

4. Check if the network configuration on the host is the same as in the container configuration file, and fix it if needed.
5. Check the `lxc_container.xml` file for any weird or missing configuration. Note that some LXC configuration options cannot be mapped to `libvirt` configuration. Although the conversion should usually be fine, check [Section 31.4, “Differences between the libvirt LXC Driver and LXC”](#) for more details.
6. Define the container in `libvirt` based on the created XML definition:

```
tux > sudo virsh -c lxc:/// define lxc_container.xml
```

32.2 Container Migration

After the host is migrated, the LXC container in `libvirt` will not boot. It needs to be migrated to openSUSE Leap 15 as well to get everything working.

PROCEDURE 32.2: CONTAINER MIGRATION

1. The `baseproduct` file is missing (and `zypper` keeps complaining about it). Create the relevant symbolic link:

```
root # R00TFS=/var/lib/lxc/lxc_container/rootfs
root # ln -s $R00TFS/etc/products.d/SUSE_SLES.prod $R00TFS/etc/products.d/
baseproduct
```

2. Add the DVD repository. Note that you need to replace the DVD device with the one attached to your container:

3. Disable or remove previous repositories:

```
root # zypper --root $R00TFS lr
  | Alias                | Name                | Enabled | Refresh
-----+-----+-----+-----+-----
1 | SLES12                | SLES12              | Yes    | No
2 | SUSE-[...] -Server-12-SP3 38 | SUSE-[...] -Server-12-SP3 138 | Yes    | No

root # zypper --root $R00TFS rr 2
```

```
root # zypper --root $R00TFS ar \
cd:///?devices=/dev/dvd "openSUSE 15"
```

4. Disable or remove previous repositories:

```
root # zypper --root $R00TFS lr
  | Alias                | Name                | Enabled | Refresh
-----+-----+-----+-----+-----
1 | openSUSE 42.3 Main    | openSUSE 42.3 Main  | Yes    | No
2 | openSUSE 42.3 Update  | openSUSE 42.3 Update | Yes    | No

root # zypper --root $R00TFS rr 2
```

5. Upgrade the container:

```
root # zypper --root $R00TFS dup
```

6. Install the *Minimal* pattern to make sure everything required is installed:

```
root # zypper --root $ROOTFS in -t pattern Minimal
```

32.3 Starting the Container

After the host and container migration is complete, the container can be started:

```
root # virsh -c lxc:/// start lxc_container
```

If you need to get a console to view the logging messages produced by the container, run:

```
root # virsh -c lxc:/// console lxc_container
```

Glossary

General

Create Virtual Machine Wizard

A software program available in YaST and Virtual Machine Manager that provides a graphical interface to guide you through the steps to create virtual machines. It can also be run in text mode by entering `virt-install` at a command prompt in the host environment.

Dom0

The term is used in Xen environments, and refers to a virtual machine. The host operating system is actually a virtual machine running in a privileged domain and can be called Dom0. All other virtual machines on the host run in unprivileged domains and can be called domain U's.

hardware-assisted

Intel* and AMD* provide virtualization hardware-assisted technology. This reduces the frequency of VM IN/OUT (fewer VM traps), because software is a major source of overhead, and increases the efficiency (the execution is done by the hardware). Moreover, this reduces the memory footprint, provides better resource control, and allows secure assignment of specific I/O devices.

Host Environment

The desktop or command line environment that allows interaction with the host computer's environment. It provides a command line environment and can also include a graphical desktop, such as GNOME or IceWM. The host environment runs as a special type of virtual machine that has privileges to control and manage other virtual machines. Other commonly used terms include *Dom0*, privileged domain, and host operating system.

Hypervisor

The software that coordinates the low-level interaction between virtual machines and the underlying physical computer hardware.

KVM

See *Chapter 3, Introduction to KVM Virtualization*

Paravirtualized Frame Buffer

The video output device that drives a video display from a memory buffer containing a complete frame of data for virtual machine displays running in paravirtual mode.

VHS

Virtualization Host Server

The physical computer running a SUSE virtualization platform software. The virtualization environment consists of the hypervisor, the host environment, virtual machines, and associated tools, commands, and configuration files. Other commonly used terms include host, Host Computer, Host Machine (HM), Virtual Server (VS), Virtual Machine Host (VMH), and VM Host Server (VHS).

VirtFS

VirtFS is a new paravirtualized file system interface designed for improving pass-through technologies in the KVM environment. It is based on the VirtIO framework.

Virtual Machine

A virtualized PC environment (VM) capable of hosting a guest operating system and associated applications. Could be also called a VM Guest.

Virtual Machine Manager

A software program that provides a graphical user interface for creating and managing virtual machines.

Virtualized

A guest operating system or application running on a virtual machine.

Xen

See *Chapter 2, Introduction to Xen Virtualization*

xl

A set of commands for Xen that lets administrators manage virtual machines from a command prompt on the host computer. It replaced the deprecated `xm` tool stack.

CPU

CPU capping

Virtual CPU capping allows you to set vCPU capacity to 1–100 percent of the physical CPU capacity.

CPU hotplugging

CPU hotplugging is used to describe the functions of replacing/adding/removing a CPU without shutting down the system.

CPU over-commitment

Virtual CPU over-commitment is the ability to assign more virtual CPUs to VMs than the actual number of physical CPUs present in the physical system. This procedure does not increase the overall performance of the system, but might be useful for testing purposes.

CPU pinning

Processor affinity, or CPU pinning enables the binding and unbinding of a process or a thread to a central processing unit (CPU) or a range of CPUs.

Network

Bridged Networking

A type of network connection that lets a virtual machine be identified on an external network as a unique identity that is separate from and unrelated to its host computer.

Empty Bridge

A type of network bridge that has no physical network device or virtual network device provided by the host. This lets virtual machines communicate with other virtual machines on the same host but not with the host or on an external network.

External Network

The network outside a host's internal network environment.

Internal Network

A type of network configuration that restricts virtual machines to their host environment.

Local Bridge

A type of network bridge that has a virtual network device but no physical network device provided by the host. This lets virtual machines communicate with the host and other virtual machines on the host. Virtual machines can communicate on an external network through the host.

Network Address Translation (NAT)

A type of network connection that lets a virtual machine use the IP address and MAC address of the host.

No Host Bridge

A type of network bridge that has a physical network device but no virtual network device provided by the host. This lets virtual machines communicate on an external network but not with the host. This lets you separate virtual machine network communications from the host environment.

Traditional Bridge

A type of network bridge that has both a physical network device and a virtual network device provided by the host.

Storage

AHCI

The Advanced Host Controller Interface (AHCI) is a technical standard defined by Intel* that specifies the operation of Serial ATA (SATA) host bus adapters in a non-implementation-specific manner.

Block Device

Data storage devices, such as CD-ROM drives or disk drives, that move data in the form of blocks. Partitions and volumes are also considered block devices.

File-Backed Virtual Disk

A virtual disk based on a file, also called a disk image file.

Raw Disk

A method of accessing data on a disk at the individual byte level instead of through its file system.

Sparse image file

A disk image file that does not reserve its entire amount of disk space but expands as data is written to it.

xvda

The drive designation given to the first virtual disk on a paravirtual machine.

Linux Containers

cgroups

Kernel Control Groups (commonly called “cgroups”) are a kernel feature that allows aggregating or partitioning tasks (processes) and all their children into hierarchical organized groups to isolate resources.

See also *Book “System Analysis and Tuning Guide”, Chapter 9 “Kernel Control Groups”*.

chroot

A *change root* (chroot, or change root jail) is a section in the file system that is isolated from the rest of the file system. For this purpose, the chroot or pivot_root command is used to change the root of the file system. A program that is executed in such a “chroot jail” cannot access files outside the designated directory tree.

container

Can be seen as a kind of “virtual machine” on the host server that can run any Linux system, for example openSUSE, SUSE Linux Enterprise Desktop, or SUSE Linux Enterprise Server. The main difference with a normal virtual machine is that the container shares its kernel with the host it runs on.

Kernel namespaces

A kernel feature to isolate some resources like network, users, and others for a group of processes.

Acronyms

ACPI

Advanced Configuration and Power Interface (ACPI) specification provides an open standard for device configuration and power management by the operating system.

AER

Advanced Error Reporting

AER is a capability provided by the PCI Express specification which allows for reporting of PCI errors and recovery from some of them.

APIC

Advanced Programmable Interrupt Controller (APIC) is a family of interrupt controllers.

BDF

Bus:Device:Function

Notation used to succinctly describe PCI and PCIe devices.

CG

Control Groups

Feature to limit, account and isolate resource usage (CPU, memory, disk I/O, etc.).

EDF

Earliest Deadline First

This scheduler provides weighted CPU sharing in an intuitive way and uses real-time algorithms to ensure time guarantees.

EPT

Extended Page Tables

Performance in a virtualized environment is close to that in a native environment. Virtualization does create some overheads, however. These come from the virtualization of the CPU, the *MMU*, and the I/O devices. In some recent x86 processors AMD and Intel have begun to provide hardware extensions to help bridge this performance gap. In 2006, both vendors introduced their first generation hardware support for x86 virtualization with AMD-Virtualization (AMD-V) and Intel® VT-x technologies. Recently Intel introduced its second generation of hardware support that incorporates MMU-virtualization, called Extended Page

Tables (EPT). EPT-enabled systems can improve performance compared to using shadow paging for *MMU* virtualization. EPT increases memory access latencies for a few workloads. This cost can be reduced by effectively using large pages in the guest and the hypervisor.

FLASK

Flux Advanced Security Kernel

Xen implements a type of mandatory access control via a security architecture called FLASK using a module of the same name.

HAP

High Assurance Platform

HAP combines hardware and software technologies to improve workstation and network security.

HVM

Hardware Virtual Machine (commonly called like this by Xen).

IOMMU

Input/Output Memory Management Unit

IOMMU (AMD* technology) is a memory management unit (*MMU*) that connects a direct memory access-capable (DMA-capable) I/O bus to the main memory.

KSM

Kernel Same Page Merging

KSM allows for automatic sharing of identical memory pages between guests to save host memory. KVM is optimized to use KSM if enabled on the VM Host Server.

MMU

Memory Management Unit

is a computer hardware component responsible for handling accesses to memory requested by the CPU. Its functions include translation of virtual addresses to physical addresses (that is, virtual memory management), memory protection, cache control, bus arbitration and in simpler computer architectures (especially 8-bit systems) bank switching.

PAE

Physical Address Extension

32-bit x86 operating systems use Physical Address Extension (PAE) mode to enable addressing of more than 4 GB of physical memory. In PAE mode, page table entries (PTEs) are 64 bits in size.

PCID

Process-context identifiers

These are a facility by which a logical processor may cache information for multiple linear-address spaces so that the processor may retain cached information when software switches to a different linear address space. `INVPID` instruction is used for fine-grained *TLB* flush, which is benefit for kernel.

PCIe

Peripheral Component Interconnect Express

PCIe was designed to replace older PCI, PCI-X and AGP bus standards. PCIe has numerous improvements including a higher maximum system bus throughput, a lower I/O pin count and smaller physical footprint. Moreover it also has a more detailed error detection and reporting mechanism (*AER*), and a native hotplug functionality. It is also backward compatible with PCI.

PSE and PSE36

Page Size Extended

PSE refers to a feature of x86 processors that allows for pages larger than the traditional 4 KiB size. PSE-36 capability offers 4 more bits, in addition to the normal 10 bits, which are used inside a page directory entry pointing to a large page. This allows a large page to be located in 36-bit address space.

PT

Page Table

A page table is the data structure used by a virtual memory system in a computer operating system to store the mapping between virtual addresses and physical addresses. Virtual addresses are those unique to the accessing process. Physical addresses are those unique to the hardware (RAM).

QXL

QXL is a cirrus VGA framebuffer (8M) driver for virtualized environment.

RVI or NPT

Rapid Virtualization Indexing, Nested Page Tables

An AMD second generation hardware-assisted virtualization technology for the processor memory management unit (*MMU*).

SATA

Serial ATA

SATA is a computer bus interface that connects host bus adapters to mass storage devices such as hard disks and optical drives.

Seccomp2-based sandboxing

Sandboxed environment where only predetermined system calls are permitted for added protection against malicious behavior.

SMEP

Supervisor Mode Execution Protection

This prevents the execution of user-mode pages by the Xen hypervisor, making many application-to-hypervisor exploits much harder.

SPICE

Simple Protocol for Independent Computing Environments

SXP

An SXP file is a Xen Configuration File.

TCG

Tiny Code Generator

Instructions are emulated rather than executed by the CPU.

THP

Transparent Huge Pages

This allows CPUs to address memory using pages larger than the default 4 KB. This helps reduce memory consumption and CPU cache usage. KVM is optimized to use THP (via mad-*write* and opportunistic methods) if enabled on the VM Host Server.

TLB

Translation Lookaside Buffer

TLB is a cache that memory management hardware uses to improve virtual address translation speed. All current desktop, notebook, and server processors use a TLB to map virtual and physical address spaces, and it is nearly always present in any hardware that uses virtual memory.

VCPU

A scheduling entity, containing each state for virtualized CPU.

VDI

Virtual Desktop Infrastructure

VFIO

Since kernel v3.6; a new method of accessing PCI devices from user space called VFIO.

VHS

Virtualization Host Server

VM root

VMM will run in *VMX* root operation and guest software will run in *VMX* non-root operation. Transitions between *VMX* root operation and *VMX* non-root operation are called *VMX* transitions.

VMCS

Virtual Machine Control Structure

VMX non-root operation and VMX transitions are controlled by a data structure called a virtual-machine control structure (VMCS). Access to the VMCS is managed through a component of processor state called the VMCS pointer (one per logical processor). The value of the VMCS pointer is the 64-bit address of the VMCS. The VMCS pointer is read and written using the instructions VMPTRST and VMPTRLD. The *VMM* configures a VMCS using the VMREAD, VMWRITE, and VMCLEAR instructions. A *VMM* could use a different VMCS for each virtual machine that it supports. For a virtual machine with multiple logical processors (virtual processors), the *VMM* could use a different VMCS for each virtual processor.

VMDq

Virtual Machine Device Queue

Multi-queue network adapters exist which support multiple VMs at the hardware level, having separate packet queues associated to the different hosted VMs (by means of the IP addresses of the VMs).

VMM

Virtual Machine Monitor (Hypervisor)

When the processor encounters an instruction or event of interest to the Hypervisor (*VMM*), it exits from guest mode back to the VMM. The VMM emulates the instruction or other event, at a fraction of native speed, and then returns to guest mode. The transitions from guest mode to the VMM and back again are high-latency operations, during which guest execution is completely stalled.

VMX

Virtual Machine eXtensions

VPID

New support for software control of *TLB* (VPID improves *TLB* performance with small *VMM* development effort).

VT-d

Virtualization Technology for Directed I/O

Like *IOMMU* for Intel* (<https://software.intel.com/en-us/articles/intel-virtualization-technology-for-directed-io-vt-d-enhancing-intel-platforms-for-efficient-virtualization-of-io-devices>)⁷.

vTPM

Component to establish end-to-end integrity for guests via Trusted Computing.

A Appendix

B XM, XL Toolstacks and Libvirt framework

B.1 Xen Toolstacks

Since the early Xen 2.x releases, **xend** has been the de facto toolstack for managing Xen installations. In Xen 4.1, a new toolstack called libxenlight (also known as libxl) was introduced with technology preview status. libxl is a small, low-level library written in C. It has been designed to provide a simple API for all client toolstacks (XAPI (<http://wiki.xen.org/wiki/XAPI>), **libvirt**, **xl**). In Xen 4.2, libxl was promoted to officially supported status and **xend** was marked deprecated. **xend** has been included in the Xen 4.3 and 4.4 series to give users ample time to convert their tooling to libxl. It has been removed from the upstream Xen project and will no longer be provided starting with the Xen 4.5 series and openSUSE Leap 42.1.

. Starting with openSUSE Leap 42.1, **xend** is no longer supported.

One of the major differences between **xend** and libxl is that the former is stateful, while the latter is stateless. With **xend**, all client applications such as **xm** and **libvirt** see the same system state. **xend** is responsible for maintaining state for the entire Xen host. In libxl, client applications such as **xl** or **libvirt** must maintain state. Thus domains created with **xl** or not visible or known to other libxl applications such as **libvirt**. Generally, it is discouraged to mix and match libxl applications and is preferred that a single libxl application be used to manage a Xen host. In openSUSE Leap, we recommend to use **libvirt** to manage Xen hosts. This allows management of the Xen system through **libvirt** applications such as **virt-manager**, **virt-install**, **virt-viewer**, libguestfs, etc. If **xl** is used to manage the Xen host, any virtual machines under its management will not be accessible to **libvirt**. Hence, they are not accessible to any of the **libvirt** applications.

B.1.1 Upgrading from xend/xm to xl/libxl

The **xl** application, along with its configuration format (see **man xl.cfg**), was designed to be backward-compatible with the **xm** application and its configuration format (see **man xm.cfg**). Existing **xm** configuration should be usable with **xl**. Since libxl is stateless, and **xl** does not support the notion of managed domains, SUSE recommends using **libvirt** to manage Xen hosts. SUSE has provided a tool called **xen2libvirt**, which provides a simple mechanism to import domains previously managed by **xend** into **libvirt**. See [Section B.2, "Import Xen Domain Configuration into libvirt"](#) for more information on **xen2libvirt**.

B.1.2 XL design

The basic structure of every `xl` command is:

```
xl subcommand OPTIONS DOMAIN
```

`DOMAIN` is the numeric domain id, or the domain name (which will be internally translated to domain id), and `OPTIONS` are subcommand specific options.

Although `xl/libxl` was designed to be backward-compatible with `xm/xend`, there are a few differences that should be noted:

- Managed or persistent domains. `libvirt` now provides this functionality.
- `xl/libxl` does not support Python code in the domain configuration files.
- `xl/libxl` does not support creating domains from SXP format configuration files (`xm create -F`).
- `xl/libxl` does not support sharing storage across DomU's via `w!` in domain configuration files.

`xl/libxl` is relatively new and under heavy development, hence a few features are still missing with regard to the `xm/xend` toolstack:

- SCSI LUN/Host pass-through (PVSCSI)
- USB pass-through (PVUSB)
- Direct Kernel Boot for fully virtualized Linux guests for Xen

B.1.3 Checklist before Upgrade

Before upgrading a Leap 42.1 Xen host to Leap 15:

- You must remove any Python code from your `xm` domain configuration files.
- It is recommended to capture the `libvirt` domain XML from all existing virtual machines using `virsh dumpxml DOMAIN_NAME DOMAIN_NAME.xml`.
- It is recommended to do a backup of `/etc/xen/xend-config.sxp` and `/boot/grub/menu.lst` files to keep references of previous parameters used for Xen.



Note

Currently, live migrating virtual machines running on a Leap 42.1 Xen host to a Leap 15 Xen host is not supported. The `xend` and `libxl` toolstacks are not runtime-compatible. Virtual machine downtime will be required to move the virtual machines.

B.2 Import Xen Domain Configuration into `libvirt`

`xen2libvirt` is a command line tool to import legacy Xen domain configuration into the `libvirt` virtualization library (see The Virtualization book for more information on `libvirt`). `xen2libvirt` provides an easy way to import domains managed by the deprecated `xm/xend` tool stack into the new `libvirt/libxl` tool stack. Several domains can be imported at once using its `--recursive mode`

`xen2libvirt` is included in the `xen-tools` package. If needed, install it with

```
tux > sudo zypper install xen-tools
```

`xen2libvirt` general syntax is

```
xen2libvirt <options> /path/to/domain/config
```

where `options` can be:

`-h, --help`

Prints short information about `xen2libvirt` usage.

`-c, --convert-only`

Converts the domain configuration to the `libvirt` XML format, but does not do the import to `libvirt`.

`-r, --recursive`

Converts and/or imports all domains configuration recursively, starting at the specified path.

`-f, --format`

Specifies the format of the source domain configuration. Can be either `xm`, or `sexpr` (S-expression format).

`-v, --verbose`

Prints more detailed information about the import process.

EXAMPLE B.1: CONVERTING XEN DOMAIN CONFIGURATION TO `libvirt`

Suppose you have a Xen domain managed with `xm` with the following configuration saved in `/etc/xen/sle12.xm`:

```
kernel = "/boot/vmlinuz-2.6-xenU"
memory = 128
name = "SLE12"
root = "/dev/hda1 ro"
disk = [ "file:/var/xen/sle12.img,hda1,w" ]
```

Convert it to `libvirt` XML without importing it, and look at its content:

```
tux > sudo xen2libvirt -f xm -c /etc/xen/sle12.xm > /etc/libvirt/qemu/sles12.xml
# cat /etc/libvirt/qemu/sles12.xml
<domain type='xen'>
  <name>SLE12</name>
  <uuid>43e1863c-8116-469c-a253-83d8be09aa1d</uuid>
  <memory unit='KiB'>131072</memory>
  <currentMemory unit='KiB'>131072</currentMemory>
  <vcpu placement='static'>1</vcpu>
  <os>
    <type arch='x86_64' machine='xenpv'>linux</type>
    <kernel>/boot/vmlinuz-2.6-xenU</kernel>
  </os>
  <clock offset='utc' adjustment='reset'/>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>restart</on_crash>
  <devices>
    <disk type='file' device='disk'>
      <driver name='file'/>
      <source file='/var/xen/sle12.img'/>
      <target dev='hda1' bus='xen'/>
    </disk>
    <console type='pty'>
      <target type='xen' port='0'/>
    </console>
  </devices>
</domain>
```

To import the domain into `libvirt`, you can either run the same `xen2libvirt` command without the `-c` option, or use the exported file `/etc/libvirt/qemu/sles12.xml` and define a new Xen domain using `virsh`:

```
tux > sudo virsh define /etc/libvirt/qemu/sles12.xml
```

B.3 Differences between the `xm` and `xl` Applications

The purpose of this chapter is to list all differences between `xm` and `xl` applications. Generally, `xl` is designed to be compatible with `xm`. Replacing `xm` with `xl` in custom scripts or tools is usually sufficient.

You can also use the `libvirt` framework using the `virsh` command. In this documentation only the first `OPTION` for `virsh` will be shown. To get more help on this option do a:

```
virsh help OPTION
```

B.3.1 Notation Conventions

To easily understand the difference between `xl` and `xm` commands, the following notation is used in this section:

TABLE B.1: NOTATION CONVENTIONS

Notation	Meaning
(-) minus	Option exists in <code>xm</code> , but <code>xl</code> does not include it.
(+) plus	Option exists in <code>xl</code> , but <code>xm</code> does not include it.

B.3.2 New Global Options

TABLE B.2: NEW GLOBAL OPTIONS

Options	Task
(+) <code>-v</code>	Verbose, increase the verbosity of the output
(+) <code>-N</code>	Dry run, do not actually execute the command

Options	Task
(+) <u>-f</u>	Force execution. <u>xl</u> will refuse to run some commands if it detects that <u>xend</u> is also running, this option will force the execution of those commands, even though it is unsafe

B.3.3 Unchanged Options

List of common options of xl and xm, and their libvirt equivalents.

TABLE B.3: COMMON OPTIONS

Options	Task	<u>libvirt</u> equivalent
destroy <u>DOMAIN</u>	Immediately terminate the domain.	<u>virsh</u> <u>destroy</u>
domid <u>DOMAIN_NAME</u>	Convert a domain name to a <u>DOMAIN_ID</u> .	<u>virsh</u> <u>domid</u>
domname <u>DOMAIN_ID</u>	Convert a <u>DOMAIN_ID</u> to a <u>DOMAIN_NAME</u> .	<u>virsh</u> <u>domname</u>
help	Display the short help message (that is, common commands).	<u>virsh</u> <u>help</u>
pause <u>DOMAIN_ID</u>	Pause a domain. When in a paused state, the domain will still consume allocated resources such as memory, but will not be eligible for scheduling by the Xen hypervisor.	<u>virsh</u> <u>suspend</u>

Options	Task	<u>libvirt</u> equivalent
unpause <u>DOMAIN_ID</u>	Move a domain out of the paused state. This will allow a previously paused domain to be eligible for scheduling by the Xen hypervisor.	<u>virsh</u> <u>resume</u>
rename <u>DOMAIN_ID</u> <u>NEW_DOMAIN_NAME</u>	Change the domain name of <u>DOMAIN_ID</u> to <u>NEW_DOMAIN_NAME</u> .	<ol style="list-style-type: none"> <li data-bbox="1050 510 1414 636">1. <pre>tux > virsh dumpxml DOMAINNAME > DOMXML</pre> <li data-bbox="1050 667 1414 748">2. modify the domain's name in <u>DOMXML</u> <li data-bbox="1050 779 1414 882">3. <pre>tux > virsh undefine DOMAINNAME</pre> <li data-bbox="1050 913 1414 994">4. <pre>tux > virsh define DOMAINNAME</pre>
sysrq <u>DOMAIN</u> <letter>	Send a Magic System Request to the domain, each type of request is represented by a different letter. It can be used to send SysRq requests to Linux guests, see https://www.kernel.org/doc/html/latest/admin-guide/sysrq.html for more information. It requires PV drivers to be installed in your guest OS.	<u>virsh</u> <u>send-keys</u> can send Magic Sys Req only for KVM
vncviewer <u>OPTIONS</u> <u>DOMAIN</u>	Attach to domain's VNC server, forking a <u>vncviewer</u> process.	<u>virt-viewer</u> <u>DOMAIN_ID</u> <u>virsh</u> <u>VNCDISPLAY</u>

Options	Task	<u>libvirt</u> equivalent
<u>vcpu-set</u> <u>DOMAIN_ID</u> <u>VCPUS</u>	Enable the vcpu-count virtual CPUs for the domain in question. Like <u>mem-set</u> , this command can only allocate up to the maximum virtual CPU count configured at boot for the domain.	<u>virsh</u> <u>setvcpus</u>
<u>vcpu-list</u> <u>DOMAIN_ID</u>	List VCPU information for a specific domain. If no domain is specified, VCPU information for all domains will be provided.	<u>virsh</u> <u>vcpuinfo</u>
<u>vcpu-pin</u> <u>DOMAIN_ID</u> <VCPUs all> <CPUs all>	Pin the VCPU to only run on the specific CPUs. The keyword all can be used to apply the CPU list to all VCPUs in the domain.	<u>virsh</u> <u>vcpupin</u>
<u>dmesg</u> [-c]	Read the Xen message buffer, similar to dmesg on a Linux system. The buffer contains informational, warning, and error messages created during Xen's boot process.	
<u>top</u>	Execute the <u>xentop</u> command, which provides real time monitoring of domains. <u>xentop</u> is a curses interface.	<u>virsh</u> <u>nodecpustats</u> <u>virsh</u> <u>nodememstats</u>

Options	Task	<u>libvirt</u> equivalent
<u>uptime</u> [-s] <u>DOMAIN</u>	Print the current uptime of the domains running. With the <u>xl</u> command, the <u>DOMAIN</u> argument is mandatory.	
<u>debug-keys</u> <u>KEYS</u>	Send debug keys to Xen. It is the same as pressing the Xen <i>conswitch</i> (Ctrl-A by default) three times and then pressing "keys".	
<u>cpupool-migrate</u> <u>DOMAIN</u> <u>CPU_POOL</u>	Move a domain specified by <u>DOMAIN_ID</u> or <u>DOMAIN</u> into a <u>CPU_POOL</u> .	
<u>cpupool-destroy</u> <u>CPU_POOL</u>	Deactivate a cpu pool. This is possible only if no domain is active in the cpu-pool.	
<u>block-detach</u> <u>DOMAIN_ID</u> <u>DevId</u>	Detach a domain's virtual block device. <i>devId</i> may be the symbolic name or the numeric device id given to the device by Dom0. You will need to run <u>xl</u> <u>block-list</u> to determine that number.	<u>virsh</u> <u>detach-disk</u>
<u>network-attach</u> <u>DOMAIN_ID</u> <u>NETWORK_DEVICE</u>	Create a new network device in the domain specified by <u>DOMAIN_ID</u> . <u>network-device</u> describes the device to attach, using the same format as the vif string in the domain configuration file	<u>virsh</u> <u>attach-interface</u> <u>virsh</u> <u>attach-device</u>

Options	Task	<u>libvirt</u> equivalent
<u>pci-attach</u> <u>DOMAIN</u> <BDF> [Virtual Slot]	Hotplug a new pass-through PCI device to the specified domain. <i>BDF</i> is the PCI Bus/Device/Function of the physical device to be passed through.	<u>virsh</u> <u>attach-device</u>
<u>pci-list</u> <u>DOMAIN_ID</u>	List pass-through PCI devices for a domain	
<u>getenforce</u>	Determine if the <i>FLASK</i> security module is loaded and enforcing its policy.	
<u>setenforce</u> <1 0 Enforcing Permissive>	Enable or disable enforcing of the <i>FLASK</i> access controls. The default is permissive and can be changed using the <code>flask_enforcing</code> option on the hypervisor's command line.	

B.3.4 Removed Options

List of xm options which are no more available with the XL tool stack and a replacement solution if available.

B.3.4.1 Domain Management

The list of Domain management removed command and their replacement.

TABLE B.4: DOMAIN MANAGEMENT REMOVED OPTIONS

Domain Management Removed Options		
Options	Task	Equivalent
(-) <u>log</u>	Print the Xend log.	This log file can be found in <u>/var/log/xend.log</u>
(-) <u>delete</u>	Remove a domain from Xend domain management. The <u>list</u> option shows the domain names	virsh <u>undefine</u>
(-) <u>new</u>	Adds a domain to Xend domain management	virsh <u>define</u>
(-) <u>start</u>	Start a Xend managed domain that was added using the <u>xm new</u> command	virsh <u>start</u>
(-) <u>dryrun</u>	Dry run - prints the resulting configuration in <i>SXP</i> but does not create the domain	<u>xl -N</u>
(-) <u>reset</u>	Reset a domain	virsh <u>reset</u>
(-) <u>domstate</u>	Show domain state	virsh <u>domstate</u>
(-) <u>serve</u>	Proxy Xend XMLRPC over stdio	
(-) <u>resume</u> <u>DOMAIN</u> <u>OPTIONS</u>	Moves a domain out of the suspended state and back into memory	virsh <u>resume</u>
(-) <u>suspend</u> <u>DOMAIN</u>	Suspend a domain to a state file so that it can be later resumed using the <u>resume</u> subcommand. Similar to	virsh <u>managesave</u> virsh <u>suspend</u>

Domain Management Removed Options		
Options	Task	Equivalent
	the <u>save</u> subcommand although the state file may not be specified	

B.3.4.2 USB Devices

USB options are not available with xl/libxl tool stack. virsh has the attach-device and detach-device options but it does not work yet with USB.

TABLE B.5: USB DEVICES MANAGEMENT REMOVED OPTIONS

USB Devices Management Removed Options	
Options	Task
(-) <u>usb-add</u>	Add a new USB physical bus to a domain
(-) <u>usb-del</u>	Delete a USB physical bus from a domain
(-) <u>usb-attach</u>	Attach a new USB physical bus to domain's virtual port
(-) <u>usb-detach</u>	Detach a USB physical bus from domain's virtual port
(-) <u>usb-list</u>	List domain's attachment state of all virtual port
(-) <u>usb-list-assignable-devices</u>	List all the assignable USB devices
(-) <u>usb-hc-create</u>	Create a domain's new virtual USB host controller
(-) <u>usb-hc-destroy</u>	Destroy a domain's virtual USB host controller

B.3.4.3 CPU Management

CPU management options has changed. New options are available, see: [Section B.3.5.10, “x1 cpupool -*”](#)

TABLE B.6: CPU MANAGEMENT REMOVED OPTIONS

CPU Management Removed Options	
Options	Task
(-) <u>cpupool-new</u>	Adds a CPU pool to Xend CPU pool management
(-) <u>cpupool-start</u>	Starts a Xend CPU pool
(-) <u>cpupool-delete</u>	Removes a CPU pool from Xend management

B.3.4.4 Other Options

TABLE B.7: OTHER OPTIONS

Other Removed Options	
Options	Task
(-) <u>shell</u>	Launch an interactive shell
(-) <u>change-vnc-passwd</u>	Change vnc password
(-) <u>vtpm-list</u>	List virtual TPM devices
(-) <u>block-configure</u>	Change block device configuration

B.3.5 Changed Options

B.3.5.1 create

x1 create CONFIG_FILE OPTIONS VARs



Note: libvirt Equivalent:

virsh create

TABLE B.8: **xL create** CHANGED OPTIONS

<u>create</u> Changed Options	
Options	Task
(*) <u>-f=FILE</u> , <u>--defconfig=FILE</u>	Use the given configuration file

TABLE B.9: **xm create** REMOVED OPTIONS

<u>create</u> Removed Options	
Options	Task
(-) <u>-s</u> , <u>--skipdtd</u>	Skip DTD checking - skips checks on XML before creating
(-) <u>-x</u> , <u>--xmldryrun</u>	XML dry run
(-) <u>-F=FILE</u> , <u>--config=FILE</u>	Use the given <i>SXP</i> formatted configuration script
(-) <u>--path</u>	Search path for configuration scripts
(-) <u>--help_config</u>	Print the available configuration variables (vars) for the configuration script
(-) <u>-n</u> , <u>--dryrun</u>	Dry run — prints the configuration in <i>SXP</i> but does not create the domain
(-) <u>-c</u> , <u>--console_autoconnect</u>	Connect to the console after the domain is created
(-) <u>-q</u> , <u>--quiet</u>	Quiet mode
(-) <u>-p</u> , <u>--paused</u>	Leave the domain paused after it is created

TABLE B.10: `xl create` ADDED OPTIONS

<code>create</code> Added Options	
<i>Options</i>	<i>Task</i>
(+) <code>-V</code> , <code>--vncviewer</code>	Attach to domain's VNC server, forking a vncviewer process
(+) <code>-A</code> , <code>--vncviewer-autopass</code>	Pass VNC password to vncviewer via stdin

B.3.5.2 `console`

`xl console` *OPTIONS* *DOMAIN*



Note: libvirt Equivalent

`virsh console`

TABLE B.11: `xl console` ADDED OPTIONS

<code>console</code> Added Option	
<i>Option</i>	<i>Task</i>
(+) <code>-t</code> [pv serial]	Connect to a PV console or connect to an emulated serial console. PV consoles are the only consoles available for PV domains while HVM domains can have both

B.3.5.3 `info`

`xl info`

TABLE B.12: `xm info` REMOVED OPTIONS

<code>info</code> Removed Options	
<i>Options</i>	<i>Task</i>
(-) <code>-n</code> , <code>--numa</code>	Numa info

<u>info</u> Removed Options	
<i>Options</i>	<i>Task</i>
(-) <u>-c</u> , <u>--config</u>	List Xend configuration parameters

B.3.5.4 `dump-core`

xl `dump-core` DOMAIN FILENAME



Note: libvirt Equivalent

virsh `dump`

TABLE B.13: `xl dump-core` REMOVED OPTIONS

<u>dump-core</u> Removed Options	
<i>Options</i>	<i>Task</i>
(-) <u>-L</u> , <u>--live</u>	Dump core without pausing the domain
(-) <u>-C</u> , <u>--crash</u>	Crash domain after dumping core
(-) <u>-R</u> , <u>--reset</u>	Reset domain after dumping core

B.3.5.5 `list`

xl `list` options DOMAIN



Note: libvirt Equivalent

virsh `list --all`

TABLE B.14: `xm list` REMOVED OPTIONS

<code>list</code> Removed Options	
<i>Options</i>	<i>Task</i>
(-) <code>-l</code> , <code>--long</code>	The output for <code>xm list</code> presents the data in <i>SXP</i> format
(-) <code>--state==STATE</code>	Output information for VMs in the specified state

TABLE B.15: `xl list` ADDED OPTIONS

<code>list</code> Added Options	
<i>Options</i>	<i>Task</i>
(+) <code>-Z</code> , <code>--context</code>	Also prints the security labels
(+) <code>-v</code> , <code>--verbose</code>	Also prints the domain UUIDs, the shutdown reason and security labels

B.3.5.6 `mem-*`



Note: libvirt Equivalent

`virsh setmem`

`virsh setmaxmem`

TABLE B.16: `xl mem-*` CHANGED OPTIONS

<code>mem-*</code> Changed Options	
<i>Options</i>	<i>Task</i>
<code>mem-max</code> <code>DOMAIN_ID</code> <code>MEM</code>	Appending <code>t</code> for terabytes, <code>g</code> for gigabytes, <code>m</code> for megabytes, <code>k</code> for kilobytes and <code>b</code> for bytes. Specify the maximum amount of memory the domain can use.

<u>mem-*</u> Changed Options	
Options	Task
<u>mem-set</u> <u>DOMAIN_ID</u> <u>MEM</u>	Set the domain's used memory using the balloon driver

B.3.5.7 `migrate`

`xl migrate` OPTIONS DOMAIN HOST



Note: libvirt Equivalent

`virsh migrate --live hvm-sles11-qcow2 xen+ CONNECTOR://USER@IP_ADDRESS/`

TABLE B.17: `xm migrate` REMOVED OPTIONS

<u>migrate</u> Removed Options	
Options	Task
(-) <u>-l, --live</u>	Use live migration. This will migrate the domain between hosts without shutting down the domain
(-) <u>-r, --resource Mbs</u>	Set maximum Mbs allowed for migrating the domain
(-) <u>-c, --change_home_server</u>	Change home server for managed domains
(-) <u>--max_iters= MAX_ITERS</u>	Number of iterations before final suspend (default:30)
(-) <u>--max_factor= MAX_FACTOR</u>	Max amount of memory to transfer before final suspend (default: 3*RAM).
(-) <u>--min_remaining= MIN_REMAINING</u>	Number of dirty pages before final suspend (default:50)

<u>migrate</u> Removed Options	
Options	Task
(-) <u>--abort_if_busy</u>	Abort migration instead of doing final suspend
(-) <u>--log_progress</u>	Log progress of migration to <u>xend.log</u>
(-) <u>-s</u> , <u>--ssl</u>	Use ssl connection for migration

TABLE B.18: **xL migrate** ADDED OPTIONS

<u>migrate</u> Added Options	
Options	Task
(+) <u>-s</u> <u>SSHCOMMAND</u>	Use <sshcommand> instead of <u>ssh</u>
(+) <u>-e</u>	On the new host, do not wait in the background (on <host>) for the death of the domain
(+) <u>-C</u> <u>CONFIG</u>	Send <config> instead of the configuration file used when creating the domain

B.3.5.8 Domain Management

xL reboot OPTIONS DOMAIN



Note: libvirt Equivalent

virsh reboot

TABLE B.19: **xm reboot** REMOVED OPTIONS

<u>reboot</u> Removed Options	
Options	Task
(-) <u>-a</u> , <u>--all</u>	Reboot all domains

<u>reboot</u> Removed Options	
<i>Options</i>	<i>Task</i>
(-) <u>-w</u> , <u>--wait</u>	Wait for reboot to complete before returning. This may take a while, as all services in the domain need to be shut down cleanly

TABLE B.20: **xL** reboot **ADDED OPTIONS**

<u>reboot</u> Added Options	
<i>Option</i>	<i>Task</i>
(+) <u>-F</u>	Fallback to ACPI reset event for HVM guests with no PV drivers

xL save *OPTIONS* *DOMAIN* *CHECK_POINT_FILE* *CONFIG_FILE*



Note: libvirt Equivalent

virsh save

TABLE B.21: **xL** save **ADDED OPTIONS**

<u>save</u> Added Options	
<i>Option</i>	<i>Task</i>
(+) <u>-c</u>	Leave domain running after creating the snapshot

xL restore *OPTIONS* *CONFIG_FILE* *CHECK_POINT_FILE*



Note: libvirt Equivalent

virsh restore

TABLE B.22: `xl` restore ADDED OPTIONS

<code>restore</code> Added Options	
<i>Options</i>	<i>Task</i>
(+) <code>-p</code>	Do not unpause domain after restoring it
(+) <code>-e</code>	Do not wait in the background for the death of the domain on the new host
(+) <code>-d</code>	Enable debug messages
(+) <code>-V, --vncviewer</code>	Attach to domain's VNC server, forking a vncviewer process
(+) <code>-A, --vncviewer-autopass</code>	Pass VNC password to vncviewer via stdin

`xl shutdown OPTIONS DOMAIN`



Note: libvirt Equivalent

`virsh shutdown`

TABLE B.23: `xm` shutdown REMOVED OPTIONS

<code>shutdown</code> Removed Options	
<i>Options</i>	<i>Task</i>
(-) <code>-w, --wait</code>	Wait for the domain to complete shutdown before returning
(-) <code>-a</code>	Shutdown all guest domains
(-) <code>-R</code>	
(-) <code>-H</code>	

TABLE B.24: **xL** shutdown ADDED OPTIONS

<u>shutdown</u> Added Options	
<i>Option</i>	<i>Task</i>
(+) <u>-F</u>	If the guest does not support PV shutdown control then fallback to sending an ACPI power event

TABLE B.25: **xL** trigger CHANGED OPTIONS

<u>trigger</u> Changed Options	
<i>Option</i>	<i>Task</i>
<u>_ trigger DOMAIN</u> <nmi reset init power sleep s3resume> <u>VCPU</u>	Send a trigger to a domain. Only available for HVM domains

B.3.5.9 **xL** sched-*

xL sched-credit OPTIONS



Note: libvirt Equivalent
virsh schedinfo

TABLE B.26: **xm** sched-credit REMOVED OPTIONS

<u>sched-credit</u> Removed Options	
<i>Options</i>	<i>Task</i>
<u>-d DOMAIN</u> , <u>--domain= DOMAIN</u>	Domain
<u>-w WEIGHT</u> , <u>--weight= WEIGHT</u>	A domain with a weight of 512 will get twice as much CPU as a domain with a weight of 256 on a contended host. Legal weights range from 1 to 65535 and the default is 256

<u>sched-credit</u> Removed Options	
<i>Options</i>	<i>Task</i>
<u>-c</u> <u>CAP</u> , <u>--cap= CAP</u>	The CAP optionally fixes the maximum amount of CPU a domain can consume

TABLE B.27: **x1** sched-credit **ADDED OPTIONS**

<u>sched-credit</u> Added Options	
<i>Options</i>	<i>Task</i>
(+) <u>-p</u> <u>CPUPPOOL</u> , <u>--cpupool= CPUPPOOL</u>	Restrict output to domains in the specified cpupool
(+) <u>-s</u> , <u>--schedparam</u>	Specify to list or set pool-wide scheduler parameters
(+) <u>-t</u> <u>TSLICE</u> , <u>--tslice_ms= TSLICE</u>	Timeslice tells the scheduler how long to allow VMs to run before pre-empting
(+) <u>-r</u> <u>RLIMIT</u> , <u>--ratelimit_us= RLIMIT</u>	Ratelimit attempts to limit the number of schedules per second

x1 sched-credit2 OPTIONS



Note: libvirt Status

virsh only supports credit scheduler, not credit2 scheduler

TABLE B.28: **xm** sched-credit2 **REMOVED OPTIONS**

<u>sched-credit2</u> Removed Options	
<i>Options</i>	<i>Task</i>
<u>-d</u> <u>DOMAIN</u> , <u>--domain= DOMAIN</u>	Domain
<u>-w</u> <u>WEIGHT</u> , <u>--weight= WEIGHT</u>	Legal weights range from 1 to 65535 and the default is 256

TABLE B.29: **x1** sched-credit2 ADDED OPTIONS

<u>sched-credit2</u> Added Options	
Option	Task
(+) <u>-p CPUPOOL</u> , <u>--cpupool= CPUPOOL</u>	Restrict output to domains in the specified cpupool

x1 sched-sedf OPTIONSTABLE B.30: **xm** sched-sedf REMOVED OPTIONS

<u>sched-sedf</u> Removed Options	
Options	Task
<u>-p PERIOD</u> , <u>--period= PERIOD</u>	The normal <i>EDF</i> scheduling usage in milliseconds
<u>-s SLICE</u> , <u>--slice= SLICE</u>	The normal <i>EDF</i> scheduling usage in milliseconds
<u>-l LATENCY</u> , <u>--latency= LATENCY</u>	Scaled period if domain is doing heavy I/O
<u>-e EXTRA</u> , <u>--extra= EXTRA</u>	Flag for allowing domain to run in extra time (0 or 1)
<u>-w WEIGHT</u> , <u>--weight= WEIGHT</u>	Another way of setting CPU slice

TABLE B.31: **x1** sched-sedf ADDED OPTIONS

<u>sched-sedf</u> Added Options	
Options	Task
(+) <u>-c CPUPOOL</u> , <u>--cpupool= CPUPOOL</u>	Restrict output to domains in the specified cpupool
(+) <u>-d DOMAIN</u> , <u>--domain= DOMAIN</u>	Domain

B.3.5.10 **xl cpupool -***

xl cpupool-cpu-remove CPU_POOL <CPU nr> |node:<node nr>

xl cpupool-list [-c|--cpus] CPU_POOL

TABLE B.32: **xl cpupool-list** REMOVED OPTIONS

<u>cpupool-*</u> Removed Options	
Option	Task
(-) <u>-l</u> , <u>--long</u>	Output all CPU pool details in <i>SXP</i> format

xl cpupool-cpu-add CPU_POOL cpu-nr|node:node-nr

xl cpupool-create OPTIONS CONFIG_FILE [Variable=Value ...]

TABLE B.33: **xl cpupool-create** REMOVED OPTIONS

<u>cpupool-create</u> Removed Options	
Options	Task
(-) <u>-f</u> <u>FILE</u> , <u>--defconfig=</u> <u>FILE</u>	Use the given Python configuration script. The configuration script is loaded after arguments have been processed
(-) <u>-n</u> , <u>--dryrun</u>	Dry run - prints the resulting configuration in <i>SXP</i> but does not create the CPU pool
(-) <u>--help_config</u>	Print the available configuration variables (vars) for the configuration script
(-) <u>--path=</u> <u>PATH</u>	Search path for configuration scripts. The value of PATH is a colon-separated directory list
(-) <u>-F=</u> <u>FILE</u> , <u>--config=</u> <u>FILE</u>	CPU pool configuration to use (<i>SXP</i>)

B.3.5.11 **PCI and Block Devices**

xl pci-detach [-f] DOMAIN_ID <BDF>



Note: libvirt Equivalent

virsh detach-device

TABLE B.34: **xl pci-detach** ADDED OPTIONS

<u>pci-detach</u> Added Options	
Option	Task
(+) <u>-f</u>	If <u>-f</u> is specified, xl is going to forcefully remove the device even without guest's collaboration

TABLE B.35: **xm block-list** REMOVED OPTIONS

<u>block-list</u> Removed Options	
Option	Task
(-) <u>-l</u> , <u>--long</u>	List virtual block devices for a domain

TABLE B.36: OTHER OPTIONS

Option	libvirt equivalent
xl <u>block-attach</u> <u>DOMAIN</u> <disk-spec-component(s)>	<u>virsh attach-disk/attach-device</u>
xl <u>block-list</u> <u>DOMAIN_ID</u>	<u>virsh domblklist</u>

B.3.5.12 Network

TABLE B.37: NETWORK OPTIONS

Option	libvirt equivalent
xl <u>network-list</u> <u>DOMAIN(s)</u>	<u>virsh domiflist</u>
xl <u>network-detach</u> <u>DOMAIN_ID</u> devid mac	<u>virsh detach-interface</u>

Option	<u>libvirt</u> equivalent
<u>xl network-attach</u> <u>DOMAIN(s)</u>	<u>virsh</u> <u>attach-interface/attach-device</u>

TABLE B.38: **xl** `network-attach` REMOVED OPTIONS

Removed Options	
Option	Task
(-) <u>-l</u> , <u>--long</u>	

B.3.6 New Options

TABLE B.39: NEW OPTIONS

Options	Task
<u>config-update</u> <u>DOMAIN</u> <u>CONFIG_FILE</u> <u>OPTIONS</u> <u>VARs</u>	Update the saved configuration for a running domain. This has no immediate effect but will be applied when the guest is next restarted. This command is useful to ensure that runtime modifications made to the guest will be preserved when the guest is restarted
<u>migrate-receive</u>	
<u>sharing</u> <u>DOMAIN</u>	List count of shared pages. List specifically for that domain. Otherwise, list for all domains
<u>vm-list</u>	Prints information about guests. This list excludes information about service or auxiliary domains such as Dom0 and stubdoms
<u>cpupool-rename</u> <u>CPU_POOL</u> <u>NEWNAME</u>	Renames a cpu-pool to newname
<u>cpupool- numa-split</u>	Splits up the machine into one cpu-pool per numa node

Options	Task
<code>cd-insert</code> <u>DOMAIN</u> <VirtualDevice> <type:path>	Insert a CD-ROM into a guest domain's existing virtual CD drive. The virtual drive must already exist but can be current empty
<code>cd-eject</code> <u>DOMAIN</u> <VirtualDevice>	Eject a CD-ROM from a guest's virtual CD drive. Only works with HVM domains
<code>pci-assignable-list</code>	List all the assignable PCI devices. These are devices in the system which are configured to be available for pass-through and are bound to a suitable PCI back-end driver in Dom0 rather than a real driver
<code>pci-assignable-add</code> <BDF>	Make the device at PCI Bus/Device/Function <i>BDF</i> assignable to guests. This will bind the device to the pciback driver
<code>pci-assignable-remove</code> <u>OPTIONS</u> <BDF>	Make the device at PCI Bus/Device/Function <i>BDF</i> assignable to guests. This will at least unbind the device from pciback
<code>loadpolicy</code> <u>POLICY_FILE</u>	Load <i>FLASK</i> policy from the given policy file. The initial policy is provided to the hypervisor as a multiboot module; this command allows runtime updates to the policy. Loading new security policy will reset runtime changes to device labels

B.4 External links

For more information on Xen tool stacks refer to the following online resources:

XL in Xen

[XL in Xen 4.2 \(http://wiki.xenproject.org/wiki/XL_in_Xen_4.2\)](http://wiki.xenproject.org/wiki/XL_in_Xen_4.2) ↗

xl command

XL (<http://xenbits.xen.org/docs/unstable/man/xl.1.html>) ↗ **command line**.

xl.cfg

xl.cfg (<http://xenbits.xen.org/docs/unstable/man/xl.cfg.5.html>) ↗ **domain configuration file syntax**.

xl disk

xl disk (<https://xenbits.xen.org/docs/4.3-testing/misc/xl-disk-configuration.txt>) ↗ **configuration option**.

XL vs Xend

XL vs Xend (http://wiki.xenproject.org/wiki/XL_vs_Xend_Feature_Comparison) ↗ **feature comparison**.

BDF doc

BDF documentation (http://wiki.xen.org/wiki/Bus:Device.Function_%28BDF%29_Notation) ↗.

libvirt

virsh (<http://libvirt.org/virshcmdref.html>) ↗ **command**.

B.5 Saving a Xen Guest Configuration in an **xm** Compatible Format

Although **xl** is now the current toolkit for managing Xen guests (apart from the preferred **libvirt**), you may need to export the guest configuration to the previously used **xm** format. To do this, follow these steps:

1. First export the guest configuration to a file:

```
tux > virsh dumpxml guest_id > guest_cfg.xml
```

2. Then convert the configuration to the **xm** format:

```
tux > virsh domxml-to-native xen-xm guest_cfg.xml > guest_xm_cfg
```

C GNU Licenses

This appendix contains the GNU Free Documentation License version 1.2.

GNU Free Documentation License

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary

formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute
and/or modify this document
under the terms of the GNU Free
Documentation License, Version 1.2
or any later version published by the Free
Software Foundation;
with no Invariant Sections, no Front-Cover
Texts, and no Back-Cover Texts.
A copy of the license is included in the
section entitled "GNU
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being LIST
THEIR TITLES, with the
Front-Cover Texts being LIST, and with the
Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.



AutoYaST Guide

openSUSE Leap 15.2



AutoYaST Guide

openSUSE Leap 15.2

AutoYaST is a system for unattended mass deployment of openSUSE Leap systems. AutoYaST installations are performed using an AutoYaST control file (also called a “profile”) with your customized installation and configuration data.

Publication Date: July 06, 2020

SUSE LLC

1800 South Novell Place

Provo, UT 84606

USA

<https://documentation.suse.com> ↗

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

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

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

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

Contents

1	Introduction to AutoYaST 1
1.1	Motivation 1
1.2	Overview and Concept 1
I UNDERSTANDING AND CREATING THE AUTOYAST CONTROL FILE 4	
2	The AutoYaST Control File 5
2.1	Introduction 5
2.2	Format 5
2.3	Structure 6
	Resources and Properties 7 • Nested Resources 7 • Attributes 8
3	Creating an AutoYaST Control File 9
3.1	Collecting Information 9
3.2	Using the Configuration Management System (CMS) 9
	Creating a New Control File 10
3.3	Creating/Editing a Control File Manually 11
3.4	Creating a Control File via Script with XSLT 12
II AUTOYAST CONFIGURATION EXAMPLES 14	
4	Configuration and Installation Options 15
4.1	General Options 15
	The Mode Section 16 • Configuring the Installation Settings Screen 20 • The Self-Update Section 20 • The Semi-Automatic Section 22 • The Signature Handling Section 22 • The Wait Section 24 • Examples for the general Section 26
4.2	Reporting 28

- 4.3 The Boot Loader 29
 - Loader Type 29 • Globals 30 • Device map 35
- 4.4 Partitioning 35
 - Automatic Partitioning 36 • Guided Partitioning 36 • Expert Partitioning 37 • Advanced Partitioning Features 53 • Logical Volume Manager (LVM) 58 • Software RAID 60 • Multipath Support 65 • bcache Configuration 67 • NFS Configuration 69
- 4.5 iSCSI Initiator Overview 69
- 4.6 Fibre Channel over Ethernet Configuration (FCoE) 70
- 4.7 Country Settings 72
- 4.8 Software 74
 - Package Selection with Patterns and Packages Sections 74 • Deploying Images 75 • Installing Additional/Customized Packages or Products 75 • Kernel Packages 82 • Removing Automatically Selected Packages 82 • Installing Recommended Packages/Patterns 83 • Installing Packages in Stage 2 83 • Installing Patterns in Stage 2 84 • Online Update in Stage 2 84
- 4.9 Upgrade 84
- 4.10 Services and Targets 86
- 4.11 Network Configuration 87
 - Interfaces 90 • Persistent Names of Network Interfaces 94 • Domain Name System 95 • Routing 96 • s390 Options 97 • Proxy 98
- 4.12 NIS Client and Server 98
- 4.13 NIS Serve 99
- 4.14 Hosts Definition 101
- 4.15 Windows Domain Membership 102
- 4.16 Samba Server 103
- 4.17 Authentication Client 104
- 4.18 NFS Client and Server 105

- 4.19 NTP Client 106
- 4.20 Mail Server Configuration 107
- 4.21 Apache HTTP Server Configuration 109
- 4.22 Squid Server 118
- 4.23 FTP Server 125
- 4.24 TFTP Server 129
- 4.25 Firstboot Workflow 130
- 4.26 Security Settings 130
 - Password Settings Options 131 • Boot Settings 131 • Login Settings 132 • New user settings (**useradd** settings) 132
- 4.27 Linux Audit Framework (LAF) 132
- 4.28 Users and Groups 134
 - Users 134 • User Defaults 139 • Groups 141 • Login Settings 142
- 4.29 Custom User Scripts 143
 - Pre-Install Scripts 143 • Post-partitioning Scripts 144 • Chroot Environment Scripts 145 • Post-Install Scripts 145 • Init Scripts 145 • Script XML Representation 147 • Script Example 150
- 4.30 System Variables (Sysconfig) 152
- 4.31 Adding Complete Configurations 153
- 4.32 Ask the User for Values during Installation 154
 - Default Value Scripts 163 • Scripts 164
- 4.33 Kernel Dumps 169
 - Memory Reservation 170 • Dump Saving 172 • E-Mail Notification 174 • Kdump Kernel Settings 176 • Expert Settings 177
- 4.34 DNS Server 178
- 4.35 DHCP Server 180

- 4.36 Firewall Configuration 184
 - General Firewall Configuration 184 • Firewall Zones Configuration 185 • A Full Example 186
- 4.37 Miscellaneous Hardware and System Components 187
 - Printer 187 • Sound devices 189
- 4.38 Importing SSH Keys and Configuration 189
- 4.39 Configuration Management 190
 - Connecting to a Configuration Management Server 191 • Running in Stand-alone Mode 193 • SUSE Manager Salt Formulas Support 194

III MANAGING MASS INSTALLATIONS WITH RULES AND CLASSES 195

5 Rules and Classes 196

- 5.1 Rule-based Automatic Installation 196
 - Rules File Explained 197 • Custom Rules 200 • Match Types for Rules 200 • Combine Attributes 201 • Rules File Structure 201 • Predefined System Attributes 202 • Rules with Dialogs 204
- 5.2 Classes 207
- 5.3 Mixing Rules and Classes 209
- 5.4 Merging of Rules and Classes 209

IV UNDERSTANDING THE AUTO-INSTALLATION PROCESS 212

6 The Auto-Installation Process 213

- 6.1 Introduction 213
 - X11 Interface (graphical) 213 • Serial Console 213 • Text-based YaST Installation 213
- 6.2 Choosing the Right Boot Medium 214
 - Booting from a Flash Disk (for example, USB stick) 214 • Booting from DVD-ROM 214 • Booting via PXE over the Network 214

6.3	Invoking the Auto-Installation Process	215
	Command Line Options	216
	Auto-installing a Single System	222
	Combining the linuxrc info File with the AutoYaST Control File	222
6.4	System Configuration	223
	Post-Install and System Configuration	223
	System Customization	223
V	USES FOR AUTOYAST ON INSTALLED SYSTEMS	224
7	Running AutoYaST in an Installed System	225
VI	APPENDICES	227
A	Handling Rules	228
B	AutoYaST FAQ - Frequently Asked Questions	229
C	Advanced linuxrc Options	233
C.1	Passing Parameters to linuxrc	233
C.2	info File Format	234
C.3	Advanced Network Setup	236
D	Main Differences Between openSUSE Leap 42.3 and 15 Profiles	238
D.1	Partitioning	238
	GPT Becomes the Default Partition Type on AMD64/Intel 64	238
	Setting Partition Numbers	238
	Forcing Primary Partitions	239
	Btrfs: Default Subvolume Name	239
	Btrfs: Disabling Subvolumes	239
	Reading an Existing /etc/fstab Is No Longer Supported	240
	Setting for Aligning Partitions Has Been Dropped	240
D.2	Firewall Configuration	240
	Assigning Interfaces to Zones	242
	Opening Ports	244
	Opening firewalld Services	245
	For More Information	246

- D.3 NTP Configuration 246
- D.4 AutoYaST Packages Are Needed for the Second Stage 247
- D.5 The CA Management Module Has Been Dropped 247
- D.6 Upgrade 248
 - Software 248

1 Introduction to AutoYaST

1.1 Motivation

Standard installations of openSUSE Leap are based on a wizard workflow. This is user-friendly and efficient when installing on few machines. However, it becomes repetitive and time-consuming when installing on many machines.

To avoid this, you could do mass deployments by copying the hard disk of the first successful installation. Unfortunately, that leads to the issue that even minute configuration changes between each machine need to later be dealt with individually. For example, when using static IP addresses, these IP addresses would need to be reset for each machine.

A regular installation of openSUSE Leap is semi-automated by default. The user is prompted to select the necessary information at the beginning of the installation (usually language only). YaST then generates a proposal for the underlying system depending on different factors and system parameters. Usually—and especially for new systems—such a proposal can be used to install the system and provides a usable installation. The steps following the proposal are fully automated.

AutoYaST can be used where no user intervention is required or where customization is required. Using an AutoYaST control file, YaST prepares the system for a custom installation and does not interact with the user, unless specified in the file controlling the installation.

AutoYaST is not an automated GUI system. This means that usually many screens will be skipped—you will never see the language selection interface, for example. AutoYaST will simply pass the language parameter to the sub-system without displaying any language related interface.

1.2 Overview and Concept

Using AutoYaST, multiple systems can easily be installed in parallel and quickly. They need to share the same environment and similar, but not necessarily identical, hardware. The installation is defined by an XML configuration file (usually named `autoinst.xml`) called the “AutoYaST control file”. It can initially be created using existing configuration resources easily be tailored for any specific environment.

AutoYaST is fully integrated and provides various options for installing and configuring a system. The main advantage over other auto-installation systems is the possibility to configure a computer by using existing modules and avoiding using custom scripts which are normally executed at the end of the installation.

This document will guide you through the three steps of auto-installation:

- **Preparation:** All relevant information about the target system is collected and turned into the appropriate directives of the control file. The control file is transferred onto the target system where its directives will be parsed and fed into YaST.
- **Installation:** YaST performs the installation of the basic system using the data from the AutoYaST control file.
- **Configuration:** After the installation of the basic system, the system configuration is performed in the second stage of the installation. User-defined post-installation scripts from the AutoYaST control file will also be executed at this stage.



Note: Second Stage

A regular installation of openSUSE Leap 15.2 is performed in a single stage. The auto-installation process, however, is divided into two stages. After the installation of the basic system the system boots into the second stage where the system configuration is done.

The packages `autoyast2` and `autoyast2-installation` need to be installed to run the second stage in the installed system correctly. Otherwise an error will be shown before booting into the installed system.

The second stage can be turned off with the `second_stage` parameter:

```
<general>
  <mode>
    <confirm config:type="boolean">>false</confirm>
    <second_stage config:type="boolean">>false</second_stage>
  </mode>
</general>
```

The complete and detailed process is illustrated in the following figure:

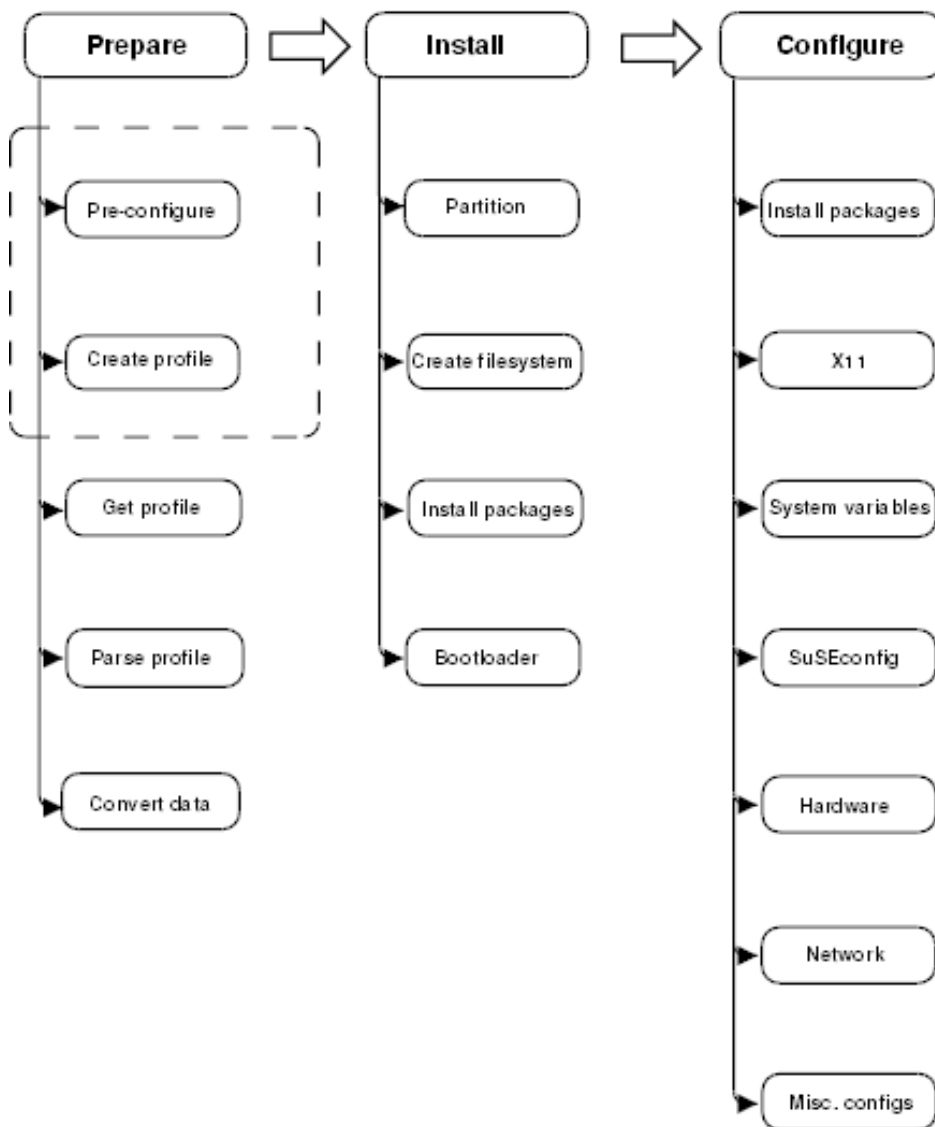


FIGURE 1.1: AUTO-INSTALLATION PROCESS

I Understanding and Creating the AutoYaST Control File

- 2 The AutoYaST Control File 5
- 3 Creating an AutoYaST Control File 9

2 The AutoYaST Control File

2.1 Introduction

The control file is a configuration description for a single system. It consists of sets of resources with properties including support for complex structures such as lists, records, trees and large embedded or referenced objects.

Important: Control Files from OS Releases Older Than SLES 12 GA and openSUSE 42.0 Are Incompatible

Many major changes were introduced with SLES 12 and openSUSE Leap 42.0, such as the switch to systemd and GRUB 2. These changes also required fundamental changes in AutoYaST. Therefore you cannot use AutoYaST control files created on SLES 11 to install openSUSE Leap 15.2 and vice versa.

2.2 Format

The XML configuration format provides a consistent file structure, which is easy to learn and to remember when attempting to configure a new system.

The AutoYaST control file uses XML to describe the system installation and configuration. XML is a commonly used markup, and many users are familiar with the concepts of the language and the tools used to process XML files. If you edit an existing control file or create a control file using an editor from scratch, it is strongly recommended to validate the control file. This can be done using a validating XML parser such as `xmllint` or `jing`, for example (see [Section 3.3, "Creating/Editing a Control File Manually"](#)).

The following example shows a control file in XML format:

EXAMPLE 2.1: AUTOYAST CONTROL FILE (PROFILE)

```
<?xml version="1.0"?>
<!DOCTYPE profile>
<profile
  xmlns="http://www.suse.com/1.0/yast2ns"
  xmlns:config="http://www.suse.com/1.0/configns">
  <partitioning config:type="list">
```

```

<drive>
  <device>/dev/sda</device>
  <partitions config:type="list">
    <partition>
      <filesystem config:type="symbol">btrfs</filesystem>
      <size>10G</size>
      <mount>/</mount>
    </partition>
    <partition>
      <filesystem config:type="symbol">xfs</filesystem>
      <size>120G</size>
      <mount>/data</mount>
    </partition>
  </partitions>
</drive>
</partitioning>
<scripts>
  <pre-scripts>
    <script>
      <interpreter>shell</interpreter>
      <filename>start.sh</filename>
      <source>
        <![CDATA[
#!/bin/sh
echo "Starting installation"
exit 0

]]>

      </source>
    </script>
  </pre-scripts>
</scripts>
</profile>

```

2.3 Structure

Below is an example of a basic control file container, the actual content of which is explained later on in this chapter.

EXAMPLE 2.2: CONTROL FILE CONTAINER

```

<?xml version="1.0"?>
<!DOCTYPE profile>
<profile

```

```
xmlns="http://www.suse.com/1.0/yast2ns"
xmlns:config="http://www.suse.com/1.0/configns">
<!-- RESOURCES -->
</profile>
```

The `<profile>` element (root node) contains one or more distinct resource elements. The permissible resource elements are specified in the schema files

2.3.1 Resources and Properties

A resource element either contains multiple and distinct property and resource elements, or multiple instances of the same resource element, or it is empty. The permissible content of a resource element is specified in the schema files.

A property element is either empty or contains a literal value. The permissible property elements and values in each resource element are specified in the schema files

An element can be either a container of other elements (a resource) or it has a literal value (a property); it can never be both. This restriction is specified in the schema files. A configuration component with more than one value must either be represented as an embedded list in a property value or as a nested resource.

An empty element, such as `<foo></foo>` or `<bar/>`, will *not* be present in the parsed data model. Usually this is interpreted as wanting a sensible default value. In cases where you need an explicitly empty string instead, use a CDATA section: `<foo><![CDATA[]]></foo>`.

2.3.2 Nested Resources

Nested resource elements allow a tree-like structure of configuration components to be built to any level.

There are two kinds of nested resources: maps and lists. Maps, also known as associative arrays, hashes, or dictionaries, contain mixed contents, identified by their tag names. Lists, or arrays, have all items of the same type.

EXAMPLE 2.3: NESTED RESOURCES

```
...
<drive>
  <device>/dev/sda</device>
  <partitions config:type="list">
    <partition>
```

```
    <size>10G</size>
    <mount>/</mount>
  </partition>
  <partition>
    <size>1G</size>
    <mount>/tmp</mount>
  </partition>
</partitions>
</drive>
....
```

In the example above, the drive resource is a map consisting of a device property and a partitions resource. The partitions resource is a list containing multiple instances of the partition resource. Each partition resource is a map containing a size and mount property. The default type of a nested resource is map. Lists must be marked as such using the config:type="list" attribute.

2.3.3 Attributes

Global attributes are used to define metadata on resources and properties. Attributes are used to define context switching. They are also used for naming and typing properties as shown in the previous sections. Attributes are in a separate namespace so they do not need to be treated as reserved words in the default namespace.

The config:type attribute determines the type of the resource or property in the parsed data model. For resources, lists need a list type whereas a map is the default type that does not need an attribute. For properties, boolean, symbol, and integer can be used, the default being a string.

Attributes are not optional. It may appear that attributes are optional, because various parts of the schema are not very consistent in their usage of data types. In some places an enumeration is represented by a symbol, elsewhere a string is required. One resource needs config:type="integer", another will parse the number from a string property. Some resources use config:type="boolean", others want yes or even 1. If in doubt, consult the schema file.

3 Creating an AutoYaST Control File

3.1 Collecting Information

To create the control file, you need to collect information about the systems you are going to install. This includes hardware data and network information among other things. Make sure you have the following information about the machines you want to install:

- Hard disk types and sizes
- Graphical interface and attached monitor, if any
- Network interface and MAC address if known (for example, when using DHCP)

Also verify that both autoyast2-installation and autoyast2 are installed.

3.2 Using the Configuration Management System (CMS)

To create the control file for one or more computers, a configuration interface based on YaST is provided. This system depends on existing modules which are usually used to configure a computer in regular operation mode, for example, after openSUSE Leap is installed.

The configuration management system lets you easily create control files and manage a repository of configurations for use in a networked environment with multiple clients.

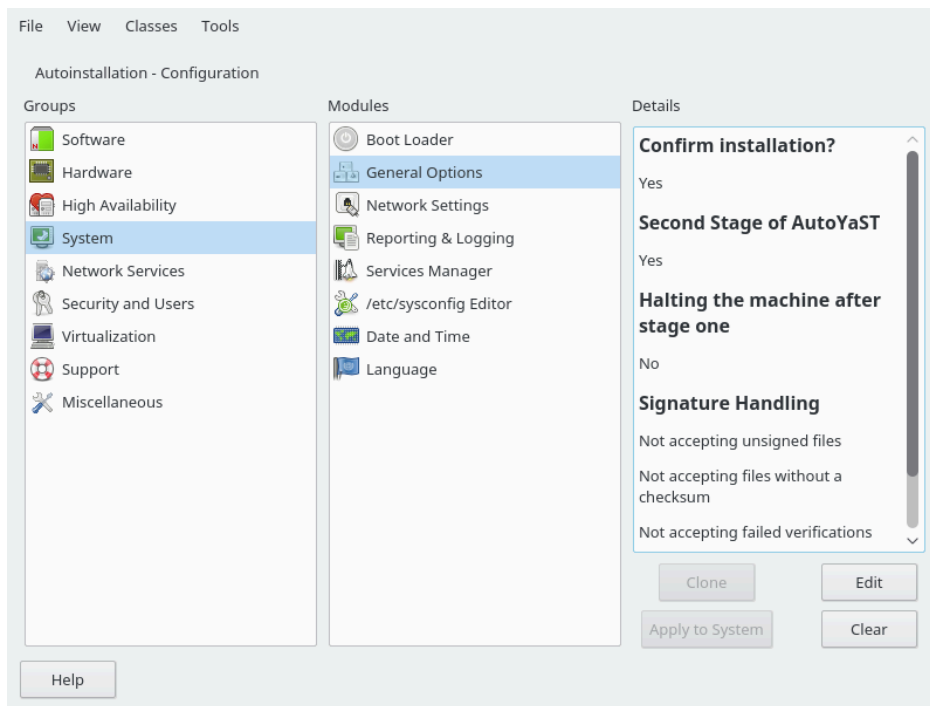


FIGURE 3.1: CONFIGURATION SYSTEM

3.2.1 Creating a New Control File

The easiest way to create an AutoYaST profile is to use an existing openSUSE Leap system as a template. On an already installed system, start *YaST* > *Miscellaneous* > *Autoinstallation*. Now select *Tools* > *Create Reference Profile* from the menu. Choose the system components you want to include in the profile. Alternatively, create a profile containing the complete system configuration by running `sudo yast clone_system` from the command line.

Both methods will create the file `/root/autoinst.xml`. The version created on the command line can be used to set up an identical clone of the system on which the profile was created. However, usually you will want to adjust the file to make it possible to install several machines that are very similar, but not identical. This can be done by adjusting the profile using your favorite text/XML editor.

With some exceptions, almost all resources of the control file can be configured using the configuration management system. The system offers flexibility and the configuration of some resources is identical to the one available in the YaST control center. In addition to the existing and familiar modules new interfaces were created for special and complex configurations, for example for partitioning, general options and software.

Furthermore, using a CMS guarantees the validity of the resulting control file and its direct use for starting automated installation.

Make sure the configuration system is installed (package `autoyast2`) and call it using the YaST control center or as root with the following command (make sure the `DISPLAY` variable is set correctly to start the graphical user interface instead of the text-based one):

```
/sbin/yast2 autoyast
```

3.3 Creating/Editing a Control File Manually

If editing the control file manually, make sure it has a valid syntax. To check the syntax, use the tools already available on the distribution. For example, to verify that the file is well-formed (has a valid XML structure), use the utility `xmllint` available with the `libxml2` package:

```
xmllint <control file>
```

If the control file is not well formed, for example, if a tag is not closed, `xmllint` will report the errors.

To validate the control file, use the tool `jing` from the package with the same name. During validation, misplaced or missing tags and attributes and wrong attribute values are detected.

```
jing /usr/share/YaST2/schema/autoyast/rng/profile.rng <control file>
```

`/usr/share/YaST2/schema/autoyast/rng/profile.rng` is provided by the package `yast2-schema`. This file describes the syntax and classes of an AutoYaST profile.

Before going on with the autoinstallation, fix any errors resulting from such checks. The autoinstallation process cannot be started with an invalid and not well-formed control file.

You can use any XML editor available on your system or any text editor with XML support (for example, Emacs, Vim). However, it is not optimal to create the control file manually for many machines and it should only be seen as an interface between the autoinstallation engine and the Configuration Management System (CMS).



Tip: Using Emacs as an XML Editor

The built-in `nxml-mode` turns Emacs into a fully-fledged XML editor with automatic tag completion and validation. Refer to the Emacs help for instructions on how to set up `nxml-mode`.

3.4 Creating a Control File via Script with XSLT

If you have a template and want to change a few things via script or command line, use an XSLT processor like `xsltproc`. For example, if you have an AutoYaST control file and want to fill out the host name via script for any reason. (If doing this often, you should consider scripting it.)

First, create an XSL file:

EXAMPLE 3.1: EXAMPLE FILE FOR REPLACING THE HOST NAME/DOMAIN BY SCRIPT

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:y2="http://www.suse.com/1.0/yast2ns"
  xmlns:config="http://www.suse.com/1.0/configns"
  xmlns="http://www.suse.com/1.0/yast2ns"
  version="1.0">
  <xsl:output method="xml" encoding="UTF-8" indent="yes" omit-xml-declaration="no" cdata-
section-elements="source"/>

  <!-- the parameter names -->
  <xsl:param name="hostname"/>
  <xsl:param name="domain"/>

  <xsl:template match="/">
    <xsl:apply-templates select="@*|node()"/>
  </xsl:template>

  <xsl:template match="y2:dns">
    <xsl:copy>
      <!-- where to copy the parameters -->
      <domain><xsl:value-of select="string($domain)"/></domain>
      <hostname><xsl:value-of select="string($hostname)"/></hostname>
      <xsl:apply-templates select="@*|node()"/>
    </xsl:copy>
  </xsl:template>

  <xsl:template match="@*|node()" >
    <xsl:copy>
      <xsl:apply-templates select="@*|node()"/>
    </xsl:copy>
  </xsl:template>

</xsl:stylesheet>
```

This file expects the host name and the domain name as parameters from the user.

```
<xsl:param name="hostname"/>
<xsl:param name="domain"/>
```

There will be a copy of those parameters in the DNS section of the control file. This means that if there already is a domain element in the DNS section, you will get a second one, which will cause conflicts.

For more information about XSLT, go to the official Web page www.w3.org/TR/xslt (<http://www.w3.org/TR/xslt>) ↗

II AutoYaST Configuration Examples

4 Configuration and Installation Options **15**

4 Configuration and Installation Options

This section contains configuration examples for services, registration, user and group management, upgrades, partitioning, configuration management, SSH key management, firewall configuration, and other installation options.

This chapter introduces important parts of a control file for standard purposes. To learn about other available options, use the configuration management system.

Note that for some configuration options to work, additional packages need to be installed, depending on the software selection you have configured. If you choose to install a minimal system then some packages might be missing and need to be added to the individual package selection.

YaST will install packages required in the second phase of the installation and before the post-installation phase of AutoYaST has started. However, if necessary YaST modules are not available in the system, important configuration steps will be skipped. For example, no security settings will be configured if `yast2-security` is not installed.

4.1 General Options

The general section includes all settings that influence the installation workflow. The overall structure of this section looks like the following:

```
<?xml version="1.0"?>
<!DOCTYPE profile>
<profile xmlns="http://www.suse.com/1.0/yast2ns"
  xmlns:config="http://www.suse.com/1.0/configns">
  <general>
    <ask-list>①
      ...
    </ask-list>
    <cio_ignore>
      ...
    </cio_ignore>
    <mode>②
      ...
    </mode>
    <proposals>③
      ...
    </proposals>
```

```

<self_update> ④
  ...
</self_update>
<self_update_url>
  ...
</self_update_url>
<semi-automatic config:type="list"> ⑤
  ...
</semi-automatic>
<signature-handling> ⑥
  ...
</signature-handling>
<storage> ⑦
  ...
</storage>
<wait> ⑧
  ...
</wait>
</general>
</profile>

```

- ① *Section 4.32, "Ask the User for Values during Installation"*
- ② *Section 4.1.1, "The Mode Section"*
- ③ *Section 4.1.2, "Configuring the Installation Settings Screen"*
- ④ *Section 4.1.3, "The Self-Update Section"*
- ⑤ *Section 4.1.4, "The Semi-Automatic Section"*
- ⑥ *Section 4.1.5, "The Signature Handling Section"*
- ⑦ *Section 4.4, "Partitioning"*
- ⑧ *Section 4.1.6, "The Wait Section"*

4.1.1 The Mode Section

The mode section configures the behavior of AutoYaST with regard to user confirmations and rebooting. The following elements are allowed in the `mode` section:

`activate_systemd_default_target`

If you set this entry to `false`, the default `systemd` target will not be activated via the call `systemctl isolate`. Setting this value is optional. The default is `true`.

```
<general>
```

```
<mode>
  <activate_systemd_default_target config:type="boolean">
    true
  </activate_systemd_default_target>
</mode>
...
</general>
```

confirm

By default, the installation stops at the *Installation Settings* screen. Up to this point, no changes have been made to the system and settings may be changed on this screen. To proceed and finally start the installation, the user needs to confirm the settings. By setting this value to false the settings are automatically accepted and the installation starts. Only set to false to carry out a fully unattended installation. Setting this value is optional. The default is true.

```
<general>
  <mode>
    <confirm config:type="boolean">true</confirm>
  </mode>
  ...
</general>
```

confirm_base_product_license

If you set this to true, the EULA of the base product will be shown. The user needs to accept this license. Otherwise the installation will be canceled. Setting this value is optional. The default is false. This setting applies to the base product license only. Use the flag confirm_license in the add-on section for additional licenses (see [Section 4.8.3, "Installing Additional/Customized Packages or Products"](#) for details).

```
<general>
  <mode>
    <confirm_base_product_license config:type="boolean">
      false
    </confirm_base_product_license>
  </mode>
  ...
</general>
```

final_halt

When set to true, the machine shuts down after everything is installed and configured at the end of the second stage. If you enable final_halt, you do not need to set the final_reboot option to true.


```
<general>
  <mode>
    <final_halt config:type="boolean">>false</final_halt>
  </mode>
  ...
</general>
```

final_reboot

When set to true, the machine reboots after everything is installed and configured at the end of the second stage. If you enable final_reboot, you do not need to set the final_halt option to true.

```
<general>
  <mode>
    <final_reboot config:type="boolean">>true</final_reboot>
  </mode>
  ...
</general>
```

final_restart_services

If you set this entry to false, services will *not* be restarted at the end of the installation (when everything is installed and configured at the end of the second stage). Setting this value is optional. The default is true.

```
<general>
  <mode>
    <final_restart_services config:type="boolean">
      true
    </final_restart_services>
  </mode>
  ...
</general>
```

forceboot

Some openSUSE releases use Kexec to avoid the reboot after the first stage. They immediately boot into the installed system. You can force a reboot by setting this to true. Setting this value is optional. The default is set by the product.

```
<general>
  <mode>
    <forceboot config:type="boolean">>false</forceboot>
  </mode>
  ...
```

```
</general>
```

! Important: Drivers May Need a Reboot

Some drivers, for example the proprietary drivers for Nvidia and ATI graphics cards, need a reboot and will not work properly when using Kexec. Therefore the default on openSUSE Leap products is to always do a proper reboot.

halt

Shuts down the machine after the first stage. All packages and the boot loader have been installed and all your chroot scripts have run. Instead of rebooting into stage two, the machine is turned off. If you turn it on again, the machine boots and the second stage of the autoinstallation starts. Setting this value is optional. The default is false.

```
<general>
  <mode>
    <halt config:type="boolean">false</halt>
  </mode>
  ...
</general>
```

max_systemd_wait

Specifies how long AutoYaST waits (in seconds) at most for systemd to set up the default target. Setting this value is optional and should not normally be required. The default is 30 (seconds).

```
<general>
  <mode>
    <max_systemd_wait config:type="integer">30</max_systemd_wait>
  </mode>
  ...
</general>
```

ntp_sync_time_before_installation

Specify the NTP server with which to synchronize time before starting the installation. Time synchronization will only occur if this option is set. Keep in mind that you need a network connection and access to a time server. Setting this value is optional. By default no time synchronization will occur.

```
<general>
  <mode>
```

```

    <ntp_sync_time_before_installation>
      &ntpname;
    </max_systemd_wait>
  </mode>
  ...
</general>

```

second_stage

A regular installation of openSUSE Leap is performed in a single stage. The auto-installation process, however, is divided into two stages. After the installation of the basic system the system boots into the second stage where the system configuration is done. Set this option to false to disable the second stage. Setting this value is optional. The default is true.

```

<general>
  <mode>
    <second_stage config:type="boolean">true</second_stage>
  </mode>
  ...
</general>

```

4.1.2 Configuring the Installation Settings Screen

AutoYaST allows you to configure the *Installation Settings* screen, which shows a summary of the installation settings. On this screen, the user can change the settings before confirming them to start the installation. Using the proposal tag, you can control which settings (“proposals”) are shown in the installation screen. A list of valid proposals for your products is available from the /control.xml file on the installation medium. This setting is optional. By default all configuration options will be shown.

```

<proposals config:type="list">
  <proposal>partitions_proposal</proposal>
  <proposal>timezone_proposal</proposal>
  <proposal>software_proposal</proposal>
</proposals>

```

4.1.3 The Self-Update Section

During the installation, YaST can update itself to solve bugs in the installer that were discovered after the release. Refer to the *Deployment Guide* for further information about this feature.

! Important: Quarterly Media Update: Self-Update Disabled

The installer self-update is only available if you use the GM images of the Unified Installer and Packages ISOs. If you install from the ISOs published as quarterly update (they can be identified by the string QU in the name), the installer cannot update itself, because this feature has been disabled in the update media.

Use the following tags to configure the YaST self-update:

self_update

If set to true or false, this option enables or disables the YaST self-update feature. Setting this value is optional. The default is true.

```
<general>
  <self_update config:type="boolean">true</self_update>
  ...
</general>
```

Alternatively, you can specify the boot parameter self_update=1 on the kernel command line.

self_update_url

Location of the update repository to use during the YaST self-update. For more information, refer to the *Deployment Guide*.

! Important: Installer Self-Update Repository Only

The self_update_url parameter expects only the installer self-update repository URL. Do not supply any other repository URL—for example the URL of the software update repository.

```
<general>
  <self_update_url>
    http://example.com/updates/$arch
  </self_update_url>
  ...
</general>
```

The URL may contain the variable \$arch. It will be replaced by the system's architecture, such as x86_64, s390x, etc.

Alternatively, you can specify the boot parameter `self_update=1` together with `self_update=URL` on the kernel command line.

4.1.4 The Semi-Automatic Section

AutoYaST offers to start some YaST modules during the installation. This is useful to give the administrators installing the machine the possibility to manually configure some aspects of the installation while at the same time automating the rest of the installation. Within the semi-automatic section, you can start the following YaST modules:

- The network settings module (`networking`)
- The partitioner (`partitioning`)
- The registration module (`scc`)

The following example starts all three supported YaST modules during the installation:

```
<general>
<semi-automatic config:type="list">
  <semi-automatic_entry>networking</semi-automatic_entry>
  <semi-automatic_entry>scc</semi-automatic_entry>
  <semi-automatic_entry>partitioning</semi-automatic_entry>
</semi-automatic>
</general>
```

4.1.5 The Signature Handling Section

By default AutoYaST will only install signed packages from sources with known GPG keys. Use this section to overwrite the default settings.



Warning: Overwriting the Signature Handling Defaults

Installing unsigned packages, packages with failing checksum checks, or packages from sources you do not trust is a major security risk. Packages may have been modified and may install malicious software on your machine. Only overwrite the defaults in this section if you are sure the repository and packages can be trusted. SUSE is not responsible for any problems arising from software installed with integrity checks disabled.

Default values for all options are false. If an option is set to false and a package or repository fails the respective test, it is silently ignored and will not be installed.

accept_unsigned_file

If set to true, AutoYaST will accept unsigned files like the content file.

```
<general>
  <signature-handling>
    <accept_unsigned_file config:type="boolean">
      false
    </accept_unsigned_file>
  </signature-handling>
  ...
</general>
```

accept_file_without_checksum

If set to true, AutoYaST will accept files without a checksum in the content file.

```
<general>
  <signature-handling>
    <accept_file_without_checksum config:type="boolean">
      false
    </accept_file_without_checksum>
  </signature-handling>
  ...
</general>
```

accept_verification_failed

If set to true, AutoYaST will accept signed files even when the signature verification fails.

```
<general>
  <signature-handling>
    <accept_verification_failed config:type="boolean">
      false
    </accept_verification_failed>
  </signature-handling>
  ...
</general>
```

accept_unknown_gpg_key

If set to true, AutoYaST will accept new GPG keys of the installation sources, for example the key used to sign the content file.

```
<general>
  <signature-handling>
```

```
<accept_unknown_gpg_key config:type="boolean">
  false
</accept_unknown_gpg_key>
</signature-handling>
...
<general>
```

accept_non_trusted_gpg_key

Set this option to true to accept known keys you have not yet trusted.

```
<general>
<signature-handling>
  <accept_non_trusted_gpg_key config:type="boolean">
    false
  </accept_non_trusted_gpg_key>
</signature-handling>
...
<general>
```

import_gpg_key

If set to true, AutoYaST will accept and import new GPG keys on the installation source in its database.

```
<general>
<signature-handling>
  <import_gpg_key config:type="boolean">
    false
  </import_gpg_key>
</signature-handling>
...
<general>
```

4.1.6 The Wait Section

In the second stage of the installation the system is configured by running modules, for example the network configuration. Within the wait section you can define scripts that will get executed before and after a specific module has run. You can also configure a span of time in which the system is inactive (“sleeps”) before and after each module.

pre-modules

Defines scripts and sleep time executed before a configuration module starts. The following code shows an example setting the sleep time to ten seconds and executing an echo command before running the network configuration module.

```

<general>
  <wait>
    <pre-modules config:type="list">
      <module>
        <name>networking</name>
        <sleep>
          <time config:type="integer">10</time>
          <feedback config:type="boolean">true</feedback>
        </sleep>
        <script>
          <source>echo foo</source>
          <debug config:type="boolean">>false</debug>
        </script>
      </module>
    </pre-modules>
    ...
  </wait>
</general>

```

post-modules

Defines scripts and sleep time executed after a configuration module starts. The following code shows an example setting the sleep time to ten seconds and executing an echo command after running the network configuration module.

```

<general>
  <wait>
    <post-modules config:type="list">
      <module>
        <name>networking</name>
        <sleep>
          <time config:type="integer">10</time>
          <feedback config:type="boolean">true</feedback>
        </sleep>
        <script>
          <source>echo foo</source>
          <debug config:type="boolean">>false</debug>
        </script>
      </module>
    </post-modules>
    ...
  </wait>
</general>

```


4.1.7 Examples for the general Section

Find examples covering several use cases in this section.

EXAMPLE 4.1: GENERAL OPTIONS

This example shows the most commonly used options in the general section. The scripts in the pre- and post-modules sections are only dummy scripts illustrating the concept.

```
<?xml version="1.0"?>
<!DOCTYPE profile>
<profile xmlns="http://www.suse.com/1.0/yast2ns"
  xmlns:config="http://www.suse.com/1.0/configs">
  <general>
    <!-- Use cio_ignore on &zseries; only -->
    <cio_ignore config:type="boolean">false</cio_ignore>
    <mode>
      <halt config:type="boolean">false</halt>
      <forceboot config:type="boolean">false</forceboot>
      <final_reboot config:type="boolean">false</final_reboot>
      <final_halt config:type="boolean">false</final_halt>
      <confirm_base_product_license config:type="boolean">
        false
      </confirm_base_product_license>
      <confirm config:type="boolean">true</confirm>
      <second_stage config:type="boolean">true</second_stage>
    </mode>
    <proposals config:type="list">
      <proposal>partitions_proposal</proposal>
    </proposals>
    <self_update config:type="boolean">true</self_update>
    <self_update_url>http://example.com/updates/$arch</self_update_url>
    <signature-handling>
      <accept_unsigned_file config:type="boolean">
        true
      </accept_unsigned_file>
      <accept_file_without_checksum config:type="boolean">
        true
      </accept_file_without_checksum>
      <accept_verification_failed config:type="boolean">
        true
      </accept_verification_failed>
      <accept_unknown_gpg_key config:type="boolean">
        true
      </accept_unknown_gpg_key>
      <import_gpg_key config:type="boolean">true</import_gpg_key>
      <accept_non_trusted_gpg_key config:type="boolean">
        true
      </accept_non_trusted_gpg_key>
    </signature-handling>
  </general>
</profile>
```

```

    </accept_non_trusted_gpg_key>
</signature-handling>
<storage>
  <partition_alignment config:type="symbol">
    align_cylinder
  </partition_alignment>
</storage>
<wait>
  <pre-modules config:type="list">
    <module>
      <name>networking</name>
      <sleep>
        <time config:type="integer">10</time>
        <feedback config:type="boolean">true</feedback>
      </sleep>
      <script>
        <source>&gt;![CDATA[
echo "Sleeping 10 seconds"
  ]&gt;</source>
        <debug config:type="boolean">>false</debug>
      </script>
    </module>
  </pre-modules>
  <post-modules config:type="list">
    <module>
      <name>networking</name>
      <sleep>
        <time config:type="integer">10</time>
        <feedback config:type="boolean">true</feedback>
      </sleep>
      <script>
        <source>&gt;![CDATA[
echo "Sleeping 10 seconds"
  ]&gt;</source>
        <debug config:type="boolean">>false</debug>
      </script>
    </module>
  </post-modules>
</wait>
</general>
</profile>

```

4.2 Reporting

The `report` resource manages three types of pop-ups that may appear during installation:

- message pop-ups (usually non-critical, informative messages),
- warning pop-ups (if something might go wrong),
- error pop-ups (in case an error occurs).

EXAMPLE 4.2: REPORTING BEHAVIOR

```
<report>
  <errors>
    <show config:type="boolean">true</show>
    <timeout config:type="integer">0</timeout>
    <log config:type="boolean">true</log>
  </errors>
  <warnings>
    <show config:type="boolean">true</show>
    <timeout config:type="integer">10</timeout>
    <log config:type="boolean">true</log>
  </warnings>
  <messages>
    <show config:type="boolean">true</show>
    <timeout config:type="integer">10</timeout>
    <log config:type="boolean">true</log>
  </messages>
  <yesno_messages>
    <show config:type="boolean">true</show>
    <timeout config:type="integer">10</timeout>
    <log config:type="boolean">true</log>
  </yesno_messages>
</report>
```

Depending on your experience, you can skip, log and show (with timeout) those messages. It is recommended to show all `messages` with timeout. Warnings can be skipped in some places but should not be ignored.

The default setting in auto-installation mode is to show errors without timeout and to show all warnings/messages with a timeout of 10 seconds.



Warning: Critical System Messages

Note that not all messages during installation are controlled by the `report` resource. Some critical messages concerning package installation and partitioning will show up ignoring your settings in the `report` section. Usually those messages will need to be answered with *Yes* or *No*.

4.3 The Boot Loader

This documentation is for `yast2-bootloader` and applies to GRUB 2. For older product versions shipping with legacy GRUB, refer to the documentation that comes with your distribution in `/usr/share/doc/packages/autoyast2/`

The general structure of the AutoYaST boot loader part looks like the following:

```
<bootloader>
  <loader_type>
    <!-- boot loader type (grub2 or grub2-efi) -->
  </loader_type>
  <global>
    <!--
      entries defining the installation settings for GRUB 2 and
      the generic boot code
    -->
  </global>
  <device_map config:type="list">
    <!-- entries defining the order of devices -->
  </device_map>
</bootloader>
```

4.3.1 Loader Type

This defines which boot loader (UEFI or BIOS/legacy) to use. Not all architectures support both legacy and EFI variants of the boot loader. The safest (`default`) option is to leave the decision up to the installer.

```
<loader_type>LOADER_TYPE</loader_type>
```

Possible values for `LOADER_TYPE` are:

- `default`: The installer chooses the correct boot loader. This is the default when no option is defined.
- `grub2`: Use the legacy BIOS boot loader.
- `grub2-efi`: Use the EFI boot loader.
- `none`: The boot process is not managed and configured by the installer.

4.3.2 Globals

This is an important if optional part. Define here where to install GRUB 2 and how the boot process will work. Again, `yast2-bootloader` proposes a configuration if you do not define one. Usually the AutoYaST control file includes only this part and all other parts are added automatically during installation by `yast2-bootloader`. Unless you have some special requirements, do not specify the boot loader configuration in the XML file.

```
<global>
  <activate config:type="boolean">true</activate>
  <timeout config:type="integer">10</timeout>
  <suse_btrfs config:type="boolean">true</suse_btrfs>
  <terminal>gfxterm</terminal>
  <gfxmode>1280x1024x24</gfxmode>
</global>
```

Attribute	Description
<code>activate</code>	Set the boot flag on the boot partition. The boot partition can be <code>/</code> if there is no separate <code>/boot</code> partition. If the boot partition is on a logical partition, the boot flag is set to the extended partition. <pre><activate config:type="boolean">true</activate></pre>
<code>append</code>	Kernel parameters added at the end of boot entries for normal and recovery mode.

Attribute	Description
	<pre data-bbox="805 235 1414 286"><append>nomodeset vga=0x317</append></pre>
<u>boot_boot</u>	<p data-bbox="805 322 1414 454">Write GRUB 2 to a separate <code>/boot</code> partition. If no separate <code>/boot</code> partition exists, GRUB 2 will be written to <code>/</code>.</p> <pre data-bbox="805 488 1414 539"><boot_boot>>false</boot_boot></pre>
<u>boot_custom</u>	<p data-bbox="805 580 1414 618">Write GRUB 2 to a custom device.</p> <pre data-bbox="805 651 1414 703"><boot_custom>/dev/sda3</boot_custom></pre>
<u>boot_extended</u>	<p data-bbox="805 743 1414 1016">Write GRUB 2 to the extended partition (important if you want to use generic boot code and the <code>/boot</code> partition is logical). Note: if the boot partition is logical, you should use <u><code>boot_mbr</code></u> (write GRUB 2 to MBR) rather than <u><code>generic_mbr</code></u>.</p> <pre data-bbox="805 1050 1414 1102"><boot_extended>>false</boot_extended></pre>
<u>boot_mbr</u>	<p data-bbox="805 1144 1414 1272">Write GRUB 2 to the MBR of the first disk in the order (device.map includes order of disks).</p> <pre data-bbox="805 1305 1414 1357"><boot_mbr>>false</boot_mbr></pre>
<u>boot_root</u>	<p data-bbox="805 1402 1414 1440">Write GRUB 2 to <code>/</code> partition.</p> <pre data-bbox="805 1473 1414 1525"><boot_root>>false</boot_root></pre>
<u>generic_mbr</u>	<p data-bbox="805 1565 1414 1653">Write generic boot code to the MBR (will be ignored if <u><code>boot_mbr</code></u> is set to <u><code>true</code></u>).</p> <pre data-bbox="805 1686 1414 1760"><generic_mbr config:type="boolean">>false</generic_mbr></pre>

Attribute	Description
<u>gfxmode</u>	<p>Graphical resolution of the GRUB 2 screen (requires <code><terminal></code> to be set to <code>gfx-term</code>. Valid entries are <code>auto</code>, <code>HORIZONTALxVERTICAL</code>, or <code>HORIZONTALxVERTICALx-COLOR DEPTH</code>. You can see the screen resolutions supported by GRUB 2 on a particular system by using the <code>vbeinfo</code> command at the GRUB 2 command line in the running system.</p> <pre data-bbox="810 696 1401 752" style="border: 1px solid #ccc; padding: 5px;"><gfxmode>1280x1024x24</gfxmode></pre>
<u>os_prober</u>	<p>If set to <code>true</code>, automatically searches for operating systems already installed and generates boot entries for them during the installation</p> <pre data-bbox="810 1003 1401 1093" style="border: 1px solid #ccc; padding: 5px;"><os_prober config:type="boolean">false</os_prober></pre>
<u>cpu_mitigations</u>	<p>Allows to choose a default setting of kernel boot command line parameters for CPU mitigation (and at the same time strike a balance between security and performance). Possible values are:</p> <p><u>auto</u>. 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.</p> <p><u>nosmt</u>. Provides the full set of available security mitigations. Enables all mitigations required for your CPU model. In addition, it disables Simultaneous Multithreading (SMT)</p>

Attribute	Description
	<p>to avoid side-channel attacks across multiple CPU threads. This setting may further impact performance, depending on the workload.</p> <p><u>off</u>. Disables all mitigations. Side-channel attacks against your CPU are possible, depending on the CPU model. This setting has no impact on performance.</p> <p><u>manual</u>. Does not set any mitigation level. Specify your CPU mitigations manually by using the kernel command line options.</p> <pre data-bbox="809 779 1412 840" style="border: 1px solid #ccc; padding: 5px;"><cpu_mitigations>auto</cpu_mitigations></pre> <p>If not set in AutoYaST, the respective settings can be changed via kernel command line. By default, the (product-specific) settings in the <u>/control.xml</u> file on the installation medium are used (if nothing else is specified).</p>
<u>suse_btrfs</u>	<p>Obsolete and no longer used. Booting from Btrfs snapshots is automatically enabled.</p>
<u>serial</u>	<p>Command to execute if the GRUB 2 terminal mode is set to <u>serial</u>.</p> <pre data-bbox="809 1377 1412 1547" style="border: 1px solid #ccc; padding: 5px;"><serial> serial --speed=115200 --unit=0 --word=8 --parity=no --stop=1 </serials></pre>
<u>secure_boot</u>	<p>If set to <u>false</u>, then UEFI secure boot is disabled. Works only for <u>grub2-efi</u> boot loader.</p> <pre data-bbox="809 1742 1412 1803" style="border: 1px solid #ccc; padding: 5px;"><secure_boot">false</secure_boot></pre>

Attribute	Description
<u>terminal</u>	<p>Specify the GRUB 2 terminal mode to use, Valid entries are <u>console</u>, <u>gfxterm</u>, and <u>serial</u>. If set to <u>serial</u>, the serial command needs to be specified with <code><serial></code>, too.</p> <pre data-bbox="809 506 1410 562"><terminal>serial</terminal></pre>
<u>timeout</u>	<p>The timeout in seconds until the default boot entry is booted automatically.</p> <pre data-bbox="809 719 1410 810"><timeout config:type="integer">10</timeout></pre>
<u>trusted_boot</u>	<p>If set to <u>true</u>, then Trusted GRUB is used. Trusted GRUB supports Trusted Platform Module (TPM). Works only for <u>grub2</u> boot loader.</p> <pre data-bbox="809 1061 1410 1117"><trusted_boot">true</trusted_boot></pre>
<u>vgamode</u>	<p>Adds the kernel parameter <u>vga=VALUE</u> to the boot entries.</p> <pre data-bbox="809 1274 1410 1330"><vgamode>0x317</vgamode></pre>
<u>xen_append</u>	<p>Kernel parameters added at the end of boot entries for Xen guests.</p> <pre data-bbox="809 1487 1410 1579"><xen_append>nomodeset vga=0x317</xen_append></pre>
<u>xen_kernel_append</u>	<p>Kernel parameters added at the end of boot entries for Xen kernels on the VM Host Server.</p>

Attribute	Description
	<pre data-bbox="818 237 1286 304"><xen_kernel_append>dom0_mem=768M</xen_kernel_append></pre>

4.3.3 Device map

GRUB 2 avoids mapping problems between BIOS drives and Linux devices by using device ID strings (UUIDs) or file system labels when generating its configuration files. GRUB 2 utilities create a temporary device map on the fly, which is usually sufficient, particularly on single-disk systems. However, if you need to override the automatic device mapping mechanism, create your custom mapping in this section.

```
<device_map config:type="list">
  <device_map_entry>
    <firmware>hd0</firmware> <!-- order of devices in target map -->
    <linux>/dev/disk/by-id/ata-ST3500418AS_6VM23FX0</linux> <!-- name of device (disk) -->
  </device_map_entry>
</device_map>
```

4.4 Partitioning

When it comes to partitioning, we can categorize AutoYaST use cases into three different levels:

- Automatic partitioning. The user does not care about the partitioning and trusts in AutoYaST to do the right thing.
- Guided partitioning. The user would like to set some basic settings. For example, a user would like to use LVM but has no idea about how to configure partitions, volume groups, and so on.
- Expert partitioning. The user specifies how the layout should look. However, a complete definition is not required, and AutoYaST should propose reasonable defaults for missing parts.

To some extent, it is like using the regular installer. You can skip the partitioning screen and trust in YaST, use the *Guided Proposal*, or define the partitioning layout through the *Expert Partitioner*.

4.4.1 Automatic Partitioning

AutoYaST can come up with a sensible partitioning layout without any user indication. Although it depends on the selected product to install, AutoYaST usually proposes a Btrfs root file system, a separate `/home` using XFS and a swap partition. Additionally, depending on the architecture, it adds any partition that might be needed to boot (like BIOS GRUB partitions).

However, these defaults might change depending on factors like the available disk space. For example, having a separate `/home` depends on the amount of available disk space.

If you want to influence these default values, you can use the approach described in [Section 4.4.2, “Guided Partitioning”](#).

4.4.2 Guided Partitioning

Although AutoYaST can come up with a partitioning layout without any user indication, sometimes it is useful to set some generic parameters and let AutoYaST do the rest. For example, you may be interested in using LVM or encrypting your file systems without having to deal with the details. It is similar to what you would do when using the guided proposal in a regular installation.

The `storage` section in [Example 4.3, “LVM-based Guided Partitioning”](#) instructs AutoYaST to set up a partitioning layout using LVM and deleting all Windows partitions, no matter if needed or not.

EXAMPLE 4.3: LVM-BASED GUIDED PARTITIONING

```
<general>
  <storage>
    <proposal>
      <lvm config:type="boolean">true<lvm>
      <windows_delete_mode config:type="symbol">all<windows_delete_mode>
    </proposal>
  </storage>
</general>
```

lvm

Create a LVM-based proposal. The default is `false`.

```
<lvm config:type="boolean">true</lvm>
```

resize_windows

When set to `true`, AutoYaST resizes Windows partitions if needed to make room for the installation.

```
<resize_windows config:type="boolean">false</resize_windows>
```

windows_delete_mode

- none does not remove Windows partitions.
- ondemand removes Windows partitions if needed.
- all removes all Windows partitions.

```
<windows_delete_mode config:type="symbol">ondemand</windows_delete_mode>
```

linux_delete_mode

- none does not remove Linux partitions.
- ondemand removes Linux partitions if needed.
- all removes all Linux partitions.

```
<linux_delete_mode config:type="symbol">ondemand</linux_delete_mode>
```

other_delete_mode

- none does not remove other partitions.
- ondemand removes other partitions if needed.
- all removes all other partitions.

```
<other_delete_mode config:type="symbol">ondemand</other_delete_mode>
```

encryption_password

Enables encryption using the specified password. By default, encryption is disabled.

```
<encryption_password>some-secret</encryption_password>
```

4.4.3 Expert Partitioning

As an alternative to the guided partitioning, AutoYaST allows to describe the partitioning layout through a `partitioning` section. However, AutoYaST does not need to know every single detail and it is able to build a sensible layout from a rather incomplete specification.

The `partitioning` section is a list of `drive` elements. Each of these sections describes an element of the partitioning layout like a disk, an LVM volume group, a RAID, a multi-device Btrfs file system, and so on.

In *Example 4.4, "Creating /, /home and swap partitions"*, asks AutoYaST to create a `/`, a `/home` and a `swap` partition using the whole disk. Note that some information is missing, like which file systems each partition should use. However, that is not a problem, and AutoYaST will propose sensible values for them.

EXAMPLE 4.4: CREATING /, /home AND swap PARTITIONS

```
<partitioning config:type="list">
  <drive>
    <use>all</use>
    <partitions config:type="list">
      <partition>
        <mount>/</mount>
        <size>20GiB</size>
      </partition>
      <partition>
        <mount>/home</mount>
        <size>max</size>
      </partition>
      <partition>
        <mount>swap</mount>
        <size>1GiB</size>
      </partition>
    </partitions>
  </drive>
```



Tip: Proposing a Boot Partition

AutoYaST checks whether the layout described in the profile is bootable or not. If that is not the case, it adds the missing partitions. So, if you are unsure about which partitions are needed to boot, you can rely on AutoYaST to make the right decision.

4.4.3.1 Drive Configuration

The elements listed below must be placed within the following XML structure:

```
<profile>
  <partitioning config:type="list">
    <drive>
      ...
    </drive>
```

```
</partitioning>
</profile>
```

Attribute	Values	Description
<p><u>device</u></p>	<p>The device you want to configure in this section. You can use persistent device names via ID, like <code>/dev/disk/by-id/ata-WDC_WD3200AAKS-75L9A0_WD-WMAV27368122</code> or <i>by-path</i>, like <code>/dev/disk/by-path/pci-0001:00:03.0-scsi-0:0:0:0</code>.</p> <pre><device>/dev/sda</device></pre> <p>In case of volume groups, software RAID or <code>bcache</code> devices, the name in the installed system may be different (to avoid clashes with existing devices).</p> <p>See Section 4.4.7, "Multipath Support" for further information about dealing with multipath devices.</p>	<p>Optional. If left out, AutoYaST tries to guess the device. See Tip: Skipping Devices on how to influence guessing.</p> <p>If set to <code>ask</code>, AutoYaST will ask the user which device to use during installation.</p>
<p><u>initialize</u></p>	<p>If set to <code>true</code>, the partition table gets wiped out before AutoYaST starts the partition calculation.</p> <pre><initialize config:type="boolean">true</initialize></pre>	<p>Optional. The default is <code>false</code>.</p>

Attribute	Values	Description
<u>partitions</u>	<p>A list of <partition> entries (see Section 4.4.3.2, "Partition Configuration").</p> <pre data-bbox="600 412 991 651"> <partitions config:type="list"> <partition>...</ partition> ... </partitions> </pre>	<p>Optional. If no partitions are specified, AutoYaST will create a reasonable partitioning (see Section 4.4.3.5, "Filling the Gaps").</p>
<u>pesize</u>	<p>This value only makes sense with LVM.</p> <pre data-bbox="600 801 991 864"> <pesize>8M</pesize> </pre>	<p>Optional. Default is 4M for LVM volume groups.</p>
<u>use</u>	<p>Specifies the strategy AutoYaST will use to partition the hard disk.</p> <p>Choose between:</p> <ul data-bbox="643 1128 991 1800" style="list-style-type: none"> • <u>all</u> (uses the whole device while calculating the new partitioning), • <u>linux</u> (only existing Linux partitions are used), • <u>free</u> (only unused space on the device is used, no other partitions are touched), • 1,2,3 (a list of comma separated partition numbers to use). 	<p>This parameter should be provided.</p>

Attribute	Values	Description
<u>type</u>	<p>Specify the type of the <u>dri-</u> <u>ve</u>,</p> <p>Choose between:</p> <ul style="list-style-type: none"> • <u>CT_DISK</u> for physical hard disks (default), • <u>CT_LVM</u> for LVM volume groups, • <u>CT_RAID</u> for software RAID devices, • <u>CT_BCACHE</u> for software <u>bcache</u> devices. <pre data-bbox="603 907 994 1037" style="border: 1px solid gray; padding: 5px;"> <type config:type="symbol">CT_LVM< type></pre>	<p>Optional. Default is <u>CT_DISK</u> for a normal physical hard disk.</p>
<u>disklabel</u>	<p>Describes the type of the partition table.</p> <p>Choose between:</p> <ul style="list-style-type: none"> • <u>msdos</u> • <u>gpt</u> • <u>none</u> <pre data-bbox="603 1473 994 1534" style="border: 1px solid gray; padding: 5px;"> <disklabel>gpt</disklabel></pre>	<p>Optional. By default YaST decides what makes sense. If a partition table of a different type already exists, it will be recreated with the given type only if it does not include any partition that should be kept or reused. To use the disk without creating any partition, set this element to <u>none</u>.</p>
<u>keep_unknown_lv</u>	<p>This value only makes sense for <u>type = CT_LVM</u> drives. If you are reusing a logical volume group and you set this to <u>true</u>, all existing logical</p>	<p>Optional. The default is <u>false</u>.</p>

Attribute	Values	Description
	<p>volumes in that group will not be touched unless they are specified in the <code><partitioning></code> section. So you can keep existing logical volumes without specifying them.</p> <pre><keep_unknown_lv config:type="boolean" >false</ keep_unknown_lv></pre>	
<u>enable_snapshots</u>	<p>Enables snapshots on Btrfs file systems mounted at <code>/</code> (does not apply to other file systems, or Btrfs file systems not mounted at <code>/</code>).</p> <pre><enable_snapshots config:type="boolean" >false</ enable_snapshots></pre>	Optional. The default is <u>true</u> .

Important: Beware of Data Loss

The value provided in the use property determines how existing data and partitions are treated. The value all means that the entire disk will be erased. Make backups and use the confirm property if you need to keep some partitions with important data. Otherwise, no pop-ups will notify you about partitions being deleted.

Tip: Skipping Devices

You can influence AutoYaST's device-guessing for cases where you do not specify a `<device>` entry on your own. Usually AutoYaST would use the first device it can find that looks reasonable but you can configure it to skip some devices like this:

```
<partitioning config:type="list">
  <drive>
```

```

<initialize config:type="boolean">true</initialize>
<skip_list config:type="list">
  <listentry>
    <!-- skip devices that use the usb-storage driver -->
    <skip_key>driver</skip_key>
    <skip_value>usb-storage</skip_value>
  </listentry>
  <listentry>
    <!-- skip devices that are smaller than 1GB -->
    <skip_key>size_k</skip_key>
    <skip_value>1048576</skip_value>
    <skip_if_less_than config:type="boolean">true</skip_if_less_than>
  </listentry>
  <listentry>
    <!-- skip devices that are larger than 100GB -->
    <skip_key>size_k</skip_key>
    <skip_value>104857600</skip_value>
    <skip_if_more_than config:type="boolean">true</skip_if_more_than>
  </listentry>
</skip_list>
</drive>
</partitioning>

```

For a list of all possible `<skip_key>`s, run `yast2 ayast_probe` on a system that has already been installed.

4.4.3.2 Partition Configuration

The elements listed below must be placed within the following XML structure:

```

<drive>
  <partitions config:type="list">
    <partition>
      ...
    </partition>
  </partitions>
</drive>

```

Attribute	Values	Description
<u>create</u>	Specify if this partition or logical volume must be created or if it already exists. <pre data-bbox="391 360 995 461"><create config:type="boolean" >false</create></pre>	If set to <u>false</u> , you also need to set one of <u>partition_nr</u> , <u>lv_name</u> , <u>label</u> or <u>uuid</u> to tell AutoYaST which device to use.
<u>crypt_method</u>	Partition will be encrypted using one of these methods: <ul data-bbox="432 645 995 1149" style="list-style-type: none"> • <u>luks1</u>: regular LUKS1 encryption. • <u>pervasive_luks2</u>: pervasive volume encryption. • <u>protected_swap</u>: encryption with volatile protected key. • <u>secure_swap</u>: encryption with volatile secure key. • <u>random_swap</u>: encryption with volatile random key. <pre data-bbox="391 1193 995 1283"><crypt_method config:type="symbol">luks1</crypt_method></pre>	Optional. Encryption method selection was introduced in openSUSE Leap 15.2. To mimic the behaviour of previous versions, use <u>luks1</u> . See <u>crypt_key</u> element to find out how to specify the encryption password if needed.
<u>crypt_fs</u>	Partition will be encrypted. This element is deprecated, use <u>crypt_method</u> instead. <pre data-bbox="391 1440 995 1529"><crypt_fs config:type="boolean">true</crypt_fs></pre>	Default is <u>false</u> .
<u>crypt_key</u>	Encryption key <pre data-bbox="391 1641 995 1697"><crypt_key>xxxxxxx</crypt_key></pre>	Needed if <u>crypt_method</u> has been set to a method that requires a password (i.e., <u>luks1</u> or <u>pervasive_luks2</u>).

Attribute	Values	Description
<u>mount</u>	<p>The mount point of this partition.</p> <pre><mount>/</mount> <mount>swap</mount></pre>	You should have at least a root partition (/) and a swap partition.
<u>fstop</u>	<p>Mount options for this partition.</p> <pre><fstopt> ro,noatime,user,data=ordered,acl,user_xattr </fstopt></pre>	See man mount for available mount options.
<u>label</u>	<p>The label of the partition. Useful when formatting the device (especially if the <u>mount-by</u> parameter is set to <u>label</u>) and for identifying a device that already exists (see <u>create</u> above).</p> <pre><label>mydata</label></pre>	See man e2label for an example.
<u>uuid</u>	<p>The uuid of the partition. Only useful for identifying an existing device (see <u>create</u> above), the uuid cannot be enforced for new devices.</p> <pre><uuid >1b4e28ba-2fa1-11d2-883f-b9a761bde3fb< uuid></pre>	See man uuidgen .
<u>size</u>	<p>The size of the partition, for example 4G, 4500M, etc. The /boot partition and the swap partition can have <u>auto</u> as size. Then AutoYaST calculates a reasonable size. One partition can have the value <u>max</u> to use all remaining space.</p>	Starting with openSUSE Leap 15, all values (including <u>auto</u> and <u>max</u>) can be used for resizing partitions as well.

Attribute	Values	Description
	<p>You can also specify the size in percentage. So 10% will use 10% of the size of the hard disk or volume group. You can mix auto, max, size, and percentage as you like.</p> <pre data-bbox="391 448 994 506" style="border: 1px solid #ccc; padding: 5px;"><size>10G</size></pre>	
<u>format</u>	<p>Specify if AutoYaST should format the partition.</p> <pre data-bbox="391 660 994 752" style="border: 1px solid #ccc; padding: 5px;"><format config:type="boolean">>false</format></pre>	<p>If you set <u>create</u> to <u>true</u>, then you likely want this option set to <u>true</u> as well.</p>
<u>file system</u>	<p>Specify the file system to use on this partition:</p> <ul style="list-style-type: none"> • <u>btrfs</u> • <u>ext2</u> • <u>ext3</u> • <u>ext4</u> • <u>fat</u> • <u>xf</u>s • <u>swap</u> <pre data-bbox="391 1417 994 1509" style="border: 1px solid #ccc; padding: 5px;"><filesystem config:type="symbol">ext3</filesystem></pre>	<p>Optional. The default is <u>btrfs</u> for the root partition (<u>/</u>) and <u>xf</u>s for data partitions.</p>
<u>mkfs_options</u>	<p>Specify an option string that is added to the mkfs command.</p> <pre data-bbox="391 1664 994 1722" style="border: 1px solid #ccc; padding: 5px;"><mkfs_options>-I 128</mkfs_options></pre>	<p>Optional. Only use this when you know what you are doing.</p>

Attribute	Values	Description
<u>partition_nr</u>	<p>The partition number of this partition. If you have set <u>create=false</u> or if you use LVM, then you can specify the partition via <u>partition_nr</u>. You can force AutoYaST to only create primary partitions by assigning numbers below 5.</p> <pre data-bbox="391 555 994 649"><partition_nr config:type="integer">2</partition_nr></pre>	Usually, numbers 1 to 4 are primary partitions while 5 and higher are logical partitions.
<u>partition_id</u>	<p>The <u>partition_id</u> sets the id of the partition. If you want different identifiers than 131 for Linux partition or 130 for swap, configure them with <u>partition_id</u>.</p> <pre data-bbox="391 898 994 992"><partition_id config:type="integer">131</partition_id></pre> <p>Possible values are:</p> <p>Swap: <u>130</u> Linux: <u>131</u> LVM: <u>142</u> MD RAID: <u>253</u> EFI partition: <u>259</u></p>	The default is <u>131</u> for Linux partition and <u>130</u> for swap.
<u>partition_type</u>	<p>When using an <u>msdos</u> partition table, this element sets the type of the partition. The value can be <u>primary</u> or <u>logical</u>. This value is ignored when using a <u>gpt</u> partition table, because such a distinction does not exist in that case.</p> <pre data-bbox="391 1675 994 1724"><partition_type>primary</partition_type></pre>	Optional. Allowed values are <u>primary</u> (default) and <u>logical</u> .

Attribute	Values	Description
<u>mountby</u>	<p>Instead of a partition number, you can tell AutoYaST to mount a partition by <u>device</u>, <u>label</u>, <u>uuid</u>, <u>path</u> or <u>id</u>, which are the udev path and udev id (see <u>/dev/disk/...</u>).</p> <pre data-bbox="391 504 994 600" style="border: 1px solid gray; padding: 5px;"> <mountby config:type="symbol" >label</mountby></pre>	<p>See <u>label</u> and <u>uuid</u> documentation above. The default depends on YaST and usually is <u>id</u>.</p>
<u>subvolumes</u>	<p>List of subvolumes to create for a file system of type Btrfs. This key only makes sense for file systems of type Btrfs. See Section 4.4.3.3, "Btrfs subvolumes" for more information.</p> <pre data-bbox="391 846 994 1267" style="border: 1px solid gray; padding: 5px;"> <subvolumes config:type="list"> <path>tmp</path> <path>opt</path> <path>srv</path> <path>var/crash</path> <path>var/lock</path> <path>var/run</path> <path>var/tmp</path> <path>var/spool</path> ... </subvolumes></pre>	<p>If no <u>subvolumes</u> section has been defined for a partition description, AutoYaST will create a predefined set of subvolumes for the given mount point.</p>
<u>create_subvolumes</u>	<p>Determine whether Btrfs subvolumes should be created or not.</p>	<p>It is set to <u>true</u> by default. When set to <u>false</u>, no subvolumes will be created.</p>
<u>subvolumes_prefix</u>	<p>Set the Btrfs subvolumes prefix name. If no prefix is wanted, it must be set to an empty value:</p> <pre data-bbox="391 1641 994 1738" style="border: 1px solid gray; padding: 5px;"> <subvolumes_prefix><![CDATA[]]></subvolumes_prefix></pre>	<p>It is set to <u>@</u> by default.</p>

Attribute	Values	Description
<u>lv_name</u>	<p>If this partition is on a logical volume in a volume group, specify the logical volume name here (see the <u>is_lvm_vg</u> parameter in the drive configuration).</p> <pre><lv_name>opt_lv</lv_name></pre>	
<u>stripes</u>	<p>An integer that configures LVM striping. Specify across how many devices you want to stripe (spread data).</p> <pre><stripes config:type="integer">2</stripes></pre>	
<u>stripesize</u>	<p>Specify the size of each block in KB.</p> <pre><stripesize config:type="integer">4</stripesize></pre>	
<u>lvm_group</u>	<p>If this is a physical partition used by (part of) a volume group (LVM), you need to specify the name of the volume group here.</p> <pre><lvm_group>system</lvm_group></pre>	
<u>pool</u>	<p><u>pool</u> must be set to <u>true</u> if the LVM logical volume should be an LVM thin pool.</p> <pre><pool config:type="boolean">>false</pool></pre>	
<u>used_pool</u>	<p>The name of the LVM thin pool that is used as a data store for this thin logical volume. If this is set to something non-empty, it implies that the volume is a so-called thin logical volume.</p> <pre><used_pool>my_thin_pool</used_pool></pre>	

Attribute	Values	Description
<u>raid_name</u>	<p>If this physical volume is part of a RAID, specify the name of the RAID.</p> <pre><raid_name>/dev/md/0</raid_name></pre>	
<u>raid_options</u>	<p>Specify RAID options, see below.</p> <pre><raid_options>...</raid_options></pre>	<p>Setting the RAID options at <u>partition</u> level is deprecated. See Section 4.4.6, "Software RAID".</p>
<u>bcache_backing_for</u>	<p>If this device is used as a <u>bcache backing device</u>, specify the name of the <u>bcache</u> device.</p> <pre><bcache_backing_for>/dev/bcache0</bcache_backing_for></pre>	<p>See Section 4.4.8, "bcache Configuration" for further details.</p>
<u>bcache_caching_for</u>	<p>If this device is used as a <u>bcache caching device</u>, specify the names of the <u>bcache</u> devices.</p> <pre><bcache_caching_for config:type="list"> <listentry>/dev/bcache0</listentry> </bcache_caching_for></pre>	<p>See Section 4.4.8, "bcache Configuration" for further details.</p>
<u>resize</u>	<p>This boolean must be <u>true</u> if an existing partition should be resized. In this case, you need to tell AutoYaST to reuse the device (see <u>create</u>) and specify a <u>size</u>.</p> <pre><resize config:type="boolean">false</resize></pre>	<p>Starting with openSUSE Leap 15 resizing works with physical disk partitions and with LVM volumes.</p>

4.4.3.3 Btrfs subvolumes

As mentioned in [Section 4.4.3.2, "Partition Configuration"](#), it is possible to define a set of subvolumes for each Btrfs file system. In its simplest form, this is a list of entries:

```
<subvolumes config:type="list">
  <path>tmp</path>
  <path>opt</path>
  <path>srv</path>
  <path>var/crash</path>
  <path>var/lock</path>
  <path>var/run</path>
  <path>var/tmp</path>
  <path>var/spool</path>
</subvolumes>
```

AutoYaST supports disabling copy-on-write for a given subvolume. In that case, a slightly more complex syntax should be used:

```
<subvolumes config:type="list">
<listentry>tmp</listentry>
<listentry>opt</listentry>
<listentry>srv</listentry>
<listentry>
  <path>var/lib/pgsql</path>
  <copy_on_write config:type="boolean">>false</copy_on_write>
</listentry>
</subvolumes>
```

If there is a default subvolume used for the distribution (for example `@` in openSUSE Leap), the name of this default subvolume is automatically prefixed to the names in this list. This behavior can be disabled by setting the `subvolumes_prefix`.

```
<subvolumes_prefix><![CDATA[]]></subvolumes_prefix>
```

4.4.3.4 Using the Whole Disk

AutoYaST allows to use a whole disk without creating any partition by setting the `disklabel` to `none` as described in [Section 4.4.3.1, "Drive Configuration"](#). In such cases, the configuration in the first `partition` from the `drive` will be applied to the whole disk.

In the example below, we are using the second disk (`/dev/sdb`) as the `/home` file system.

EXAMPLE 4.5: USING A WHOLE DISK AS A FILE SYSTEM

```
<partitioning config:type="list">
```

```

<drive>
  <device>/dev/sda</device>
  <partitions config:type="list">
    <partition>
      <create config:type="boolean">>true</create>
      <format config:type="boolean">>true</format>
      <mount>/</mount>
      <size>max</size>
    </partition>
  </partitions>
</drive>
<drive>
  <device>/dev/sdb</device>
  <disklabel>none</disklabel>
  <partitions config:type="list">
    <partition>
      <format config:type="boolean">>true</format>
      <mount>/home</mount>
    </partition>
  </partitions>
</drive>

```

In addition, the whole disk can be used as an LVM physical volume or as a software RAID member. See [Section 4.4.5, “Logical Volume Manager \(LVM\)”](#) and [Section 4.4.6, “Software RAID”](#) for further details about setting up an LVM or a software RAID.

For backward compatibility reasons, it is possible to achieve the same result by setting the `<partition_nr>` element to `0`. However, this usage of the `<partition_nr>` element is deprecated since openSUSE Leap 15.

4.4.3.5 Filling the Gaps

When using the Expert Partitioner approach, AutoYaST can create a partition plan from a rather incomplete profile. The following profiles show how you can describe some details of the partitioning layout and let AutoYaST do the rest.

EXAMPLE 4.6: AUTOMATED PARTITIONING ON SELECTED DRIVES

The following is an example of a single drive system, which is not pre-partitioned and should be automatically partitioned according to the described pre-defined partition plan. If you do not specify the device, it will be automatically detected.

```
<partitioning config:type="list">
```

```
<drive>
  <device>/dev/sda</device>
  <use>all</use>
</drive>
</partitioning>
```

A more detailed example shows how existing partitions and multiple drives are handled.

EXAMPLE 4.7: INSTALLING ON MULTIPLE DRIVES

```
<partitioning config:type="list">
  <drive>
    <device>/dev/sda</device>
    <use>all</use>
    <partitions config:type="list">
      <partition>
        <mount>/</mount>
        <size>10G</size>
      </partition>
      <partition>
        <mount>swap</mount>
        <size>1G</size>
      </partition>
    </partitions>
  </drive>
  <drive>
    <device>/dev/sdb</device>
    <use>free</use>
    <partitions config:type="list">
      <partition>
        <filesystem config:type="symbol">ext4</filesystem>
        <mount>/data1</mount>
        <size>15G</size>
      </partition>
      <partition>
        <filesystem config:type="symbol">xfs</filesystem>
        <mount>/data2</mount>
        <size>auto</size>
      </partition>
    </partitions>
  </drive>
</partitioning>
```

4.4.4 Advanced Partitioning Features

4.4.4.1 Wipe out Partition Table

Usually this is not needed because AutoYaST can delete partitions one by one automatically. But you need the option to let AutoYaST clear the partition table instead of deleting partitions individually.

Go to the drive section and add:

```
<initialize config:type="boolean">true</initialize>
```

With this setting AutoYaST will delete the partition table before it starts to analyze the actual partitioning and calculates its partition plan. Of course this means, that you cannot keep any of your existing partitions.

4.4.4.2 Mount Options

By default a file system to be mounted is identified in /etc/fstab by the device name. This identification can be changed so the file system is found by searching for a UUID or a volume label. Note that not all file systems can be mounted by UUID or a volume label. To specify how a partition is to be mounted, use the mountby property which has the symbol type. Possible options are:

- device (default)
- label
- UUID

If you choose to mount a new partition using a label, use the label property to specify its value. Add any valid mount option in the fourth field of /etc/fstab. Multiple options are separated by commas. Possible fstab options:

Mount read-only (ro)

No write access to the file system. Default is false.

No access time (noatime)

Access times are not updated when a file is read. Default is false.

Mountable by User (user)

The file system can be mounted by a normal user. Default is false.

Data Journaling Mode (ordered, journal, writeback)

journal

All data is committed to the journal prior to being written to the main file system.

ordered

All data is directly written to the main file system before its metadata is committed to the journal.

writeback

Data ordering is not preserved.

Access Control List (acl)

Enable access control lists on the file system.

Extended User Attributes (user_xattr)

Allow extended user attributes on the file system.

EXAMPLE 4.8: MOUNT OPTIONS

```
<partitions config:type="list">
  <partition>
    <filesystem config:type="symbol">ext4</filesystem>
    <format config:type="boolean">>true</format>
    <fstopt>ro,noatime,user,data=ordered,acl,user_xattr</fstopt>
    <mount>/local</mount>
    <mountby config:type="symbol">uuid</mountby>
    <partition_id config:type="integer">131</partition_id>
    <size>10G</size>
  </partition>
</partitions>
```



Note: Check Supported File System Options

Different file system types support different options. Check the documentation carefully before setting them.

4.4.4.3 Keeping Specific Partitions

In some cases you should leave partitions untouched and only format specific target partitions, rather than creating them from scratch. For example, if different Linux installations coexist, or you have another operating system installed, likely you do not want to wipe these out. You may also want to leave data partitions untouched.

Such scenarios require specific knowledge about the target systems and hard disks. Depending on the scenario, you might need to know the exact partition table of the target hard disk with partition IDs, sizes and numbers. With this data, you can tell AutoYaST to keep certain partitions, format others and create new partitions if needed.

The following example will keep partitions 1, 2 and 5 and delete partition 6 to create two new partitions. All remaining partitions will only be formatted.

EXAMPLE 4.9: KEEPING PARTITIONS

```
<partitioning config:type="list">
  <drive>
    <device>/dev/sdc</device>
    <partitions config:type="list">
      <partition>
        <create config:type="boolean">>false</create>
        <format config:type="boolean">>true</format>
        <mount>/</mount>
        <partition_nr config:type="integer">1</partition_nr>
      </partition>
      <partition>
        <create config:type="boolean">>false</create>
        <format config:type="boolean">>false</format>
        <partition_nr config:type="integer">2</partition_nr>
        <mount>/space</mount>
      </partition>
      <partition>
        <create config:type="boolean">>false</create>
        <format config:type="boolean">>true</format>
        <filesystem config:type="symbol">swap</filesystem>
        <partition_nr config:type="integer">5</partition_nr>
        <mount>swap</mount>
      </partition>
      <partition>
        <format config:type="boolean">>true</format>
        <mount>/space2</mount>
        <size>5G</size>
      </partition>
      <partition>
        <format config:type="boolean">>true</format>
        <mount>/space3</mount>
        <size>max</size>
      </partition>
    </partitions>
    <use>6</use>
  </drive>
```

```
</partitioning>
```

The last example requires exact knowledge of the existing partition table and the partition numbers of those partitions that should be kept. In some cases however, such data may not be available, especially in a mixed hardware environment with different hard disk types and configurations. The following scenario is for a system with a non-Linux OS with a designated area for a Linux installation.

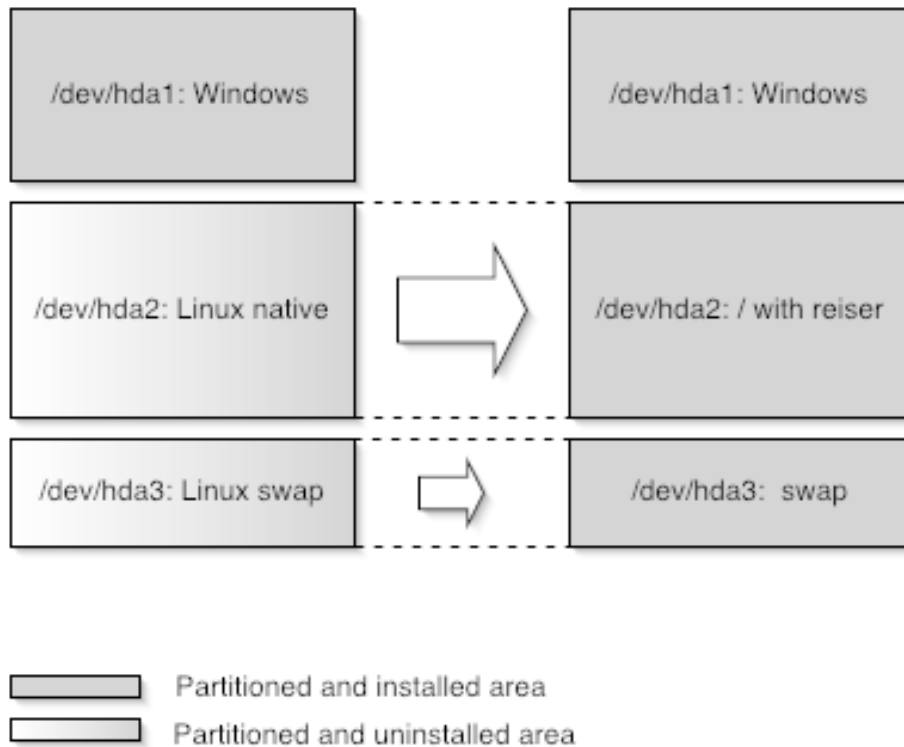


FIGURE 4.1: KEEPING PARTITIONS

In this scenario, shown in figure *Figure 4.1, "Keeping partitions"*, AutoYaST will not create new partitions. Instead it searches for certain partition types on the system and uses them according to the partitioning plan in the control file. No partition numbers are given in this case, only the mount points and the partition types (additional configuration data can be provided, for example file system options, encryption and file system type).

EXAMPLE 4.10: AUTO-DETECTION OF PARTITIONS TO BE KEPT.

```
<partitioning config:type="list">
  <drive>
    <partitions config:type="list">
      <partition>
        <create config:type="boolean">>false</create>
```



```

    <format config:type="boolean">true</format>
    <mount>/</mount>
    <partition_id config:type="integer">131</partition_id>
  </partition>
  <partition>
    <create config:type="boolean">>false</create>
    <format config:type="boolean">true</format>
    <filesystem config:type="symbol">swap</filesystem>
    <partition_id config:type="integer">130</partition_id>
    <mount>swap</mount>
  </partition>
</partitions>
</drive>
</partitioning>

```



Note: Keeping Encrypted Devices

When AutoYaST is probing the storage devices, the partitioning section from the profile is not yet analyzed. In some scenarios, it is not clear which key should be used to unlock a device. For example, this can happen when more than one encryption key is defined. To solve this problem, AutoYaST will try all defined keys on all encrypted devices until a working key is found.

4.4.5 Logical Volume Manager (LVM)

To configure LVM, first create a physical volume using the normal partitioning method described above.

EXAMPLE 4.11: CREATE LVM PHYSICAL VOLUME

The following example shows how to prepare for LVM in the `partitioning` resource. A non-formatted partition is created on device `/dev/sda1` of the type `LVM` and with the volume group `system`. This partition will use all space available on the drive.

```

<partitioning config:type="list">
  <drive>
    <device>/dev/sda</device>
    <partitions config:type="list">
      <partition>
        <create config:type="boolean">true</create>
        <lvm_group>system</lvm_group>
        <partition_type>primary</partition_type>
        <partition_id config:type="integer">142</partition_id>
      </partition>
    </partitions>
  </drive>
</partitioning>

```

```

    <partition_nr config:type="integer">1</partition_nr>
    <size>max</size>
  </partition>
</partitions>
<use>all</use>
</drive>
</partitioning>

```

EXAMPLE 4.12: LVM LOGICAL VOLUMES

```

<partitioning config:type="list">
  <drive>
    <device>/dev/sda</device>
    <partitions config:type="list">
      <partition>
        <lvm_group>system</lvm_group>
        <partition_type>primary</partition_type>
        <size>max</size>
      </partition>
    </partitions>
    <use>all</use>
  </drive>
  <drive>
    <device>/dev/system</device>
    <is_lvm_vg config:type="boolean">>true</is_lvm_vg>
    <partitions config:type="list">
      <partition>
        <filesystem config:type="symbol">ext4</filesystem>
        <lv_name>user_lv</lv_name>
        <mount>/usr</mount>
        <size>15G</size>
      </partition>
      <partition>
        <filesystem config:type="symbol">ext4</filesystem>
        <lv_name>opt_lv</lv_name>
        <mount>/opt</mount>
        <size>10G</size>
      </partition>
      <partition>
        <filesystem config:type="symbol">ext4</filesystem>
        <lv_name>var_lv</lv_name>
        <mount>/var</mount>
        <size>1G</size>
      </partition>
    </partitions>
    <pesize>4M</pesize>
  </drive>
</partitioning>

```

```
</drive>
</partitioning>
```

It is possible to set the `size` to `max` for the logical volumes. Of course, you can only use `max` for one(!) logical volume. You cannot set two logical volumes in one volume group to `max`.

4.4.6 Software RAID

The support for software RAID devices has been greatly improved in openSUSE Leap 15.2.

If needed, see [Section 4.4.6.1, "Using the Deprecated Syntax"](#) to find out further details about the old way of specifying a software RAID, which is still supported for backward compatibility.

Using AutoYaST, you can create and assemble software RAID devices. The supported RAID levels are the following:

RAID 0

This level increases your disk performance. There is *no* redundancy in this mode. If one of the drives crashes, data recovery will not be possible.

RAID 1

This mode offers the best redundancy. It can be used with two or more disks. An exact copy of all data is maintained on all disks. As long as at least one disk is still working, no data is lost. The partitions used for this type of RAID should have approximately the same size.

RAID 5

This mode combines management of a larger number of disks and still maintains some redundancy. This mode can be used on three disks or more. If one disk fails, all data is still intact. If two disks fail simultaneously, all data is lost.

Multipath

This mode allows access to the same physical device via multiple controllers for redundancy against a fault in a controller card. This mode can be used with at least two devices.

Similar to LVM, a software RAID definition in an AutoYaST profile is composed of two different parts:

- Determining which disks or partitions are going to be used as RAID members. In order to do that, you need to set the `raid_name` element in such devices.
- Defining the RAID itself by using a dedicated `drive` section.

The following example shows a RAID1 configuration that uses a partition from the first disk and another one from the second disk as RAID members:

EXAMPLE 4.13: RAID1 CONFIGURATION

```
<partitioning config:type="list">
  <drive>
    <device>/dev/sda</device>
    <partitions config:type="list">
      <partition>
        <mount>/</mount>
        <size>20G</size>
      </partition>
      <partition>
        <raid_name>/dev/md/0</raid_name>
        <size>max</size>
      </partition>
    </partitions>
    <use>all</use>
  </drive>
  <drive>
    <device>/dev/sdb</device>
    <disklabel>none</disklabel>
    <partitions config:type="list">
      <partition>
        <raid_name>/dev/md/0</raid_name>
      </partition>
    </partitions>
    <use>all</use>
  </drive>
  <drive>
    <device>/dev/md/0</device>
    <partitions config:type="list">
      <partition>
        <mount>/home</mount>
        <size>40G</size>
      </partition>
      <partition>
        <mount>/srv</mount>
        <size>10G</size>
      </partition>
    </partitions>
    <raid_options>
      <chunk_size>4</chunk_size>
      <parity_algorithm>left_asymmetric</parity_algorithm>
      <raid_type>raid1</raid_type>
    </raid_options>
  </drive>
</partitioning config:type="list">
```

```
<use>all</use>
</drive>
</partitioning>
```

If you do not want to create partitions in the software RAID, set the `disklabel` to `none` as you would do for a regular disk. In the example below, only the RAID `drive` section is shown for simplicity's sake:

EXAMPLE 4.14: RAID1 WITHOUT PARTITIONS

```
<drive>
  <device>/dev/md/0</device>
  <disklabel>none</disklabel>
  <partitions config:type="list">
    <partition>
      <mount>/home</mount>
      <size>40G</size>
    </partition>
  </partitions>
  <raid_options>
    <chunk_size>4</chunk_size>
    <parity_algorithm>left_asymmetric</parity_algorithm>
    <raid_type>raid1</raid_type>
  </raid_options>
  <use>all</use>
</drive>
```

4.4.6.1 Using the Deprecated Syntax

If the installer self-update feature is enabled, it is possible to partition a software RAID for openSUSE Leap 15. However, that scenario was not supported in previous versions and hence the way to define a software RAID was slightly different.

This section defines what the old-style configuration looks like because it is still supported for backward compatibility.

Keep the following in mind when configuring a RAID using this deprecated syntax:

- The device for RAID is always `/dev/md`.
- The property `partition_nr` is used to determine the MD device number. If `partition_nr` is equal to 0, then `/dev/md/0` is configured. Adding several `partition` sections means that you want to have multiple software RAIDs (`/dev/md/0`, `/dev/md/1`, etc.).
- All RAID-specific options are contained in the `raid_options` resource.

EXAMPLE 4.15: OLD STYLE RAID1 CONFIGURATION

```
<partitioning config:type="list">
  <drive>
    <device>/dev/sda</device>
    <partitions config:type="list">
      <partition>
        <partition_id config:type="integer">253</partition_id>
        <format config:type="boolean">>false</format>
        <raid_name>/dev/md0</raid_name>
        <raid_type>raid1</raid_type>
        <size>4G</size>
      </partition>

      <!-- Insert a configuration for the regular partitions located on
           /dev/sda here (for example / and swap) -->

    </partitions>
    <use>all</use>
  </drive>
  <drive>
    <device>/dev/sdb</device>
    <partitions config:type="list">
      <partition>
        <format config:type="boolean">>false</format>
        <partition_id config:type="integer">253</partition_id>
        <raid_name>/dev/md0</raid_name>
        <size>4gb</size>
      </partition>
    </partitions>
    <use>all</use>
  </drive>
  <drive>
    <device>/dev/md</device>
    <partitions config:type="list">
      <partition>
        <filesystem config:type="symbol">ext4</filesystem>
        <format config:type="boolean">>true</format>
        <mount>/space</mount>
        <partition_id config:type="integer">131</partition_id>
        <partition_nr config:type="integer">0</partition_nr>
        <raid_options>
          <chunk_size>4</chunk_size>
          <parity_algorithm>left_asymmetric</parity_algorithm>
        </raid_options>
      </partition>
    </partitions>
  </drive>
</partitioning>
```

```

    <use>all</use>
  </drive>
</partitioning>

```

4.4.6.2 RAID Options

The following elements must be placed within the following XML structure:

```

<partition>
  <raid_options>
    ...
  </raid_options>
</partition>

```

Attribute	Values	Description
<u>chunk_size</u>	<code><chunk_size>4</chunk_size></code>	
<u>parity_algorithm</u>	<p>Possible values are:</p> <p><u>left_asymmetric</u>, <u>left_symmetric</u>, <u>right_asymmetric</u>, <u>right_symmetric</u>.</p> <p>For RAID6 and RAID10 the following values can be used:</p> <p><u>parity_first</u>, <u>parity_last</u>, <u>left_asymmetric_6</u>, <u>left_symmetric_6</u>, <u>right_asymmetric_6</u>, <u>right_symmetric_6</u>, <u>parity_first_6</u>, <u>n2</u>, <u>o2</u>, <u>f2</u>, <u>n3</u>, <u>o3</u>, <u>f3</u></p> <pre> <parity_algorithm >left_asymmetric</ parity_algorithm> </pre>	

Attribute	Values	Description
<u>raid_type</u>	<p>Possible values are: <u>raid0</u>, <u>raid1</u>, <u>raid5</u>, <u>raid6</u> and <u>raid10</u>.</p> <pre><raid_type>raid1</raid_type></pre>	The default is <u>raid1</u> .
<u>device_order</u>	<p>This list contains the order of the physical devices:</p> <pre><device_order config:type="list"> <device>/dev/ sdb2</device> <device>/dev/ sda1</device> ... </device_order></pre>	This is optional and the default is alphabetical order.

4.4.7 Multipath Support

AutoYaST can handle multipath devices. In order to take advantage of them, you need to enable multipath support, as shown in *Example 4.16, "Using Multipath Devices"*. Alternatively, you can use the following parameter on the Kernel command line: LIBSTORAGE_MULTIPATH_AUTOSTART=ON. Unlike SUSE Linux Enterprise 12, it is not required to set the drive section type to CT_DMMULTI-PATH. You should use CT_DISK, although for historical reasons, both values are equivalent.

EXAMPLE 4.16: USING MULTIPATH DEVICES

```
<general>
  <storage>
    <start_multipath config:type="boolean">>true</start_multipath>
  </storage>
</general>
<partitioning>
  <drive>
    <partitions config:type="list">
      <partition>
        <size>20G</size>
```



```

    <mount>/</mount>
    <filesystem config:type="symbol">ext4</filesystem>
</partition>
<partition>
    <size>auto</size>
    <mount>swap</mount>
</partition>
</partitions>
<type config:type="symbol">CT_DISK</type>
<use>all</use>
</drive>
</partitioning>

```

If you want to specify the device, you could use the World Wide Identifier (WWID), its device name (for example, `/dev/dm-0`), any other path under `/dev/disk` that refers to the multipath device or any of its paths.

For example, given the `multipath` listing from [Example 4.17, “Listing multipath devices”](#), you could use `/dev/mapper/149455400000000000f86756dce9286158be4c6e3567e75ba5`, `/dev/dm-3`, any other corresponding path under `/dev/disk` (like shown in the [Example 4.18, “Using the WWID to Identify a Multipath Device”](#)), or any of its paths (`/dev/sda` or `/dev/sdb`).

EXAMPLE 4.17: LISTING MULTIPATH DEVICES

```

# multipath -l
149455400000000000f86756dce9286158be4c6e3567e75ba5 dm-3 ATA,VIRTUAL-DISK
size=40G features='0' hwhandler='0' wp=rw
|+- policy='service-time 0' prio=1 status=active
| `- 2:0:0:0 sda 8:0 active ready running
`+- policy='service-time 0' prio=1 status=enabled
  `- 3:0:0:0 sdb 8:16 active ready running

```

EXAMPLE 4.18: USING THE WWID TO IDENTIFY A MULTIPATH DEVICE

```

<drive>
  <partitions config:type="list">
    <device>/dev/mapper/149455400000000000f86756dce9286158be4c6e3567e75ba5</device>
    <partition>
      <size>20G</size>
      <mount>/</mount>
      <filesystem config:type="symbol">ext4</filesystem>
    </partition>
  </partitions>
  <type config:type="symbol">CT_DISK</type>
  <use>all</use>
</drive>

```

4.4.8 bcache Configuration

bcache is a caching system which allows the use of multiple fast drives to speed up the access to one or more slower drives. For example, you can improve the performance of a large (but slow) drive by using a fast one as a cache.

For more information about bcache on openSUSE Leap, also see the blog post at <https://www.suse.com/c/combine-the-performance-of-solid-state-drive-with-the-capacity-of-a-hard-drive-with-bcache-and-yast/>.

To set up a bcache device, AutoYaST needs a profile that specifies the following:

- To set a (slow) block device as *backing device*, use the bcache_backing_for element.
- To set a (fast) block device as *caching device*, use the bcache_caching_for element. You can use the same device to speed up the access to several drives.
- To specify the layout of the bcache device, use a drive section and set the type element to CT_BCACHE. The layout of the bcache device may contain partitions.

EXAMPLE 4.19: bcache DEFINITION

```
<partitioning config:type="list">
  <drive>
    <device>/dev/sda</device>
    <type config:type="symbol">CT_DISK</type>
    <use>all</use>
    <enable_snapshots config:type="boolean">>true</enable_snapshots>
    <partitions config:type="list">
      <partition>
        <filesystem config:type="symbol">btrfs</filesystem>
        <mount>/</mount>
        <create config:type="boolean">>true</create>
        <size>max</size>
      </partition>
      <partition>
        <filesystem config:type="symbol">swap</filesystem>
        <mount>swap</mount>
        <create config:type="boolean">>true</create>
        <size>2GiB</size>
      </partition>
    </partitions>
  </drive>

  <drive>
    <type config:type="symbol">CT_DISK</type>
    <device>/dev/sdb</device>
```

```

<disklabel>msdos</disklabel>
<use>all</use>
<partitions config:type="list">
  <partition>
    <!-- It can serve as caching device for several bcachees -->
    <bcache_caching_for config:type="list">
      <listentry>/dev/bcache0</listentry>
    </bcache_caching_for>
    <size>max</size>
  </partition>
</partitions>
</drive>

<drive>
  <type config:type="symbol">CT_DISK</type>
  <device>/dev/sdc</device>
  <use>all</use>
  <disklabel>msdos</disklabel>
  <partitions config:type="list">
    <partition>
      <!-- It can serve as backing device for one bcache -->
      <bcache_backing_for>/dev/bcache0</bcache_backing_for>
    </partition>
  </partitions>
</drive>

<drive>
  <type config:type="symbol">CT_BCACHE</type>
  <device>/dev/bcache0</device>
  <bcache_options>
    <cache_mode>writethrough</cache_mode>
  </bcache_options>
  <use>all</use>
  <partitions config:type="list">
    <partition>
      <mount>/data</mount>
      <size>20GiB</size>
    </partition>
    <partition>
      <mount>swap</mount>
      <filesystem config:type="symbol">swap</filesystem>
      <size>1GiB</size>
    </partition>
  </partitions>
</drive>
</partitioning>

```

For the time being, the only supported option in the `bcache_options` section is `cache_mode`, described in the table below.

Attribute	Values	Description
<code>cache_mode</code>	<code><cache_mode>writethrough</cache_mode></code>	Cache mode for <code>bcache</code> . Possible values are <code>writethrough</code> , <code>writeback</code> , <code>writearound</code> and <code>none</code> .

4.4.9 NFS Configuration

AutoYaST allows to install openSUSE Leap onto *Network File System* (NFS) shares. To do so, you have to create a drive with the `CT_NFS` type and provide the NFS share name (`SERVER:PATH`) as device name. The information relative to the mount point is included as part of its first partition section. Note that for an NFS drive, only the first partition is taken into account.

For more information on how to configure an NFS client and server after the system has been installed, refer to *Section 4.4.9, "NFS Configuration"*.

EXAMPLE 4.20: NFS SHARE DEFINITION

```
<partitioning config:type="list">
  <drive>
    <device>192.168.1.1:/exports/root_fs</device>
    <type config:type="symbol">CT_NFS</type>
    <use>all</use>
    <partitions config:type="list">
      <partition>
        <mount>/</mount>
        <fstopt>nolock</fstopt>
      </partition>
    </partitions>
  </drive>
</partitioning>
```

4.5 iSCSI Initiator Overview

Using the `iscsi-client` resource, you can configure the target machine as an iSCSI client.

EXAMPLE 4.21: ISCSI CLIENT

```
<iscsi-client>
  <initiatorname>iqn.2013-02.de.suse:01:e229358d2dea</initiatorname>
  <targets config:type="list">
    <listentry>
      <authmethod>None</authmethod>
      <portal>192.168.1.1:3260</portal>
      <startup>onboot</startup>
      <target>iqn.2001-05.com.doe:test</target>
      <iface>default</iface>
    </listentry>
  </targets>
  <version>1.0</version>
</iscsi-client>
```

Attribute	Description
<u>initiatorname</u>	<u>InitiatorName</u> is a value from <u>/etc/iscsi/initiatorname.iscsi</u> . In case you have iBFT, this value will be added from there and you are only able to change it in the BIOS setup.
<u>version</u>	Version of the YaST module. Default: 1.0
<u>targets</u>	List of targets. Each entry contains: <u>authmethod</u> Authentication method: None/CHAP <u>portal</u> Portal address <u>startup</u> Value: manual/onboot <u>target</u> Target name <u>iface</u> Interface name

4.6 Fibre Channel over Ethernet Configuration (FCoE)

Using the fcoe_cfg resource, you can configure a Fibre Channel over Ethernet (FCoE).

EXAMPLE 4.22: FCOE CONFIGURATION

```
<fcoe-client>
  <fcoe_cfg>
    <DEBUG>no</DEBUG>
    <USE_SYSLOG>yes</USE_SYSLOG>
  </fcoe_cfg>
  <interfaces config:type="list">
    <listentry>
      <dev_name>eth3</dev_name>
      <mac_addr>01:000:000:000:42:42</mac_addr>
      <device>Gigabit 1313</device>
      <vlan_interface>200</vlan_interface>
      <fcoe_vlan>eth3.200</fcoe_vlan>
      <fcoe_enable>yes</fcoe_enable>
      <dc_b_required>yes</dc_b_required>
      <auto_vlan>no</auto_vlan>
      <dc_b_capable>no</dc_b_capable>
      <cfg_device>eth3.200</cfg_device>
    </listentry>
  </interfaces>
  <service_start>
    <fcoe config:type="boolean">true</fcoe>
    <lldpad config:type="boolean">true</lldpad>
  </service_start>
</fcoe-client>
```

Attribute	Description	Values
<u>fcoe_cfg</u>	<p><u>DEBUG</u> is used to enable or disable debugging messages from the fcoe service script and fcoemon.</p> <p><u>USE_SYSLOG</u> messages are sent to the system log if set to yes.</p>	yes/no
<u>interfaces</u>	List of network cards including the status of VLAN and FCoE configuration.	

Attribute	Description	Values
<u>service_start</u>	<p>Enable or disable the start of the services <u>fcoe</u> and <u>lldpad</u> boot time.</p> <p>Starting the <u>fcoe</u> service means starting the Fibre Channel over Ethernet service daemon <u>fcoemon</u> which controls the FCoE interfaces and establishes a connection with the <u>lldpad</u> daemon.</p> <p>The <u>lldpad</u> service provides the Link Layer Discovery Protocol agent daemon <u>lldpad</u>, which informs <u>fcoemon</u> about DCB (Data Center Bridging) features and configuration of the interfaces.</p>	yes/no

4.7 Country Settings

Language, timezone, and keyboard settings.

EXAMPLE 4.23: LANGUAGE

```
<language>
  <language>en_GB</language>
  <languages>de_DE,en_US</languages>
</language>
```

Attribute	Description	Values
<u>language</u>	Primary language	A list of available languages can be found under <u>/usr/share/YaST2/data/languages</u>
<u>languages</u>	Secondary languages separated by commas	A list of available languages can be found under <u>/usr/share/YaST2/data/languages</u>

If the configured value for the primary language is unknown, it will be reset to the default, en_US.

EXAMPLE 4.24: TIMEZONE

```
<timezone>
  <hwclock>UTC</hwclock>
  <timezone>Europe/Berlin</timezone>
</timezone>
```

Attribute	Description	Values
<u>hwclock</u>	Whether the hardware clock uses local time or UTC	localtime/UTC
<u>timezone</u>	Timezone	A list of available time zones can be found under <u>/usr/share/YaST2/data/timezone_raw.ycp</u>

EXAMPLE 4.25: KEYBOARD

```
<keyboard>
  <keymap>german</keymap>
</keyboard>
```


Attribute	Description	Values
<u>keymap</u>	Keyboard layout	Keymap-code values or keymap-alias values are valid. A list of available entries can be found in <u>/usr/share/YaST2/data/keyboards.rb</u> . For example, <u>english-us, us, english-uk, uk,...</u>

4.8 Software

4.8.1 Package Selection with Patterns and Packages Sections

Patterns or packages are configured like this:

EXAMPLE 4.26: PACKAGE SELECTION IN THE CONTROL FILE WITH PATTERNS AND PACKAGES SECTIONS

```
<software>
  <patterns config:type="list">
    <pattern>directory_server</pattern>
  </patterns>
  <packages config:type="list">
    <package>apache</package>
    <package>postfix</package>
  </packages>
  <do_online_update config:type="boolean">true</do_online_update>
</software>
```



Note: Package and Pattern Names

The values are real package or pattern names. If the package name has been changed because of an upgrade, you will need to adapt these settings too.

4.8.2 Deploying Images

You can use images during installation to speed up the installation.

EXAMPLE 4.27: ACTIVATING IMAGE DEPLOYMENT

```
<!-- note! this is not in the software section! -->
<deploy_image>
  <image_installation config:type="boolean">false</image_installation>
</deploy_image>
```

4.8.3 Installing Additional/Customized Packages or Products

In addition to the packages available for installation on the DVD-ROMs, you can add external packages including customized kernels. Customized kernel packages must be compatible to the SUSE packages and must install the kernel files to the same locations.

Unlike in earlier in versions, you do not need a special resource in the control file to install custom and external packages. Instead you need to re-create the package database and update it with any new packages or new package versions in the source repository.

A script is provided for this task which will query packages available in the repository and create the package database. Use the command `/usr/bin/create_package_descr`. It can be found in the `inst-source-utils` package in the openSUSE Build Service. When creating the database, all languages will be reset to English.

EXAMPLE 4.28: CREATING A PACKAGE DATABASE WITH THE ADDITIONAL PACKAGE INST-SOURCE-UTILS.RPM

The unpacked DVD is located in `/usr/local/DVDs/LATEST`.

```
tux > cp /tmp/inst-source-utils-2016.7.26-1.2.noarch.rpm /usr/local/DVDs/LATEST/
suse/noarch
tux > cd /usr/local/DVDs/LATEST/suse
tux > create_package_descr -d /usr/local/CDs/LATEST/suse
```

In the above example, the directory `/usr/local/CDs/LATEST/suse` contains the architecture dependent (for example `x86_64`) and architecture independent packages (`noarch`). This might look different on other architectures.

The advantage of this method is that you can keep an up-to-date repository with fixed and updated package. Additionally this method makes the creation of custom CD-ROMs easier.

To add your own module such as the SDK (SUSE Software Development Kit), add a file `add_on_products.xml` to the installation source in the root directory.

The following example shows how the SDK module can be added to the base product repository. The complete SDK repository will be stored in the directory /sdk.

EXAMPLE 4.29: `add_on_products.xml`

This file describes an SDK module included in the base product.

```
<?xml version="1.0"?>
<add_on_products xmlns="http://www.suse.com/1.0/yast2ns"
  xmlns:config="http://www.suse.com/1.0/configs">
  <product_items config:type="list">
    <product_item>
      <name>SUSE Linux Enterprise Software Development Kit</name>
      <url>relurl:///sdk?alias=SLE_SDK</url>
      <path>/</path>
      <!-- Users are asked whether to add such a product -->
      <ask_user config:type="boolean">>false</ask_user>
      <!-- Defines the default state of pre-selected state in case of ask_user
      used. -->
      <selected config:type="boolean">>true</selected>
    </product_item>
  </product_items>
</add_on_products>
```

Besides this special case, all other modules, extensions and add-on products can be added from almost every other location during an AutoYaST installation.

Even repositories which do not have any product or module information can be added during the installation. These are called other add-ons.

EXAMPLE 4.30: **ADDING THE SDK EXTENSION AND A USER DEFINED REPOSITORY**

```
<add-on>
  <add_on_products config:type="list">
    <listentry>
      <media_url>cd:///sdk</media_url>
      <product>sle-sdk</product>
      <alias>SLE SDK</alias>
      <product_dir>/</product_dir>
      <priority config:type="integer">20</priority>
      <ask_on_error config:type="boolean">>false</ask_on_error>
      <confirm_license config:type="boolean">>false</confirm_license>
      <name>SUSE Linux Enterprise Software Development Kit</name>
    </listentry>
  </add_on_products>
  <add_on_others config:type="list">
    <listentry>
```

```

    <media_url>https://download.opensuse.org/repositories/YaST:/Head/
openSUSE_Leap_15.2/</media_url>
    <alias>yast2_head</alias>
    <priority config:type="integer">30</priority>
    <name>Latest YaST2 packages from OBS</name>
  </listentry>
</add_on_others>
</add-on>

```

The add_on_others and add_on_products sections support the same values:

Attribute	Values
<u>media_url</u>	<p>Product URL. Can have the prefix <u>cd:///</u>, <u>http://</u>, <u>ftp://</u>, etc. This entry is mandatory.</p> <p>If you use a multi-product medium such as the SUSE Linux Enterprise Packages DVD, then the URL path should point to the root directory of the multi-product medium. The specific product directory is selected using the <u>product_dir</u> value (see below).</p>
<u>product</u>	<p>Internal product name if the add-on is a product. The command <u>zypper products</u> shows the names of installed products.</p>
<u>alias</u>	<p>Repository alias name. Defined by the user.</p>
<u>product_dir</u>	<p>Optional subpath. This should only be used for multi-product media such as the SUSE Linux Enterprise Packages DVD.</p>
<u>priority</u>	<p>Sets the repository libzypp priority. Priority of 1 is the highest. The higher the number, the lower the priority. Default is 99.</p>
<u>ask_on_error</u>	<p>AutoYaST can ask the user to make add-on products, modules or extensions available instead of reporting a time-out error when no</p>

Attribute	Values
	repository can be found at the given location. Set <code>ask_on_error</code> to <code>true</code> (the default is <code>false</code>).
<code>confirm_license</code>	The user needs to confirm the license. Default is <code>false</code> .
<code>name</code>	Repository name. The command <code>zypper lr</code> shows the names of added repositories.

To use unsigned installation sources with AutoYaST, turn off the checks with the following configuration in your AutoYaST control file.



Note: Unsigned Installation Sources—Limitations

You can only disable signature checking during the first stage of the auto-installation process. In stage two, the installed system's configuration takes precedence over AutoYaST configuration.

The elements listed below must be placed within the following XML structure:

```
<general>
  <signature-handling>
    ...
  </signature-handling>
</general>
```

Default values for all options are `false`. If an option is set to `false` and a package or repository fails the respective test, it is silently ignored and will not be installed. Note that setting any of these options to `true` is a potential security risk. Never do it when using packages or repositories from third party sources.

Attribute	Values
<code>accept_unsigned_file</code>	<p>If set to <code>true</code>, AutoYaST will accept unsigned files like the content file.</p> <pre><accept_unsigned_file config:type="boolean"</pre>

Attribute	Values
	<pre data-bbox="805 228 1418 275">>true</accept_unsigned_file></pre>
<u>accept_file_without_checksum</u>	<p data-bbox="805 309 1418 389">If set to <u>true</u>, AutoYaST will accept files without a checksum in the content file.</p> <pre data-bbox="805 434 1418 555"><accept_file_without_checksum config:type="boolean" >true</accept_file_without_checksum></pre>
<u>accept_verification_failed</u>	<p data-bbox="805 595 1418 721">If set to <u>true</u>, AutoYaST will accept signed files even when the verification of the signature failed.</p> <pre data-bbox="805 766 1418 887"><accept_verification_failed config:type="boolean" >true</accept_verification_failed></pre>
<u>accept_unknown_gpg_key</u>	<p data-bbox="805 927 1418 1052">If set to <u>true</u>, AutoYaST will accept new GPG keys of the installation sources, for example the key used to sign the content file.</p> <pre data-bbox="805 1097 1418 1218"><accept_unknown_gpg_key config:type="boolean" >true</accept_unknown_gpg_key></pre>
<u>accept_non_trusted_gpg_key</u>	<p data-bbox="805 1258 1418 1339">Set this option to <u>true</u> to accept known keys you have not yet trusted.</p> <pre data-bbox="805 1384 1418 1505"><accept_non_trusted_gpg_key config:type="boolean" >true</accept_non_trusted_gpg_key></pre>
<u>import_gpg_key</u>	<p data-bbox="805 1545 1418 1671">If set to <u>true</u>, AutoYaST will accept and import new GPG keys on the installation source in its database.</p> <pre data-bbox="805 1715 1418 1792"><import_gpg_key config:type="boolean" >true</import_gpg_key></pre>

It is possible to configure the signature handling for each add-on product, module, or extension individually. The following elements must be between the signature-handling section of the individual add-on product, module, or extension. All settings are optional. If not configured, the global signature-handling from the general section is used.

Attribute	Values
<u>accept_unsigned_file</u>	<p>If set to <u>true</u>, AutoYaST will accept unsigned files like the content file for this add-on product.</p> <pre data-bbox="810 645 1414 779" style="border: 1px solid #ccc; padding: 5px;"> <accept_unsigned_file config:type="boolean" >true</accept_unsigned_file></pre>
<u>accept_file_without_checksum</u>	<p>If set to <u>true</u>, AutoYaST will accept files without a checksum in the content file for this add-on.</p> <pre data-bbox="810 976 1414 1111" style="border: 1px solid #ccc; padding: 5px;"> <accept_file_without_checksum config:type="boolean" >true</accept_file_without_checksum></pre>
<u>accept_verification_failed</u>	<p>If set to <u>true</u>, AutoYaST will accept signed files even when the verification of the signature fails.</p> <pre data-bbox="810 1308 1414 1442" style="border: 1px solid #ccc; padding: 5px;"> <accept_verification_failed config:type="boolean" >true</accept_verification_failed></pre>
<u>accept_unknown_gpg_key</u>	<p>If <u>all</u> is set to <u>true</u>, AutoYaST will accept new GPG keys on the installation source.</p> <pre data-bbox="810 1592 1414 1727" style="border: 1px solid #ccc; padding: 5px;"> <accept_unknown_gpg_key> <all config:type="boolean">true</all> </accept_unknown_gpg_key></pre> <p>Otherwise you can define single keys too.</p> <pre data-bbox="810 1821 1414 1859" style="border: 1px solid #ccc; padding: 5px;"> <accept_unknown_gpg_key></pre>

Attribute	Values
	<pre data-bbox="818 228 1374 398"><all config:type="boolean">>false</all> <keys config:type="list"> <keyid>3B3011B76B9D6523</keyid> </keys> </accept_unknown_gpg_key></pre>
<p data-bbox="186 456 608 488"><u>accept_non_trusted_gpg_key</u></p>	<p data-bbox="809 456 1374 533">This means, the key is known, but it is not trusted by you.</p> <p data-bbox="809 562 1241 593">You can trust all keys by adding:</p> <pre data-bbox="818 640 1358 741"><accept_non_trusted_gpg_key> <all config:type="boolean">>true</all> </accept_non_trusted_gpg_key></pre> <p data-bbox="809 788 1206 819">Or you can trust specific keys:</p> <pre data-bbox="818 869 1374 1077"><accept_non_trusted_gpg_key> <all config:type="boolean">>false</all> <keys config:type="list"> <keyid>3B3011B76B9D6523</keyid> </keys> </accept_non_trusted_gpg_key></pre>
<p data-bbox="186 1133 416 1164"><u>import_gpg_key</u></p>	<p data-bbox="809 1133 1406 1254">If <u>all</u> is set to <u>true</u>, AutoYaST will accept and import all new GPG keys on the installation source into its database.</p> <pre data-bbox="818 1305 1358 1406"><import_gpg_key> <all config:type="boolean">>true</all> </import_gpg_key></pre> <p data-bbox="809 1453 1329 1485">This can be done for specific keys only:</p> <pre data-bbox="818 1534 1374 1742"><import_gpg_key> <all config:type="boolean">>false</all> <keys config:type="list"> <keyid>3B3011B76B9D6523</keyid> </keys> </import_gpg_key></pre>

4.8.4 Kernel Packages

Kernel packages are not part of any selection. The required kernel is determined during installation. If the kernel package is added to any selection or to the individual package selection, installation will mostly fail because of conflicts.

To force the installation of a specific kernel, use the `kernel` property. The following is an example of forcing the installation of the default kernel. This kernel will be installed even if an SMP or other kernel is required.

EXAMPLE 4.31: KERNEL SELECTION IN THE CONTROL FILE

```
<software>
  <kernel>kernel-default</kernel>
  ...
</software>
```

4.8.5 Removing Automatically Selected Packages

Some packages are selected automatically either because of a dependency or because it is available in a selection.

Removing these packages might break the system consistency, and it is not recommended to remove basic packages unless a replacement which provides the same services is provided. The best example for this case are mail transfer agent (MTA) packages. By default, `postfix` will be selected and installed. To use another MTA like `sendmail`, then postfix can be removed from the list of selected package using a list in the software resource. However, note that sendmail is not shipped with openSUSE Leap. The following example shows how this can be done:

EXAMPLE 4.32: PACKAGE SELECTION IN CONTROL FILE

```
<software>
  <packages config:type="list">
    <package>sendmail</package>
  </packages>
  <remove-packages config:type="list">
    <package>postfix</package>
  </remove-packages>
</software>
```



Note: Package Removal Failure

Note that it is not possible to remove a package that is part of a pattern (see [Section 4.8.1, “Package Selection with Patterns and Packages Sections”](#)). When specifying such a package for removal, the installation will fail with the following error message:

```
The package resolver run failed. Check
your software section in the AutoYaST profile.
```

4.8.6 Installing Recommended Packages/Patterns

By default all recommended packages/patterns will be installed. To have a minimal installation which includes required packages only, you can switch off this behavior with the flag `install_recommended`. Note that this flag only affects a fresh installation and will be ignored during an upgrade.

```
<software>
  <install_recommended config:type="boolean">>false
</install_recommended>
</software>
```

Default: If this flag has not been set in the configuration file *all* recommended `packages` and *no* recommended `pattern` will be installed.

4.8.7 Installing Packages in Stage 2

To install packages after the reboot during stage two, you can use the `post-packages` element for that:

```
<software>
  <post-packages config:type="list">
    <package>yast2-cim</package>
  </post-packages>
</software>
```

4.8.8 Installing Patterns in Stage 2

You can also install patterns in stage 2. Use the `post-patterns` element for that:

```
<software>
  <post-patterns config:type="list">
    <pattern>apparmor</pattern>
  </post-patterns>
</software>
```

4.8.9 Online Update in Stage 2

You can perform an online update at the end of the installation. Set the boolean `do_online_update` to `true`. Of course this only makes sense if you add an online update repository in the `suse-register/customer-center` section, for example, or in a post-script. If the online update repository was already available in stage one via the add-on section, then AutoYaST has already installed the latest packages available. If a kernel update is done via `online-update`, a reboot at the end of stage two is triggered.

```
<software>
  <do_online_update config:type="boolean">true</do_online_update>
</software>
```

4.9 Upgrade

AutoYaST can also be used for doing a system upgrade. Besides upgrade packages, the following sections are supported too:

- `scripts/pre-scripts` Running user scripts very early, before anything else really happens.
- `add-on` Defining an additional add-on product.
- `language` Setting language.
- `timezone` Setting timezone.
- `keyboard` Setting keyboard.
- `software` Installing additional software/patterns. Removing installed packages.
- `suse_register` Running registration process.

To control the upgrade process the following sections can be defined:

EXAMPLE 4.33: UPGRADE AND BACKUP

```
<upgrade>
  <stop_on_solver_conflict config:type="boolean">true</stop_on_solver_conflict>
</upgrade>
<backup>
  <sysconfig config:type="boolean">true</sysconfig>
  <modified config:type="boolean">true</modified>
  <remove_old config:type="boolean">true</remove_old>
</backup>
```

Element	Description	Comment
<u>stop_on_solver_conflict</u>	Halt installation if there are package dependency issues.	
<u>modified</u>	Create backup of modified files.	
<u>sysconfig</u>	Create backup of <u>/etc/sysconfig</u> directory.	
<u>remove_old</u>	Remove backups from previous updates.	

To start the AutoYaST upgrade mode, you need:

PROCEDURE 4.1: STARTING AUTOYAST IN OFFLINE UPGRADE MODE

1. Copy the AutoYaST profile to /root/autoupg.xml on its file system.
2. Boot the system from the installation medium.
3. Select the Installation menu item.
4. On the command line, set autoupgrade=1.
5. Press to start the upgrade process.

PROCEDURE 4.2: STARTING AUTOYAST IN ONLINE UPGRADE MODE

1. Boot the system from the installation media.

2. Select the Installation menu item.
3. On the command line, set netsetup=dhcp autoupgrade=1 autoyast=http://192.169.3.1/autoyast.xml.
Here network will be setup via DHCP.
4. Press to start the upgrade process.

4.10 Services and Targets

With the services-manager resource you can set the default systemd target and specify in detail which system services you want to start or deactivate and how to start them.

The default-target property specifies the default systemd target into which the system boots. Valid options are graphical for a graphical login, or multi-user for a console login.

To specify the set of services that should be started on boot, use the enable and disable lists. To start a service, add its name to the enable list. To make sure that the service is not started on boot, add it to the disable list.

If a service is not listed as enabled or disabled, a default setting is used. The default setting may be either disabled or enabled.

Finally, some services like cups support on-demand activation (socket activated services). If you want to take advantage of such a feature, list the names of those services in the on_demand list instead of enable.

EXAMPLE 4.34: CONFIGURING SERVICES AND TARGETS

```
<services-manager>
  <default_target>multi-user</default_target>
  <services>
    <disable config:type="list">
      <service>libvirtd</service>
    </disable>
    <enable config:type="list">
      <service>sshd</service>
    </enable>
    <on_demand config:type="list">
      <service>cups</service>
    </on_demand>
  </services>
</services-manager>
```

4.11 Network Configuration

Network configuration is used to connect a single workstation to an Ethernet-based LAN or to configure a dial-up connection. More complex configurations (multiple network cards, routing, etc.) are also provided.

If the following setting is set to `true`, YaST will keep network settings created during the installation (via `linuxrc`) and/or merge it with network settings from the AutoYaST control file (if defined).



Note: The `linuxrc` program

For a detailed description of how `linuxrc` works and its keywords, see [Appendix C, Advanced `linuxrc` Options](#).

AutoYaST settings have higher priority than any configuration files already present. YaST will write `ifcfg-*` files based on the entries in the control file without removing old ones. If there is an empty or absent DNS and routing section, YaST will keep any pre-existing values. Otherwise, settings from the control file will be applied.

```
<keep_install_network
config:type="boolean">true</keep_install_network>
```

During the second stage, installation of additional packages will take place before the network, as described in the profile, is configured. `keep_install_network` is set by default to `true` to ensure that a network is available in case it is needed to install those packages. If all packages are installed during the first stage and the network is not needed early during the second one, setting `keep_install_network` to `false` will avoid copying the configuration.

To configure network settings and activate networking automatically, one global resource, `<networking>` is used to store the whole network configuration.

EXAMPLE 4.35: NETWORK CONFIGURATION

```
<networking>
<dns>
  <dhcp_hostname config:type="boolean">true</dhcp_hostname>
  <domain>site</domain>
  <hostname>linux-bqua</hostname>
  <nameservers config:type="list">
    <nameserver>&dnsip;</nameserver>
    <nameserver>&dnsip117;</nameserver>
    <nameserver>&dnsip118;</nameserver>
```

```

</nameservers>
<resolv_conf_policy>auto</resolv_conf_policy>
<searchlist config:type="list">
  <search>&exampledomain;</search>
  <search>&exampledomain1;</search>
</searchlist>
</dns>
<interfaces config:type="list">
  <interface>
    <bootproto>dhcp</bootproto>
    <device>eth0</device>
    <startmode>auto</startmode>
  </interface>
</interfaces>
<ipv6 config:type="boolean">true</ipv6>
<keep_install_network config:type="boolean">>false</keep_install_network>
#false means use Wicked, true means use NetworkManager
<managed config:type="boolean">>false</managed>
<net-udev config:type="list">
  <rule>
    <name>eth0</name>
    <rule>ATTR{address}</rule>
    <value>&wsImac;</value>
  </rule>
</net-udev>
<s390-devices config:type="list">
  <listentry>
    <chanids>0.0.0800:0.0.0801:0.0.0802</chanids>
    <type>qeth</type>
  </listentry>
</s390-devices>
<routing>
  <ipv4_forward config:type="boolean">>false</ipv4_forward>
  <ipv6_forward config:type="boolean">>false</ipv6_forward>
  <routes config:type="list">
    <route>
      <destination>&routerintipII;</destination>
      <device>eth0</device>
      <extrapara>foo</extrapara>
      <gateway>-</gateway>
      <netmask>-</netmask>
    </route>
    <route>
      <destination>default</destination>
      <device>eth0</device>
      <gateway>&routerintipI;</gateway>
      <netmask>-</netmask>
    </route>
  </routes>
</routing>

```

```

    </route>
    <route>
      <destination>default</destination>
      <device>lo</device>
      <gateway>&gateip;</gateway>
      <netmask>-</netmask>
    </route>
  </routes>
</routing>
</networking>

```

As shown in the example above, the `<networking>` section can be composed of a few subsections:

- `interfaces` describes the configuration of the network interfaces, including their IP addresses, how they are started, etc.
- `dns` specifies DNS related settings, as the host name, the list of name servers, etc.
- `routing` defines the routing rules.
- `s390-devices` s390-specific devices settings.
- `net-udev` enumerates the udev rules used to set persistent names.

EXAMPLE 4.36: BRIDGE INTERFACE CONFIGURATION

```

<interfaces config:type="list">
  <interface>
    <device>br0</device>
    <bootproto>static</bootproto>
    <bridge>yes</bridge>
    <bridge_forwarddelay>0</bridge_forwarddelay>
    <bridge_ports>eth0 eth1</bridge_ports>
    <bridge_stp>off</bridge_stp>
    <ipaddr>192.168.122.100</ipaddr>
    <netmask>255.255.255.0</netmask>
    <network>192.168.122.0</network>
    <prefixlen>24</prefixlen>
    <startmode>auto</startmode>
  </interface>
  <interface>
    <device>eth0</device>
    <bootproto>none</bootproto>
    <startmode>hotplug</startmode>
  </interface>

```



```

<interface>
  <device>eth1</device>
  <bootproto>none</bootproto>
  <startmode>hotplug</startmode>
</interface>
</interfaces>

```



Tip: IPv6 Address Support

Using IPv6 addresses in AutoYaST is fully supported. To disable IPv6 Address Support, set `<ipv6 config:type="boolean">false</ipv6>`

4.11.1 Interfaces

The `interfaces` section allows the user to define the interfaces configuration, including how they are started, their IP addresses, networks, and more. The following elements must be enclosed in `<interfaces>...</interfaces>` tags.

Element	Description	Comment
<u>bootproto</u>	<p>Boot protocol used by the interface. Possible values:</p> <ul style="list-style-type: none"> • <u>static</u> for statically assigned addresses. It is required to specify the IP using the <u>ipaddr</u> element. • <u>dhcp4</u>, <u>dhcp6</u> or <u>dhcp</u> for setting the IP address with DHCP (IPv4, IPv6 or any). • <u>dhcp+autoip</u> to get the IPv4 configuration from <i>Zeroconf</i> and get IPv6 from DHCP. 	Required.

Element	Description	Comment
	<ul style="list-style-type: none"> • <u>autoip</u> to get the IPv4 configuration from <i>Zeroconf</i>. • <u>ibft</u> to get the IP address using the iBFT protocol. • <u>none</u> to skip setting an address. This value is used for bridges and bonding slaves. 	
<u>broadcast</u>	Broadcast IP address.	Used only with <u>static</u> boot protocol.
<u>device</u>	Device name.	Deprecated. Use <u>name</u> instead.
<u>name</u>	Device name, for example: <u>eth0</u> .	Required.
<u>ipaddr</u>	IP address assigned to the interface.	Used only with <u>static</u> boot protocol. It can include a network prefix, for example: <u>192.168.1.1/24</u> .
<u>remote_ipaddr</u>	Remote IP address for Point-to-Point connections.	Used only with <u>static</u> boot protocol.
<u>netmask</u>	Network mask, for example: <u>255.255.255.0</u> .	Deprecated. Use <u>prefixlen</u> instead or include the network prefix in the <u>ipaddr</u> element.
<u>network</u>	Network IP address.	Deprecated. Use <u>ipaddr</u> with <u>prefixlen</u> instead.

Element	Description	Comment
<u>prefixlen</u>	Network prefix, for example: <u>24</u> .	Used only with <u>static</u> boot protocol.
<u>startmode</u>	<p>When to bring up an interface. Possible values are:</p> <ul style="list-style-type: none"> • <u>hotplug</u> when the device is plugged in. Useful for USB network cards, for example. • <u>auto</u> when the system boots. <u>onboot</u> is a deprecated alias. • <u>ifplugd</u> when the device is managed by the <u>ifplugd</u> daemon. • <u>manual</u> when the device is supposed to be started manually. • <u>nfsroot</u> when the device is needed to mount the root file system, for example, when <u>/</u> is on a NFS volume. • <u>off</u> to never start the device. 	
<u>ifplugd_priority</u>	Priority for <u>ifplugd</u> daemon. It determines in which order the devices are activated.	Used only with <u>ifplugd</u> start mode.

Element	Description	Comment
<u>usercontrol</u>	Parameter is no longer used.	Deprecated.
<u>bonding_slaveX</u>	Name of the bonding device.	Required for bonding devices. <u>X</u> is replaced by a number starting from 0, for example <u>bonding_slave0</u> . Each slave needs to have a unique number.
<u>bonding_module_opts</u>	Options for bonding device.	Used only with <u>bond</u> device.
<u>mtu</u>	Maximum transmission unit for the interface.	Optional.
<u>ethtool_options</u>	Ethtool options during device activation.	Optional.
<u>zone</u>	Firewall zone name which the interface is assigned to.	Optional.
<u>vlan_id</u>	Identifier used for this VLAN.	Used only with <u>vlan</u> device.
<u>etherdevice</u>	Device to which VLAN is attached.	Used only with a <u>vlan</u> device and required for it.
<u>bridge</u>	<u>yes</u> if interface is a bridge.	Deprecated. It is inferred from other attributes.
<u>bridge_ports</u>	Space-separated list of bridge ports, for example, <u>eth0 eth1</u> .	Used only with <u>bridge</u> device and required for it.
<u>bridge_stp</u>	Spanning tree protocol. Possible values are <u>on</u> (when enabled) and <u>off</u> (when disabled).	Used only with a <u>bridge</u> device.

Element	Description	Comment
<u>bridge_forward_delay</u>	Forward delay for bridge, for example: <u>15</u> .	Used only with <u>bridge</u> devices. Valid values are between <u>4</u> and <u>30</u> .

4.11.2 Persistent Names of Network Interfaces

The `net-udev` element allows to specify a set of udev rules that can be used to assign persistent names to interfaces.

Element	Description	Comment
<u>name</u>	Network interface name, for example <u>eth3</u> .	Required.
<u>rule</u>	<u>ATTR{address}</u> for a MAC based rule, <u>KERNELS</u> for a bus ID based rule.	required
<u>value</u>	for example <u>f0:de:f1:6b:da:69</u> for a MAC rule, <u>0000:00:1c.1</u> or <u>0.0.0700</u> for a bus ID rule	Required.



Tip: Handling Collisions in Device Names

When creating an incomplete *udev* rule set, the chosen device name can collide with existing device names. For example, when renaming a network interface to eth0, a collision with a device automatically generated by the kernel can occur. AutoYaST tries to handle such cases in a best effort manner and renames colliding devices.

EXAMPLE 4.37: ASSIGNING A PERSISTENT NAME USING THE MAC ADDRESS

```
<net-udev config:type="list">
  <rule>
    <name>eth1</name>
    <rule>ATTR{address}</rule>
    <value>52:54:00:68:54:fb</value>
```

```
</rule>
</net-udev>
```

4.11.3 Domain Name System

The `dns` section is used to define name-service related settings, such as the host name, or name servers.

Element	Description	Comment
<u>hostname</u>	Host name, excluding the domain name part. For example: <code>foo</code> (instead of <code>foo.bar</code>).	If a host name is not specified and is not taken from a DHCP server (see <code>dhcp_hostname</code>), AutoYaST will generate a random one.
<u>nameservers</u>	List of name servers. Example: <pre><nameservers config:type="list"> <nameserver>192.168.1.116</nameserver> <nameserver>192.168.1.117</nameserver> </nameservers></pre>	
<u>searchlist</u>	Search list for host name lookup. <pre><searchlist config:type="list"> <search>example.com</search> </searchlist></pre>	Optional.
<u>dhcp_hostname</u>	Specifies whether the host name must be taken from DHCP or not.	

Element	Description	Comment
	<pre><dhcp_hostname config:type="boolean">true</ dhcp_hostname></pre>	

4.11.4 Routing

The `routing` table allows to specify a list of routes and the packets forwarding settings for IPv4 and IPv6.

Element	Description	Comment
<code>ipv4_forward</code>	Whether IP forwarding must be enabled for IPv4.	Optional.
<code>ipv6_forward</code>	Whether IP forwarding must be enabled for IPv6.	Optional.
<code>routes</code>	List of routes.	Optional.

The following table describes how routes are defined.

Element	Description	Comment
<code>destination</code>	Route destination. An address prefix can be specified, for example: <code>192.168.122.0/24</code> .	Required.
<code>device</code>	Interface associated to the route.	Required.
<code>gateway</code>	Gateway's IP address.	Optional.
<code>netmask</code>	Destination's netmask.	Deprecated. Specifying the prefix as part of the <code>destination</code> value is preferred.

4.11.5 s390 Options

The following elements must be between the `<s390-devices>...</s390-devices>` tags.

Element	Description	Comment
type	qeth, ctc or iucv	
chanids	channel ids separated by a colon (preferred) or a space <pre><chanids>0.0.0700:0.0.0701:0.0.0702</chanids></pre>	
layer2	<pre><layer2 config:type="boolean">true</layer2></pre>	boolean; default: false
portname	QETH port name (deprecated since openSUSE 42.2)	
protocol	CTC / LCS protocol, a small number (as a string) <pre><protocol>1</protocol></pre>	optional
router	IUCV router/user	

In addition to the options mentioned above, AutoYaST also supports IBM Z-specific options in other sections of the configuration file. In particular, you can define the logical link address, or LLADDR (in the case of Ethernet, that is the MAC address). To do so, use the option LLADDR in the device definition.



Tip: LLADDR for VLANs

VLAN devices inherit their LLADDR from the underlying physical devices. To set a particular address for a VLAN device, set the LLADDR option for the underlying physical device.

4.11.6 Proxy

Configure your Internet proxy (caching) settings.

Configure proxies for HTTP, HTTPS, and FTP with `http_proxy`, `https_proxy` and `ftp_proxy`, respectively. Addresses or names that should be directly accessible need to be specified with `no_proxy` (space separated values). If you are using a proxy server with authorization, fill in `proxy_user` and `proxy_password`,

EXAMPLE 4.38: NETWORK CONFIGURATION: PROXY

```
<proxy>
  <enabled config:type="boolean">true</enabled>
  <ftp_proxy>http://192.168.1.240:3128</ftp_proxy>
  <http_proxy>http://192.168.1.240:3128</http_proxy>
  <no_proxy>www.example.com .example.org localhost</no_proxy>
  <proxy_password>testpw</proxy_password>
  <proxy_user>testuser</proxy_user>
</proxy>
```

4.12 NIS Client and Server

Using the `nis` resource, you can configure the target machine as a NIS client. The following example shows a detailed configuration using multiple domains.

EXAMPLE 4.39: NETWORK CONFIGURATION: NIS

```
<nis>
  <nis_broadcast config:type="boolean">true</nis_broadcast>
  <nis_broken_server config:type="boolean">true</nis_broken_server>
  <nis_by_dhcp config:type="boolean">false</nis_by_dhcp>
  <nis_domain>test.com</nis_domain>
  <nis_local_only config:type="boolean">true</nis_local_only>
  <nis_options></nis_options>
  <nis_other_domains config:type="list">
    <nis_other_domain>
      <nis_broadcast config:type="boolean">false</nis_broadcast>
      <nis_domain>domain.com</nis_domain>
      <nis_servers config:type="list">
        <nis_server>10.10.0.1</nis_server>
      </nis_servers>
    </nis_other_domain>
  </nis_other_domains>
```

```

<nis_servers config:type="list">
  <nis_server>192.168.1.1</nis_server>
</nis_servers>
<start_autofs config:type="boolean">>true</start_autofs>
<start_nis config:type="boolean">>true</start_nis>
</nis>

```

4.13 NIS Serve

You can configure the target machine as a NIS server. NIS Master Server and NIS Slave Server and a combination of both are available.

EXAMPLE 4.40: NIS SERVER CONFIGURATION

```

<nis_server>
  <domain>mydomain.de</domain>
  <maps_to_serve config:type="list">
    <nis_map>auto.master</nis_map>
    <nis_map>ethers</nis_map>
  </maps_to_serve>
  <merge_passwd config:type="boolean">>false</merge_passwd>
  <mingid config:type="integer">0</mingid>
  <minuid config:type="integer">0</minuid>
  <nopush config:type="boolean">>false</nopush>
  <pwd_chfn config:type="boolean">>false</pwd_chfn>
  <pwd_chsh config:type="boolean">>false</pwd_chsh>
  <pwd_srcdir>/etc</pwd_srcdir>
  <securenets config:type="list">
    <securenet>
      <netmask>255.0.0.0</netmask>
      <network>127.0.0.0</network>
    </securenet>
  </securenets>
  <server_type>master</server_type>
  <slaves config:type="list"/>
  <start_ybind config:type="boolean">>false</start_ybind>
  <start_yppasswdd config:type="boolean">>false</start_yppasswdd>
  <start_ypxfrd config:type="boolean">>false</start_ypxfrd>
</nis_server>

```

Attribute	Values	Description
<u>domain</u>	NIS domain name.	

Attribute	Values	Description
<u>maps_to_serve</u>	List of maps which are available for the server.	Values: auto.master, ethers, group, hosts, netgrp, networks, passwd, protocols, rpc, services, shadow
<u>merge_passwd</u>	Select if your passwd file should be merged with the shadow file (only possible if the shadow file exists).	Value: true/false
<u>mingid</u>	Minimum GID to include in the user maps.	
<u>minuid</u>	Minimum UID to include in the user maps.	
<u>nopush</u>	Do not push the changes to slave servers. (Useful if there are none).	Value: true/false
<u>pwd_chfn</u>	YPPWD_CHFN - allow changing the full name	Value: true/false
<u>pwd_chsh</u>	YPPWD_CHSH - allow changing the login shell	Value: true/false
<u>pwd_srcdir</u>	YPPWD_SRCDIR - source directory for passwd data	Default: <u>/etc</u>
<u>securenets</u>	List of allowed hosts to query the NIS server	A host address will be allowed if network is equal to the bitwise AND of the host's address and the netmask.

Attribute	Values	Description
		The entry with netmask 255.0.0.0 and network 127.0.0.0 must exist to allow connections from the local host. Entering netmask 0.0.0.0 and network 0.0.0.0 gives access to all hosts.
<u>server_type</u>	Select whether to configure the NIS server as a master or a slave or not to configure a NIS server.	Values: master, slave, none
<u>slaves</u>	List of host names to configure as NIS server slaves.	
<u>start_yplib</u>	This host is also a NIS client (only when client is configured locally).	Value: true/false
<u>start_yppasswdd</u>	Also start the password daemon.	Value: true/false
<u>start_ypxfrd</u>	Also start the map transfer daemon. Fast Map distribution; it will speed up the transfer of maps to the slaves.	Value: true/false

4.14 Hosts Definition

Using the host resource, you can add more entries to the /etc/hosts file. Already existing entries will not be deleted. The following example shows details.

EXAMPLE 4.41: /ETC/HOSTS

```
<host>
  <hosts config:type="list">
    <hosts_entry>
      <host_address>133.3.0.1</host_address>
      <names config:type="list">
        <name>booking</name>
      </names>
    </hosts_entry>
    <hosts_entry>
      <host_address>133.3.0.5</host_address>
      <names config:type="list">
        <name>test-machine</name>
      </names>
    </hosts_entry>
  </hosts>
</host>
```

4.15 Windows Domain Membership

Using the samba-client resource, you can configure membership of a workgroup, NT domain, or Active Directory domain.

EXAMPLE 4.42: SAMBA CLIENT CONFIGURATION

```
<samba-client>
  <disable_dhcp_hostname config:type="boolean">>true</disable_dhcp_hostname>
  <global>
    <security>domain</security>
    <usershare_allow_guests>No</usershare_allow_guests>
    <usershare_max_shares>100</usershare_max_shares>
    <workgroup>WORKGROUP</workgroup>
  </global>
  <winbind config:type="boolean">>false</winbind>
</samba-client>
```

Attribute	Values	Description
<u>disable_dhcp_hostname</u>	Do not allow DHCP to change the host name.	Value: true/false

Attribute	Values	Description
<u>global/security</u>	Kind of authentication regime (domain technology or Active Directory server (ADS)).	Value: ADS/domain
<u>global/usershare_allow_guests</u>	Sharing guest access is allowed.	Value: No/Yes
<u>global/usershare_max_shares</u>	Max. number of shares from <u>smb.conf</u> .	0 means that shares are not enabled.
<u>global/workgroup</u>	Workgroup or domain name.	
<u>winbind</u>	Using winbind.	Value: true/false

4.16 Samba Server

Configuration of a simple Samba server.

EXAMPLE 4.43: SAMBA SERVER CONFIGURATION

```
<samba-server>
  <accounts config:type="list"/>
  <backend/>
  <config config:type="list">
    <listentry>
      <name>global</name>
      <parameters>
        <security>domain</security>
        <usershare_allow_guests>No</usershare_allow_guests>
        <usershare_max_shares>100</usershare_max_shares>
        <workgroup>WORKGROUP</workgroup>
      </parameters>
    </listentry>
  </config>
  <service>Disabled</service>
  <trustdom/>
  <version>2.11</version>
</samba-server>
```

Attribute	Values	Description
<u>accounts</u>	List of Samba accounts.	
<u>backend</u>	List of available back-ends	Value: true/false
<u>config</u>	Setting additional user-defined parameters in <u>/etc/samba/smb.conf</u> .	The example shows parameters in the <u>global</u> section of <u>/etc/samba/smb.conf</u> .
<u>service</u>	Samba service starts during boot.	Value: Enabled/Disabled
<u>trustdom/</u>	Trusted Domains.	A map of two maps (keys: <u>establish</u> , <u>revoke</u>). Each map contains entries in the format <u>key: domainname</u> <u>value: password</u> .
<u>version</u>	Samba version.	Default: 2.11

4.17 Authentication Client

The configuration file must be in the JSON format. Verify that both autoyast2 and autoyast2-installation are installed. Use the *Autoinstallation Configuration* module in YaST to generate a valid JSON configuration file. Launch YaST and switch to the *Miscellaneous > Autoinstallation Configuration*. Choose *Network Services > User Logon Management*, click *Edit*, and configure the available settings. Click *OK* when done. To save the generated configuration file, use the *File > Save*.



Tip: Using ldaps://

To use LDAP with native SSL (rather than TLS), add the ldaps resource.

4.18 NFS Client and Server

Configuring a system as an NFS client or an NFS server can be done using the configuration system. The following examples show how both NFS client and server can be configured.

From openSUSE Leap 15.2 on, the structure of NFS client configuration has changed. Some global configuration options were introduced: `enable_nfs4` to switch NFS4 support on/off and `idmapd_domain` to define domain name for `rpc.idmapd` (this only makes sense when NFS4 is enabled). Attention: the old structure is not compatible with the new one and the control files with an NFS section created on older releases will not work with newer products.

For more information on how to install openSUSE Leap onto NFS shares, refer to [Section 4.4.9, "NFS Configuration"](#).

EXAMPLE 4.44: NETWORK CONFIGURATION: NFS CLIENT

```
<nfs>
  <enable_nfs4 config:type="boolean">true</enable_nfs4>
  <idmapd_domain>suse.cz</idmapd_domain>
  <nfs_entries config:type="list">
    <nfs_entry>
      <mount_point>/home</mount_point>
      <nfs_options>sec=krb5i,intr,rw</nfs_options>
      <server_path>saurus.suse.cz:/home</server_path>
      <vfstype>nfs4</vfstype>
    </nfs_entry>
    <nfs_entry>
      <mount_point>/work</mount_point>
      <nfs_options>defaults</nfs_options>
      <server_path>bivoj.suse.cz:/work</server_path>
      <vfstype>nfs</vfstype>
    </nfs_entry>
    <nfs_entry>
      <mount_point>/mnt</mount_point>
      <nfs_options>defaults</nfs_options>
      <server_path>fallback.suse.cz:/srv/dist</server_path>
      <vfstype>nfs</vfstype>
    </nfs_entry>
  </nfs_entries>
</nfs>
```

EXAMPLE 4.45: NETWORK CONFIGURATION: NFS SERVER

```
<nfs_server>
  <nfs_exports config:type="list">
    <nfs_export>
      <allowed config:type="list">
```



```

    <allowed_clients>*(ro,root_squash,sync)</allowed_clients>
  </allowed>
  <mountpoint>/home</mountpoint>
</nfs_export>
<nfs_export>
  <allowed config:type="list">
    <allowed_clients>*(ro,root_squash,sync)</allowed_clients>
  </allowed>
  <mountpoint>/work</mountpoint>
</nfs_export>
</nfs_exports>
<start_nfsserver config:type="boolean">>true</start_nfsserver>
</nfs_server>

```

4.19 NTP Client

! Important: NTP Client Profile Incompatible

Starting with openSUSE Leap 15, the NTP client profile has a new format and is *not* compatible with previous profiles. You need to update your NTP client profile used in prior openSUSE Leap versions to be compatible with version 15 and newer.

Following is an example of the NTP client configuration:

EXAMPLE 4.46: NETWORK CONFIGURATION: NTP CLIENT

```

<ntp-client>
<ntp_policy>auto</ntp_policy> ❶
<ntp_servers config:type="list">
  <ntp_server>
    <address>cz.pool.ntp.org</address> ❷
    <iburst config:type="boolean">>false</iburst> ❸
    <offline config:type="boolean">>false</offline> ❹
  </ntp_server>
</ntp_servers>
<ntp_sync>15</ntp_sync> ❺
</ntp-client>

```

- ❶ The `ntp_policy` takes the same values as the `NETCONFIG_NTP_POLICY` option in `/etc/sysconfig/network/config`. The most common options are 'static' and 'auto' (default). See [man 8 netconfig](#) for more details.
- ❷ URL of the time server or pool of time servers.

- ③ `iburst` speeds up the initial time synchronization for the specific time source after `chronyd` is started.
- ④ When the `offline` option is set to `true` it will prevent the client from polling the time server if it is not available when `chronyd` is started. Polling will not resume until it is started manually with `chronyc online`. This command does not survive a reboot. Setting it to `false` ensures that clients will always attempt to contact the time server, without administrator intervention.
- ⑤ For `ntp_sync`, enter 'systemd' (default) when running an NTP daemon, an *integer* interval in seconds to synchronize using cron, or 'manual' for no automatic synchronization.

The following example illustrates an IPv6 configuration. You may use the server's IP address, host name, or both:

```
<peer>
  <address>2001:418:3ff::1:53</address>
  <comment/>
  <options/>
  <type>server</type>
</peer>

<peer>
  <address>2.pool.ntp.org</address>
  <comment/>
  <options/>
  <type>server</type>
</peer>
```

4.20 Mail Server Configuration

For the mail configuration of the client, this module lets you create a detailed mail configuration. The module contains various options. We recommended you use it at least for the initial configuration.

EXAMPLE 4.47: MAIL CONFIGURATION

```
<mail>
  <aliases config:type="list">
    <alias>
      <alias>root</alias>
      <comment></comment>
      <destinations>foo</destinations>
```

```

</alias>
<alias>
  <alias>test</alias>
  <comment></comment>
  <destinations>foo</destinations>
</alias>
</aliases>
<connection_type config:type="symbol">permanent</connection_type>
<fetchmail config:type="list">
  <fetchmail_entry>
    <local_user>foo</local_user>
    <password>bar</password>
    <protocol>POP3</protocol>
    <remote_user>foo</remote_user>
    <server>pop.foo.com</server>
  </fetchmail_entry>
  <fetchmail_entry>
    <local_user>test</local_user>
    <password>bar</password>
    <protocol>IMAP</protocol>
    <remote_user>test</remote_user>
    <server>blah.com</server>
  </fetchmail_entry>
</fetchmail>
<from_header>test.com</from_header>
<listen_remote config:type="boolean">>true</listen_remote>
<local_domains config:type="list">
  <domains>test1.com</domains>
</local_domains>
<masquerade_other_domains config:type="list">
  <domain>blah.com</domain>
</masquerade_other_domains>
<masquerade_users config:type="list">
  <masquerade_user>
    <address>joe@test.com</address>
    <comment></comment>
    <user>joeuser</user>
  </masquerade_user>
  <masquerade_user>
    <address>bar@test.com</address>
    <comment></comment>
    <user>foo</user>
  </masquerade_user>
</masquerade_users>
<mta config:type="symbol">postfix</mta>
<outgoing_mail_server>test.com</outgoing_mail_server>
<postfix_mda config:type="symbol">local</postfix_mda>

```

```

<smtp_auth config:type="list">
  <listentry>
    <password>bar</password>
    <server>test.com</server>
    <user>foo</user>
  </listentry>
</smtp_auth>
<use_amavis config:type="boolean">>true</use_amavis>
<virtual_users config:type="list">
  <virtual_user>
    <alias>test.com</alias>
    <comment></comment>
    <destinations>foo.com</destinations>
  </virtual_user>
  <virtual_user>
    <alias>geek.com</alias>
    <comment></comment>
    <destinations>bar.com</destinations>
  </virtual_user>
</virtual_users>
</mail>

```

4.21 Apache HTTP Server Configuration

This section is used for configuration of an Apache HTTP server.

For less experienced users, we would suggest to configure the Apache server using the [HTTP server](#) YaST module. After that, call the [AutoYaST configuration](#) module, select the [HTTP server](#) YaST module and clone the Apache settings. These settings can be exported via the menu [File](#).

EXAMPLE 4.48: HTTP SERVER CONFIGURATION

```

<http-server>
  <Listen config:type="list">
    <listentry>
      <ADDRESS/>
      <PORT>80</PORT>
    </listentry>
  </Listen>
  <hosts config:type="list">
    <hosts_entry>
      <KEY>main</KEY>
      <VALUE config:type="list">
        <listentry>

```

```

<KEY>DocumentRoot</KEY>
<OVERHEAD>
#
# Global configuration that will be applicable for all
# virtual hosts, unless deleted here or overridden elsewhere.
#
</OVERHEAD>
<VALUE>"/srv/www/htdocs"</VALUE>
</listentry>
<listentry>
<KEY>_SECTION</KEY>
<OVERHEAD>
#
# Configure the DocumentRoot
#
</OVERHEAD>
<SECTIONNAME>Directory</SECTIONNAME>
<SECTIONPARAM>"/srv/www/htdocs"</SECTIONPARAM>
<VALUE config:type="list">
<listentry>
<KEY>Options</KEY>
<OVERHEAD>
# Possible values for the Options directive are "None", "All",
# or any combination of:
#   Indexes Includes FollowSymLinks SymLinksifOwnerMatch
#   ExecCGI MultiViews
#
# Note that "MultiViews" must be named *explicitly*
# --- "Options All"
# does not give it to you.
#
# The Options directive is both complicated and important.
# Please see
# http://httpd.apache.org/docs/2.4/mod/core.html#options
# for more information.
</OVERHEAD>
<VALUE>None</VALUE>
</listentry>
<listentry>
<KEY>AllowOverride</KEY>
<OVERHEAD>
# AllowOverride controls what directives may be placed in
# .htaccess files. It can be "All", "None", or any combination
# of the keywords:
#   Options FileInfo AuthConfig Limit
</OVERHEAD>
<VALUE>None</VALUE>

```

```

</listentry>
<listentry>
  <KEY>_SECTION</KEY>
  <OVERHEAD>
  # Controls who can get stuff from this server.
  </OVERHEAD>
  <SECTIONNAME>IfModule</SECTIONNAME>
  <SECTIONPARAM>!mod_access_compat.c</SECTIONPARAM>
  <VALUE config:type="list">
    <listentry>
      <KEY>Require</KEY>
      <VALUE>all granted</VALUE>
    </listentry>
  </VALUE>
</listentry>
<listentry>
  <KEY>_SECTION</KEY>
  <SECTIONNAME>IfModule</SECTIONNAME>
  <SECTIONPARAM>mod_access_compat.c</SECTIONPARAM>
  <VALUE config:type="list">
    <listentry>
      <KEY>Order</KEY>
      <VALUE>allow,deny</VALUE>
    </listentry>
    <listentry>
      <KEY>Allow</KEY>
      <VALUE>from all</VALUE>
    </listentry>
  </VALUE>
</listentry>
</VALUE>
</listentry>
<listentry>
  <KEY>Alias</KEY>
  <OVERHEAD>
  # Aliases: aliases can be added as needed (with no limit).
  # The format is Alias fakename realname
  #
  # Note that if you include a trailing / on fakename then the
  # server will require it to be present in the URL. So "/icons"
  # is not aliased in this example, only "/icons/". If the fakename
  # is slash-terminated, then the realname must also be slash
  # terminated, and if the fakename omits the trailing slash, the
  # realname must also omit it.
  # We include the /icons/ alias for FancyIndexed directory listings.
  # If you do not use FancyIndexing, you may comment this out.
  #

```

```

</OVERHEAD>
<VALUE>/icons/ "/usr/share/apache2/icons/"</VALUE>
</listentry>
<listentry>
  <KEY>_SECTION</KEY>
  <OVERHEAD>
  </OVERHEAD>
  <SECTIONNAME>Directory</SECTIONNAME>
  <SECTIONPARAM>"/usr/share/apache2/icons"</SECTIONPARAM>
  <VALUE config:type="list">
    <listentry>
      <KEY>Options</KEY>
      <VALUE>Indexes MultiViews</VALUE>
    </listentry>
    <listentry>
      <KEY>AllowOverride</KEY>
      <VALUE>None</VALUE>
    </listentry>
    <listentry>
      <KEY>_SECTION</KEY>
      <SECTIONNAME>IfModule</SECTIONNAME>
      <SECTIONPARAM>!mod_access_compat.c</SECTIONPARAM>
      <VALUE config:type="list">
        <listentry>
          <KEY>Require</KEY>
          <VALUE>all granted</VALUE>
        </listentry>
      </VALUE>
    </listentry>
    <listentry>
      <KEY>_SECTION</KEY>
      <SECTIONNAME>IfModule</SECTIONNAME>
      <SECTIONPARAM>mod_access_compat.c</SECTIONPARAM>
      <VALUE config:type="list">
        <listentry>
          <KEY>Order</KEY>
          <VALUE>allow,deny</VALUE>
        </listentry>
        <listentry>
          <KEY>Allow</KEY>
          <VALUE>from all</VALUE>
        </listentry>
      </VALUE>
    </listentry>
  </VALUE>
</listentry>
<listentry>

```

```

<KEY>ScriptAlias</KEY>
<OVERHEAD>
# ScriptAlias: This controls which directories contain server
# scripts. ScriptAliases are essentially the same as Aliases,
# except that documents in the realname directory are treated
# as applications and run by the server when requested rather
# than as documents sent to the client.
# The same rules about trailing "/" apply to ScriptAlias
# directives as to Alias.
#
</OVERHEAD>
<VALUE>/cgi-bin/ "/srv/www/cgi-bin/"</VALUE>
</listentry>
<listentry>
  <KEY>_SECTION</KEY>
  <OVERHEAD>
  # "/srv/www/cgi-bin" should be changed to wherever your
  # ScriptAliased CGI directory exists, if you have that configured.
  #
  </OVERHEAD>
  <SECTIONNAME>Directory</SECTIONNAME>
  <SECTIONPARAM>"/srv/www/cgi-bin"</SECTIONPARAM>
  <VALUE config:type="list">
    <listentry>
      <KEY>AllowOverride</KEY>
      <VALUE>None</VALUE>
    </listentry>
    <listentry>
      <KEY>Options</KEY>
      <VALUE>+ExecCGI -Includes</VALUE>
    </listentry>
    <listentry>
      <KEY>_SECTION</KEY>
      <SECTIONNAME>IfModule</SECTIONNAME>
      <SECTIONPARAM>!mod_access_compat.c</SECTIONPARAM>
      <VALUE config:type="list">
        <listentry>
          <KEY>Require</KEY>
          <VALUE>all granted</VALUE>
        </listentry>
      </VALUE>
    </listentry>
    <listentry>
      <KEY>_SECTION</KEY>
      <SECTIONNAME>IfModule</SECTIONNAME>
      <SECTIONPARAM>mod_access_compat.c</SECTIONPARAM>
      <VALUE config:type="list">

```



```

        <listentry>
            <KEY>Order</KEY>
            <VALUE>allow,deny</VALUE>
        </listentry>
        <listentry>
            <KEY>Allow</KEY>
            <VALUE>from all</VALUE>
        </listentry>
    </VALUE>
</listentry>
</VALUE>
</listentry>
<listentry>
    <KEY>_SECTION</KEY>
    <OVERHEAD>
    # UserDir: The name of the directory that is appended onto a
    # user's home directory if a ~user request is received.
    # To disable it, simply remove userdir from the list of modules
    # in APACHE_MODULES in /etc/sysconfig/apache2.
    #
    </OVERHEAD>
    <SECTIONNAME>IfModule</SECTIONNAME>
    <SECTIONPARAM>mod_userdir.c</SECTIONPARAM>
    <VALUE config:type="list">
        <listentry>
            <KEY>UserDir</KEY>
            <OVERHEAD>
            # Note that the name of the user directory ("public_html")
            # cannot simply be changed here, since it is a compile time
            # setting. The apache package would need to be rebuilt.
            # You could work around by deleting /usr/sbin/suexec, but
            # then all scripts from the directories would be executed
            # with the UID of the webserver.
            </OVERHEAD>
            <VALUE>public_html</VALUE>
        </listentry>
        <listentry>
            <KEY>Include</KEY>
            <OVERHEAD>
            # The actual configuration of the directory is in
            # /etc/apache2/mod_userdir.conf.
            </OVERHEAD>
            <VALUE>/etc/apache2/mod_userdir.conf</VALUE>
        </listentry>
    </VALUE>
</listentry>
<listentry>

```

```

<KEY>IncludeOptional</KEY>
<OVERHEAD>
# Include all *.conf files from /etc/apache2/conf.d/.
#
# This is mostly meant as a place for other RPM packages to drop
# in their configuration snippet.
#
#
# You can comment this out here if you want those bits include
# only in a certain virtual host, but not here.
</OVERHEAD>
<VALUE>/etc/apache2/conf.d/*.conf</VALUE>
</listentry>
<listentry>
  <KEY>IncludeOptional</KEY>
  <OVERHEAD>
  # The manual... if it is installed ('?' means it will not complain)
  </OVERHEAD>
  <VALUE>/etc/apache2/conf.d/apache2-manual?conf</VALUE>
</listentry>
<listentry>
  <KEY>ServerName</KEY>
  <VALUE>linux-wtyj</VALUE>
</listentry>
<listentry>
  <KEY>ServerAdmin</KEY>
  <OVERHEAD>
  </OVERHEAD>
  <VALUE>root@linux-wtyj</VALUE>
</listentry>
<listentry>
  <KEY>NameVirtualHost</KEY>
  <VALUE>192.168.43.2</VALUE>
</listentry>
</VALUE>
</hosts_entry>
<hosts_entry>
  <KEY>192.168.43.2/secondserver.suse.de</KEY>
  <VALUE config:type="list">
    <listentry>
      <KEY>DocumentRoot</KEY>
      <VALUE>/srv/www/htdocs</VALUE>
    </listentry>
    <listentry>
      <KEY>ServerName</KEY>
      <VALUE>secondserver.suse.de</VALUE>
    </listentry>
  </listentry>

```

```

<listentry>
  <KEY>ServerAdmin</KEY>
  <VALUE>second_server@suse.de</VALUE>
</listentry>
<listentry>
  <KEY>_SECTION</KEY>
  <SECTIONNAME>Directory</SECTIONNAME>
  <SECTIONPARAM>/srv/www/htdocs</SECTIONPARAM>
  <VALUE config:type="list">
    <listentry>
      <KEY>AllowOverride</KEY>
      <VALUE>None</VALUE>
    </listentry>
    <listentry>
      <KEY>Require</KEY>
      <VALUE>all granted</VALUE>
    </listentry>
  </VALUE>
</listentry>
</VALUE>
</hosts_entry>
</hosts>
<modules config:type="list">
  <module_entry>
    <change>enable</change>
    <name>socache_shmcb</name>
    <userdefined config:type="boolean">>true</userdefined>
  </module_entry>
  <module_entry>
    <change>enable</change>
    <name>reqtimeout</name>
    <userdefined config:type="boolean">>true</userdefined>
  </module_entry>
  <module_entry>
    <change>enable</change>
    <name>authn_core</name>
    <userdefined config:type="boolean">>true</userdefined>
  </module_entry>
  <module_entry>
    <change>enable</change>
    <name>authz_core</name>
    <userdefined config:type="boolean">>true</userdefined>
  </module_entry>
</modules>
<service config:type="boolean">>true</service>
<version>2.9</version>
</http-server>

```

List Name	List Elements	Description
Listen		List of host <u>Listen</u> settings
	PORT	port address
	ADDRESS	Network address. All addresses will be taken if this entry is empty.
hosts		List of Hosts configuration
	KEY	Host name; <code><KEY>main</KEY></code> defines the main hosts, for example <code><KEY>192.168.43.2/secondserver.suse.de</KEY></code>
	VALUE	List of different values describing the host.
modules		Module list. Only user-defined modules need to be described.
	name	Module name
	userdefined	For historical reasons, it is always set to <u>true</u> .
	change	For historical reasons, it is always set to <u>enable</u> .

Element	Description	Comment
version	Version of used Apache server	Only for information. Default 2.9
service	Enable Apache service	Optional. Default: false



Note: Firewall

To run an Apache server correctly, make sure the firewall is configured appropriately.

4.22 Squid Server

Squid is a caching and forwarding Web proxy.

EXAMPLE 4.49: SQUID SERVER CONFIGURATION

```
<squid>
  <acls config:type="list">
    <listentry>
      <name>QUERY</name>
      <options config:type="list">
        <option>cgi-bin \?</option>
      </options>
      <type>urlpath_regex</type>
    </listentry>
    <listentry>
      <name>apache</name>
      <options config:type="list">
        <option>Server</option>
        <option>^Apache</option>
      </options>
      <type>rep_header</type>
    </listentry>
    <listentry>
      <name>all</name>
      <options config:type="list">
        <option>0.0.0.0/0.0.0.0</option>
      </options>
      <type>src</type>
    </listentry>
    <listentry>
      <name>manager</name>
      <options config:type="list">
        <option>cache_object</option>
      </options>
      <type>proto</type>
    </listentry>
    <listentry>
      <name>localhost</name>
      <options config:type="list">
        <option>127.0.0.1/255.255.255.255</option>
```

```

    </options>
    <type>src</type>
</listentry>
<listentry>
  <name>to_localhost</name>
  <options config:type="list">
    <option>127.0.0.0/8</option>
  </options>
  <type>dst</type>
</listentry>
<listentry>
  <name>SSL_ports</name>
  <options config:type="list">
    <option>443</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>80</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>21</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>443</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>70</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>

```

```

    <options config:type="list">
      <option>210</option>
    </options>
  </type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>1025-65535</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>280</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>488</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>591</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>Safe_ports</name>
  <options config:type="list">
    <option>777</option>
  </options>
  <type>port</type>
</listentry>
<listentry>
  <name>CONNECT</name>
  <options config:type="list">
    <option>CONNECT</option>
  </options>
  <type>method</type>
</listentry>

```

```

</acls>
<http_accesses config:type="list">
  <listentry>
    <acl config:type="list">
      <listentry>manager</listentry>
      <listentry>localhost</listentry>
    </acl>
    <allow config:type="boolean">>true</allow>
  </listentry>
  <listentry>
    <acl config:type="list">
      <listentry>manager</listentry>
    </acl>
    <allow config:type="boolean">>false</allow>
  </listentry>
  <listentry>
    <acl config:type="list">
      <listentry>!Safe_ports</listentry>
    </acl>
    <allow config:type="boolean">>false</allow>
  </listentry>
  <listentry>
    <acl config:type="list">
      <listentry>CONNECT</listentry>
      <listentry>!SSL_ports</listentry>
    </acl>
    <allow config:type="boolean">>false</allow>
  </listentry>
  <listentry>
    <acl config:type="list">
      <listentry>localhost</listentry>
    </acl>
    <allow config:type="boolean">>true</allow>
  </listentry>
  <listentry>
    <acl config:type="list">
      <listentry>all</listentry>
    </acl>
    <allow config:type="boolean">>false</allow>
  </listentry>
</http_accesses>
<http_ports config:type="list">
  <listentry>
    <host/>
    <port>3128</port>
    <transparent config:type="boolean">>false</transparent>
  </listentry>

```



```

</http_ports>
<refresh_patterns config:type="list">
  <listentry>
    <case_sensitive config:type="boolean">true</case_sensitive>
    <max>10080</max>
    <min>1440</min>
    <percent>20</percent>
    <regexp>^ftp:</regexp>
  </listentry>
  <listentry>
    <case_sensitive config:type="boolean">true</case_sensitive>
    <max>1440</max>
    <min>1440</min>
    <percent>0</percent>
    <regexp>^gopher:</regexp>
  </listentry>
  <listentry>
    <case_sensitive config:type="boolean">true</case_sensitive>
    <max>4320</max>
    <min>0</min>
    <percent>20</percent>
    <regexp>.</regexp>
  </listentry>
</refresh_patterns>
<service_enabled_on_startup config:type="boolean">true</service_enabled_on_startup>
<settings>
  <access_log config:type="list">
    <listentry>/var/log/squid/access.log</listentry>
  </access_log>
  <cache_dir config:type="list">
    <listentry>ufs</listentry>
    <listentry>/var/cache/squid</listentry>
    <listentry>100</listentry>
    <listentry>16</listentry>
    <listentry>256</listentry>
  </cache_dir>
  <cache_log config:type="list">
    <listentry>/var/log/squid/cache.log</listentry>
  </cache_log>
  <cache_mem config:type="list">
    <listentry>8</listentry>
    <listentry>MB</listentry>
  </cache_mem>
  <cache_mgr config:type="list">
    <listentry>webmaster</listentry>
  </cache_mgr>
  <cache_replacement_policy config:type="list">

```

```

    <listentry>lru</listentry>
</cache_replacement_policy>
<cache_store_log config:type="list">
    <listentry>/var/log/squid/store.log</listentry>
</cache_store_log>
<cache_swap_high config:type="list">
    <listentry>95</listentry>
</cache_swap_high>
<cache_swap_low config:type="list">
    <listentry>90</listentry>
</cache_swap_low>
<client_lifetime config:type="list">
    <listentry>1</listentry>
    <listentry>days</listentry>
</client_lifetime>
<connect_timeout config:type="list">
    <listentry>2</listentry>
    <listentry>minutes</listentry>
</connect_timeout>
<emulate_httpd_log config:type="list">
    <listentry>off</listentry>
</emulate_httpd_log>
<error_directory config:type="list">
    <listentry/>
</error_directory>
<ftp_passive config:type="list">
    <listentry>on</listentry>
</ftp_passive>
<maximum_object_size config:type="list">
    <listentry>4096</listentry>
    <listentry>KB</listentry>
</maximum_object_size>
<memory_replacement_policy config:type="list">
    <listentry>lru</listentry>
</memory_replacement_policy>
<minimum_object_size config:type="list">
    <listentry>0</listentry>
    <listentry>KB</listentry>
</minimum_object_size>
</settings>
</squid>

```

Attribute	Values	Description
<u>acls</u>	List of Access Control Settings (ACLs).	Each list entry contains the name, type, and additional options. Use the YaST Squid configuration module to get an overview of possible entries.
<u>http_accesses</u>	In the Access Control table, access can be denied or allowed to ACL Groups.	<p>If there are more ACL Groups in one definition, access will be allowed or denied to members who belong to all ACL Groups at the same time.</p> <p>The Access Control table is checked in the order listed here. The first matching entry is used.</p>
<u>http_ports</u>	Define all ports where Squid will listen for clients' HTTP requests.	<p><u>Host</u> can contain a host name or IP address or remain empty.</p> <p><u>transparent</u> disables PMTU discovery when transparent.</p>
<u>refresh_patterns</u>	Refresh patterns define how Squid treats the objects in the cache.	<p>The refresh patterns are checked in the order listed here. The first matching entry is used.</p> <p><u>Min</u> determines how long (in minutes) an object should be considered fresh if no explicit expiry time is given. <u>Max</u> is the upper limit of how long objects without an explicit</p>

Attribute	Values	Description
		expiry time will be considered fresh. <u>Percent</u> is the percentage of the object's age (time since last modification). An object without an explicit expiry time will be considered fresh.
<u>settings</u>	Map of all available general parameters with default values.	Use the YaST Squid configuration module to get an overview about possible entries.
<u>service_enabled_on_startup</u>	Squid service start when booting.	Value: true/false

4.23 FTP Server

Configure your FTP Internet server settings.

EXAMPLE 4.50: **FTP SERVER CONFIGURATION:**

```
<ftp-server>
  <AnonAuthen>2</AnonAuthen>
  <AnonCreatDirs>NO</AnonCreatDirs>
  <AnonMaxRate>0</AnonMaxRate>
  <AnonReadOnly>NO</AnonReadOnly>
  <AntiWarez>YES</AntiWarez>
  <Banner>Welcome message</Banner>
  <CertFile/>
  <ChrootEnable>NO</ChrootEnable>
  <EnableUpload>YES</EnableUpload>
  <FTPUser>ftp</FTPUser>
  <FtpDirAnon>/srv/ftp</FtpDirAnon>
  <FtpDirLocal/>
  <GuestUser/>
  <LocalMaxRate>0</LocalMaxRate>
  <MaxClientsNumber>10</MaxClientsNumber>
  <MaxClientsPerIP>3</MaxClientsPerIP>
```

```

<MaxIdleTime>15</MaxIdleTime>
<PasMaxPort>40500</PasMaxPort>
<PasMinPort>40000</PasMinPort>
<PassiveMode>YES</PassiveMode>
<SSL>0</SSL>
<SSLEnable>NO</SSLEnable>
<SSLv2>NO</SSLv2>
<SSLv3>NO</SSLv3>
<StartDaemon>2</StartDaemon>
<TLS>YES</TLS>
<Umask/>
<UmaskAnon/>
<UmaskLocal/>
<VerboseLogging>NO</VerboseLogging>
<VirtualUser>NO</VirtualUser>
</ftp-server>

```

Element	Description	Comment
<u>AnonAuthen</u>	Enable/disable anonymous and local users.	Authenticated Users Only: 1; Anonymous Only: 0; Both: 2
<u>AnonCreatDirs</u>	Anonymous users can create directories.	Values: YES/NO
<u>AnonReadOnly</u>	Anonymous users can upload.	Values: YES/NO
<u>AnonMaxRate</u>	The maximum data transfer rate permitted for anonymous clients.	KB/s
<u>AntiWarez</u>	Disallow downloading of files that were uploaded but not validated by a local admin.	Values: YES/NO
<u>Banner</u>	Specify the name of a file containing the text to display when someone connects to the server.	

Element	Description	Comment
<u>CertFile</u>	DSA certificate to use for SSL-encrypted connections	This option specifies the location of the DSA certificate to use for SSL-encrypted connections.
<u>ChrootEnable</u>	When enabled, local users will by default be placed in a <u>chroot</u> jail in their home directory after login.	Warning: This option has security implications. Values: YES/NO
<u>EnableUpload</u>	If enabled, FTP users can upload.	To allow anonymous users to upload, enable <u>AnonReadOnly</u> . Values: YES/NO
<u>FTPUser</u>	Defines the anonymous FTP user.	
<u>FtpDirAnon</u>	FTP directory for anonymous users.	Specify a directory which is used for anonymous FTP users.
<u>FtpDirLocal</u>	FTP directory for authenticated users.	Specify a directory which is used for FTP authenticated users.
<u>LocalMaxRate</u>	The maximum data transfer rate permitted for local authenticated users.	KB/s
<u>MaxClientsNumber</u>	The maximum number of clients allowed to connect.	
<u>MaxClientsPerIP</u>	Defines the maximum number of clients for one IP.	This limits the number of clients allowed to connect from a single source Internet address.

Element	Description	Comment
<u>MaxIdleTime</u>	The maximum time (timeout) a remote client may wait between FTP commands.	Minutes
<u>PasMaxPort</u>	Maximum value for a port range for passive connection replies.	<u>PassiveMode</u> needs to be set to YES.
<u>PasMinPort</u>	Minimum value for a port range for passive connection replies.	<u>PassiveMode</u> needs to be set to YES.
<u>PassiveMode</u>	Enable Passive Mode	Value: YES/NO
<u>SSL</u>	Security Settings	Disable SSL/TLS: 0; Accept SSL and TLS: 1; Refuse Connections Without SSL/TLS: 2
<u>SSLEnable</u>	If enabled, SSL connections are allowed.	Value: YES/NO
<u>SSLv2</u>	If enabled, SSL version 2 connections are allowed.	Value: YES/NO
<u>SSLv3</u>	If enabled, SSL version 3 connections are allowed.	Value: YES/NO
<u>StartDaemon</u>	How the FTP daemon will be started.	Manually: 0; when booting: 1; via <u>systemd</u> socket: 2
<u>TLS</u>	If enabled, TLS connections are allowed.	Value: YES/NO
<u>Umask</u>	File creation mask, in the format (umask for files):(umask for directories).	For example <u>177:077</u> if you feel paranoid.

Element	Description	Comment
<u>UmaskAnon</u>	The value to which the umask for file creation is set for anonymous users.	To specify octal values, remember the "0" prefix, otherwise the value will be treated as a base-10 integer.
<u>UmaskLocal</u>	Umask for authenticated users.	To specify octal values, remember the "0" prefix, otherwise the value will be treated as a base-10 integer.
<u>VerboseLogging</u>	When enabled, all FTP requests and responses are logged.	Value: YES/NO
<u>VirtualUser</u>	By using virtual users, FTP accounts can be administrated without affecting system accounts.	Value: YES/NO



Note: Firewall

Proper Firewall setting will be required for the FTP server to run correctly.

4.24 TFTP Server

Configure your TFTP Internet server settings.

Use this to enable a server for TFTP (trivial file transfer protocol). The server will be started using the systemd socket.

Note that TFTP and FTP are not the same.

EXAMPLE 4.51: TFTP SERVER CONFIGURATION:

```
<tftp-server>
  <start_tftpd config:type="boolean">true</start_tftpd>
  <tftp_directory>/tftpboot</tftp_directory>
```



```
</tftp-server>
```

Element	Description	Comment
start_tftpd	Enabling TFTP server service.	Value: true/false
tftp_directory	Boot Image Directory: Specify the directory where served files are located.	The usual value is /tftpboot. The directory will be created if it does not exist. The server uses this as its root directory (using the -s option).

4.25 Firstboot Workflow

The YaST firstboot utility (YaST Initial System Configuration), which runs after the installation is completed, lets you configure the freshly installed system. On the first boot after the installation, users are guided through a series of steps that allow for easier configuration of a system. YaST firstboot does not run by default and needs to be configured to run.

EXAMPLE 4.52: ENABLING FIRSTBOOT WORKFLOW

```
<firstboot>
  <firstboot_enabled config:type="boolean">true</firstboot_enabled>
</firstboot>
```

4.26 Security Settings

Using the features of this module, you can to change the local security settings on the target system. The local security settings include the boot configuration, login settings, password settings, user addition settings, and file permissions.

Configuring the security settings automatically is similar to the [Custom Settings](#) in the security module available in the running system. This allows you create a customized configuration.

EXAMPLE 4.53: SECURITY CONFIGURATION

See the reference for the meaning and the possible values of the settings in the following example.

```
<security>
```

```
<console_shutdown>ignore</console_shutdown>
<displaymanager_remote_access>no</displaymanager_remote_access>
<fail_delay>3</fail_delay>
<faillog_enab>yes</faillog_enab>
<gid_max>60000</gid_max>
<gid_min>101</gid_min>
<gdm_shutdown>root</gdm_shutdown>
<lastlog_enab>yes</lastlog_enab>
<encryption>md5</encryption>
<obscure_checks_enab>no</obscure_checks_enab>
<pass_max_days>99999</pass_max_days>
<pass_max_len>8</pass_max_len>
<pass_min_days>1</pass_min_days>
<pass_min_len>6</pass_min_len>
<pass_warn_age>14</pass_warn_age>
<passwd_use_cracklib>yes</passwd_use_cracklib>
<permission_security>secure</permission_security>
<run_updatedb_as>nobody</run_updatedb_as>
<uid_max>60000</uid_max>
<uid_min>500</uid_min>
</security>
```

4.26.1 Password Settings Options

Change various password settings. These settings are mainly stored in the [/etc/login.defs](#) file.

Use this resource to activate one of the encryption methods currently supported. If not set, [DES](#) is configured.

[DES](#), the Linux default method, works in all network environments, but it restricts you to passwords no longer than eight characters. [MD5](#) allows longer passwords, thus provides more security, but some network protocols do not support this, and you may have problems with NIS. [Blowfish](#) is also supported.

Additionally, you can set up the system to check for password plausibility and length etc.

4.26.2 Boot Settings

Use the security resource, to change various boot settings.

How to interpret ?

When someone at the console has pressed the `Ctrl-Alt-Del` key combination, the system usually reboots. Sometimes it is desirable to ignore this event, for example, when the system serves as both workstation and server.

Shutdown behavior of GDM

Configure a list of users allowed to shut down the machine from GDM.

4.26.3 Login Settings

Change various login settings. These settings are mainly stored in the `/etc/login.defs` file.

4.26.4 New user settings (**useradd** settings)

Set the minimum and maximum possible user and group IDs.

4.27 Linux Audit Framework (LAF)

This module allows the configuration of the audit daemon and to add rules for the audit subsystem.

EXAMPLE 4.54: LAF CONFIGURATION

```
<audit-laf>
  <auditd>
    <flush>INCREMENTAL</flush>
    <freq>20</freq>
    <log_file>/var/log/audit/audit.log</log_file>
    <log_format>RAW</log_format>
    <max_log_file>5</max_log_file>
    <max_log_file_action>ROTATE</max_log_file_action>
    <name_format>NONE</name_format>
    <num_logs>4</num_logs>
  </auditd>
  <rules/>
</audit-laf>
```

Attribute	Values	Description
<u>auditd/flush</u>	Describes how to write the data to disk.	If set to <u>INCREMENTAL</u> the Frequency parameter tells how many records to write before issuing an explicit flush to disk. <u>NONE</u> means: no special effort is made to flush data, <u>DATA</u> : keep data portion synchronized, <u>SYNC</u> : keep data and metadata fully synchronized.
<u>auditd/freq</u>	This parameter tells how many records to write before issuing an explicit flush to disk.	The parameter <u>flush</u> needs to be set to <u>INCREMENTAL</u> .
<u>auditd/log_file</u>	The full path name to the log file.	
<u>auditd/log_fomat</u>	How much information needs to be logged.	Set <u>RAW</u> to log all data (store in a format exactly as the kernel sends it) or <u>NOLOG</u> to discard all audit information instead of writing it to disk (does not affect data sent to the dispatcher).
<u>auditd/max_log_file</u>	How much information needs to be logged.	Unit: Megabytes
<u>auditd/num_logs</u>	Number of log files.	<u>max_log_file_action</u> needs to be set to <u>ROTATE</u>
<u>auditd/max_log_file_action</u>	What happens if the log capacity has been reached.	If the action is set to <u>ROTATE</u> the Number of Log Files specifies the number of

Attribute	Values	Description
		files to keep. Set to <u>SYSLOG</u> , the audit daemon will write a warning to the system log. With <u>SUSPEND</u> the daemon stops writing records to disk. <u>IGNORE</u> means do nothing, <u>KEEP_LOGS</u> is similar to <u>ROTATE</u> , but log files are not overwritten.
<u>auditd/name_format</u>	Computer Name Format describes how to write the computer name to the log file.	If <u>USER</u> is set, the user-defined name is used. <u>NONE</u> means no computer name is inserted. <u>HOSTNAME</u> uses the name returned by the 'gethostname' syscall. <u>FQD</u> uses the fully qualified domain name.
<u>rules</u>	Rules for auditctl	You can edit the rules manually, which we only recommend for advanced users. For more information about all options, see <u>man auditctl</u> .

4.28 Users and Groups

4.28.1 Users

A list of users can be defined in the <users> section. To be able to log in, make sure that either the root users are set up or rootpassword is specified as a linuxrc option.

EXAMPLE 4.55: MINIMAL USER CONFIGURATION

```
<users config:type="list">
```

```

<user>
  <username>root</username>
  <user_password>password</user_password>
  <encrypted config:type="boolean">>false</encrypted>
</user>
  <user>
  <username>tux</username>
  <user_password>password</user_password>
  <encrypted config:type="boolean">>false</encrypted>
</user>
</users>

```

The following example shows a more complex scenario. System-wide default settings from `/etc/default/useradd`, such as the shell or the parent directory for the home directory, are applied.

EXAMPLE 4.56: COMPLEX USER CONFIGURATION

```

<users config:type="list">
  <user>
    <username>root</username>
    <user_password>password</user_password>
    <uid>1001</uid>
    <gid>100</gid>
    <encrypted config:type="boolean">>false</encrypted>
    <fullname>Root User</fullname>
    <authorized_keys config:type="list">
      <listentry>command="/opt/login.sh" ssh-rsa
        AAAAB3NzaC1yc2EAAAADAQABAAQDKLt1vnW2vTJpBp3VK91rFsBvpY97NljsVLdgUrLpBZ/
        L51FerQQ+djQ/ivDASQj0+567nMGqfYGFA/De1EGMMEoeShza67qjNi14L1HBGgVojaNajMR/
        NI2d1kDyvsgRy7D7FT5UGGUNT0dLcSD3b85zWgHeYLidgcGIoKeRi7HpVD00TyhwUv4sq3ubrPCWARgPe0LdVFa9cLC8PTZdxSeKp4j
        PvMDa96DpxH1VlzJlAIHQsMkMHbsCazPNC0++Kp5ZVERiH root@example.net</listentry>
    </authorized_keys>
  </user>
  <user>
    <username>tux</username>
    <user_password>password</user_password>
    <uid>1002</uid>
    <gid>100</gid>
    <encrypted config:type="boolean">>false</encrypted>
    <fullname>Plain User</fullname>
    <home>/Users/plain</home>
    <password_settings>
      <max>120</max>
      <inact>5</inact>
    </password_settings>

```

```
</user>
</users>
```



Note: `authorized_keys` File Will Be Overwritten

If the profile defines a set of SSH authorized keys for a user in the `authorized_keys` section, an existing `$HOME/.ssh/authorized_keys` file will be overwritten. If not existing, the file will be created with the content specified. Avoid overwriting an existing `authorized_keys` file by not specifying the respective section in the AutoYaST control file.



Note: Combine `rootpassword` and Root User Options

It is possible to specify `rootpassword` in `linuxrc` and have a user section for the `root` user. If this section is missing the password, then the password from `linuxrc` will be used. Passwords in profiles take precedence over `linuxrc` passwords.



Note: Specifying a User ID (`uid`)

Each user on a Linux system has a numeric user ID. You can either specify such a user ID within the AutoYaST control file manually by using `uid`, or let the system automatically choose a user ID by not using `uid`.

User IDs should be unique throughout the system. If not, some applications such as the login manager `gdm` may no longer work as expected.

When adding users with the AutoYaST control file, it is strongly recommended not to mix user-defined IDs and automatically provided IDs. When doing so, unique IDs cannot be guaranteed. Either specify IDs for all users added with the AutoYaST control file or let the system choose the ID for all users.

Attribute	Values	Description
<code>username</code>	Text <pre><username>lukesw</username></pre>	Required. It should be a valid user name. Check <code>man 8 useradd</code> if you are not sure.
<code>fullname</code>	Text	Optional. User's full name.

Attribute	Values	Description
	<pre><fullname>Tux Torvalds</fullname></pre>	
<u>forename</u>	Text <pre><forename>Tux</forename></pre>	Optional. User's forename.
<u>surname</u>	Text <pre><surname>Skywalker</surname></pre>	Optional. User's surname.
<u>uid</u>	Number <pre><uid>1001</uid></pre>	Optional. User ID. It should be a unique and must be a non-negative number. If not specified, AutoYaST will automatically choose a user ID. Also refer to <i>Note: Specifying a User ID (uid)</i> for additional information.
<u>gid</u>	Number <pre><gid>100</gid></pre>	Optional. Initial group ID. It must be a unique and non-negative number. Moreover it must refer to an existing group.
<u>home</u>	Path <pre><home>/home/luke</home></pre>	Optional. Absolute path to the user's home directory. By default, <u>/home/username</u> will be used (for example, <u>alice</u> 's home directory will be <u>/home/alice</u>).

Attribute	Values	Description
<u>home_btrfs_subvolume</u>	Boolean <pre><home_btrfs_subvolume config:type="boolean">true</ home_btrfs_subvolume></pre>	Optional. Generates the home directory in a Btrfs subvolume. Disabled by default.
<u>shell</u>	Path <pre><shell>/usr/bin/zsh</shell></pre>	Optional. <code>/bin/bash</code> is the default value. If you choose another one, make sure that it is installed (adding the corresponding package to the <u>software</u> section).
<u>user_password</u>	Text <pre><user_password>some- password</user_password></pre>	Optional. If you enter an exclamation mark (<code>!</code>), a random password will be generated. A user's password can be written in plain text (not recommended) or in encrypted form. To create an encrypted password, use mkpasswd . Enter the password as written in <code>/etc/shadow</code> (second column). To enable or disable the use of encrypted passwords in the profile, see the <u>encrypted</u> parameter.
<u>encrypted</u>	Boolean <pre><encrypted config:type="boolean">true</ encrypted></pre>	Optional. Considered <u>false</u> if not present. Indicates if the user's password in the profile is encrypted or not. Au-

Attribute	Values	Description
		toYaST supports standard encryption algorithms (see man 3 crypt).
password_settings	Password settings <pre><password_settings> <expire/> <max>60</max> <warn>7</warn> </password_settings></pre>	Optional. Some password settings can be customized: expire (account expiration date in format YYYY-MM-DD), flag (/etc/shadow flag), inact (number of days after password expiration that account is disabled), max (maximum number of days a password is valid), min (grace period in days until which a user can change password after it has expired) and warn (number of days before expiration when the password change reminder starts).
authorized_keys	List of authorized keys <pre><authorized_keys config:type="list"> <listentry>ssh-rsa ...</listentry> </authorized_keys></pre>	A list of authorized keys to be written to \$HOME/.ssh/authorized_keys . See example below.

4.28.2 User Defaults

The profile can specify a set of default values for new users like password expiration, initial group, home directory prefix, etc. Besides using them as default values for the users that are defined in the profile, AutoYaST will write those settings to [/etc/default/useradd](#) to be read for [useradd](#).

Attribute	Values	Description
<u>group</u>	Text <pre><group>100</group></pre>	Optional. Default initial login group.
<u>groups</u>	Text <pre><groups>users</groups></pre>	Optional. List of additional groups.
<u>home</u>	Path <pre><home>/home</home></pre>	Optional. User's home directory prefix.
<u>expire</u>	Date <pre><expire>2017-12-31</expire></pre>	Optional. Default password expiration date in <u>YYYY-MM-DD</u> format.
<u>inactive</u>	Number <pre><inactive>3</inactive></pre>	Optional. Number of days after which an expired account is disabled.
<u>no_groups</u>	Boolean <pre><no_groups config:type="boolean">true</ no_groups></pre>	Optional. Do not use secondary groups.
<u>shell</u>	Path <pre><shell>/usr/bin/fish</ shell></pre>	Default login shell. <u>/bin/bash</u> is the default value. If you choose another one, make sure that it is installed (adding the corresponding package to the <u>software</u> section).

Attribute	Values	Description
<u>skel</u>	Path <pre><skel>/etc/skel</skel></pre>	Optional. Location of the files to be used as skeleton when adding a new user. You can find more information in man 8 useradd .
<u>umask</u>	File creation mode mask <pre><umask>022</umask></pre>	Set the file creation mode mask for the home directory. By default useradd will use 022 . Check man 8 useradd and man 1 umask for further information.

4.28.3 Groups

A list of groups can be defined in `<groups>` as shown in the example.

EXAMPLE 4.57: GROUP CONFIGURATION

```
<groups config:type="list">
  <group>
    <gid>100</gid>
    <groupname>users</groupname>
    <userlist>bob,alice</userlist>
  </group>
</groups>
```

Attribute	Values	Description
<u>groupname</u>	Text <pre><groupname>users</groupname></pre>	Required. It should be a valid group name. Check man 8 groupadd if you are not sure.
<u>gid</u>	Number <pre><gid>100</gid></pre>	Optional. Group ID. It must be a unique and non-negative number.

Attribute	Values	Description
<u>group_password</u>	Text <pre><group_password>password</group_password></pre>	Optional. The group's password can be written in plain text (not recommended) or in encrypted form. Check the <u>encrypted</u> to select the desired behavior.
<u>encrypted</u>	Boolean <pre><encrypted config:type="boolean">true</encrypted></pre>	Optional. Indicates if the group's password in the profile is encrypted or not.
<u>userlist</u>	Users list <pre><userlist>bob,alice</userlist></pre>	Optional. A list of users who belong to the group. User names must be separated by commas.

4.28.4 Login Settings

Two special login settings can be enabled through an AutoYaST profile: autologin and password-less login. Both of them are disabled by default.

EXAMPLE 4.58: ENABLING AUTOLOGIN AND PASSWORD-LESS LOGIN

```
<login_settings>
  <autologin_user>vagrant</autologin_user>
  <password_less_login config:type="boolean">true</password_less_login>
</login_settings>
```

Attribute	Values	Description
<u>password_less_login</u>	Boolean <pre><password_less_login config:type="boolean">true</password_less_login></pre>	Optional. Enables password-less login. It only affects graphical login.

Attribute	Values	Description
<u>autologin_user</u>	Text <pre><autologin_user>alice</autologin_user></pre>	Optional. Enables autologin for the given user.

4.29 Custom User Scripts

By adding scripts to the auto-installation process you can customize the installation according to your needs and take control in different stages of the installation.

In the auto-installation process, five types of scripts can be executed at different points in time during the installation:

All scripts need to be in the `<scripts>` section.

- pre-scripts (very early, before anything else really happens)
- postpartitioning-scripts (after partitioning and mounting to `/mnt` but before RPM installation)
- chroot-scripts (after the package installation, before the first boot)
- post-scripts (during the first boot of the installed system, no services running)
- init-scripts (during the first boot of the installed system, all services up and running)

4.29.1 Pre-Install Scripts

Executed before YaST does any real change to the system (before partitioning and package installation but after the hardware detection).

You can use a pre-script to modify your control file and let AutoYaST reread it. Find your control file in `/tmp/profile/autoinst.xml`. Adjust the file and store the modified version in `/tmp/profile/modified.xml`. AutoYaST will read the modified file after the pre-script finishes.

It is also possible to modify the storage devices in your pre-scripts. For example, you can create new partitions or change the configuration of certain technologies like multipath. AutoYaST always inspects the storage devices again after executing all the pre-install scripts.



Note: Pre-Install Scripts with Confirmation

Pre-scripts are executed at an early stage of the installation. This means if you have requested to confirm the installation, the pre-scripts will be executed before the confirmation screen shows up ([profile/install/general/mode/confirm](#)).



Note: Pre-Install and Zypper

To call *zypper* in the pre-install script you will need to set the environment variable `ZYPP_LOCKFILE_ROOT="/var/run/autoyast"` to prevent conflicts with the running YaST process.

Pre-Install Script elements must be placed as follows:

```
<scripts>
  <pre-scripts config:type="list">
    <script>
      ...
    </script>
  </pre-scripts>
</scripts>
```

4.29.2 Post-partitioning Scripts

Executed after YaST has done the partitioning and written [/etc/fstab](#). The empty system is already mounted to [/mnt](#).

Post-partitioning script elements must be placed as follows:

```
<scripts>
  <postpartitioning-scripts config:type="list">
    <script>
      ...
    </script>
  </postpartitioning-scripts>
</scripts>
```

4.29.3 Chroot Environment Scripts

Chroot scripts are executed before the machine reboots for the first time. You can execute chroot scripts before the installation chroots into the installed system and configures the boot loader or you can execute a script after the chroot into the installed system has happened (look at the `chrooted` parameter for that).

Chroot Environment script elements must be placed as follows:

```
<scripts>
  <chroot-scripts config:type="list">
    <script>
      ...
    </script>
  </chroot-scripts>
</scripts>
```

4.29.4 Post-Install Scripts

These scripts are executed after AutoYaST has completed the system configuration and after it has booted the system for the first time.

Post-install script elements must be placed as follows:

```
<scripts>
  <post-scripts config:type="list">
    <script>
      ...
    </script>
  </post-scripts>
</scripts>
```

4.29.5 Init Scripts

These scripts are executed when YaST has finished, during the initial boot process after the network has been initialized. These final scripts are executed using `/usr/lib/YaST2/bin/autoyast-initscripts.sh` and are executed only once. Init scripts are configured using the tag `init-scripts`.

The following elements must be between the `<scripts>` `<init-scripts config:type="list">` `<script>` ... `</script>` `</init-scripts>` ... `</scripts>` tags

TABLE 4.1: INIT SCRIPT XML REPRESENTATION

Element	Description	Comment
<u>location</u>	<p>Define a location from where the script gets fetched. Locations can be the same as for the profile (HTTP, FTP, NFS, etc.).</p> <pre><location> http://10.10.0.1/ myInitScript.sh</location></pre>	Either <code><location></code> or <code><source></code> must be defined.
<u>source</u>	<p>The script itself (source code), encapsulated in a CDATA tag. If you do not want to put the whole shell script into the XML profile, use the location parameter.</p> <pre><source> <![CDATA[echo "Testing the init script" > /tmp/init_out.txt]]> </source></pre>	Either <code><location></code> or <code><source></code> must be defined.
<u>filename</u>	<p>The file name of the script. It will be stored in a temporary directory under <code>/tmp</code></p> <pre><filename>myinitScript5.sh</filename></pre>	Optional in case you only have a single init script. The default name (<code>init-scripts</code>) is used in this case. If having specified more than one init script, you must set a unique name for each script.
<u>rerun</u>	<p>Normally, a script is only run once, even if you use <code>ayast_setup</code> to run an XML</p>	Optional. Default is <code>false</code> (scripts only run once).

Element	Description	Comment
	<p>file multiple times. Change this default behavior by setting this boolean to <code>true</code>.</p> <pre><rerun config:type="boolean">true</ rerun></pre>	

When added to the control file manually, scripts need to be included in a *CDATA* element to avoid confusion with the file syntax and other tags defined in the control file.

4.29.6 Script XML Representation

The XML elements described below can be used for all the script types described above, except for the `chrooted` element, which can only be used in chroot scripts.

TABLE 4.2: SCRIPT XML REPRESENTATION

Element	Description	Comment
<code>location</code>	<p>Define a location from where the script gets fetched. Locations can be the same as for the control file (HTTP, FTP, NFS, etc.).</p> <pre><location >http://10.10.0.1/myPreScript.sh</ location></pre>	Either <code>location</code> or <code>source</code> must be defined.
<code>source</code>	<p>The script itself (source code), encapsulated in a <i>CDATA</i> tag. If you do not want to put the whole shell script into the XML control file, refer to the location parameter.</p> <pre><source> <![CDATA[echo "Testing the pre script" > /tmp/pre- script_out.txt]]> </source></pre>	Either <code>location</code> or <code>source</code> must be defined.

Element	Description	Comment
<u>inter- preter</u>	Specify the interpreter that must be used for the script. Supported options are <u>shell</u> and <u>perl</u> . <pre><interpreter>perl</interpreter></pre>	Optional; default is <u>shell</u> .
<u>file name</u>	The file name of the script. It will be stored in a temporary directory under <u>/tmp</u> . <pre><filename>myPreScript5.sh</filename></pre>	Optional; default is the type of the script (pre-scripts in this case). If you have more than one script, you should define different names for each script.
<u>feedback</u>	If this boolean is <u>true</u> , output and error messages of the script (STDOUT and STDERR) will be shown in a pop-up. The user needs to confirm them via the OK button. <pre><feedback config:type="boolean">true</feedback></pre>	Optional; default is <u>false</u> .
<u>feed- back_type</u>	This can be <u>message</u> , <u>warning</u> or <u>error</u> . Set the timeout for these pop-ups in the <code><report></code> section. <pre><feedback_type>warning</feedback_type></pre>	Optional; if missing, an always-blocking pop-up is used.
<u>debug</u>	If this is <u>true</u> , every single line of a shell script is logged. Perl scripts are run with warnings turned on. <pre><debug config:type="boolean">true</debug></pre>	Optional; default is <u>true</u> .
<u>notifica- tion</u>	This text will be shown in a pop-up for the time the script is running in the background.	Optional; if not configured, no notification pop-up will be shown.

Element	Description	Comment
	<pre><notification>Please wait while script is running...</notification></pre>	
<u>param-list</u>	<p>It is possible to specify parameters given to the script being called. You may have more than one <u>param</u> entry. They are concatenated by a single space character on the script command line. If any shell quoting should be necessary (for example to protect embedded spaces) you need to include this.</p> <pre><param-list config:type="list"> <param>par1</param> <param>par2 par3</param> <param>"par4.1 par4.2"</param> </param-list></pre>	Optional; if not configured, no parameters get passed to script.
<u>rerun</u>	<p>A script is only run once. Even if you use <u>ayast_setup</u> to run an XML file multiple times, the script is only run once. Change this default behavior by setting this boolean to <u>true</u>.</p> <pre><rerun config:type="boolean">true</rerun></pre>	Optional; default is <u>false</u> , meaning that scripts only run once.
<u>chrooted</u>	<p>During installation, the new system is mounted at <u>/mnt</u>. If this parameter is set to <u>false</u>, AutoYaST does not run <u>chroot</u> and does not install the boot loader at this stage. If the parameter is set to <u>true</u>, AutoYaST performs a <u>chroot</u> into <u>/mnt</u> and installs the boot loader. The result is that to change anything in the newly-installed system, you no longer need to use the <u>/mnt</u> prefix.</p> <pre><chrooted config:type="boolean">true</chrooted></pre>	Optional; default is <u>false</u> . This option is only available for chroot environment scripts.

4.29.7 Script Example

EXAMPLE 4.59: SCRIPT CONFIGURATION

```
<?xml version="1.0"?>
<!DOCTYPE profile>
<profile xmlns="http://www.suse.com/1.0/yast2ns" xmlns:config="http://www.suse.com/1.0/
confgns">
<scripts>
  <chroot-scripts config:type="list">
    <script>
      <chrooted config:type="boolean">>true</chrooted>
      <filename>chroot.sh</filename>
      <interpreter>shell</interpreter>
      <source><![CDATA[
#!/bin/sh
echo "Testing chroot (chrooted) scripts"
ls
]]>
      </source>
    </script>
    <script>
      <filename>chroot.sh</filename>
      <interpreter>shell</interpreter>
      <source><![CDATA[
#!/bin/sh
echo "Testing chroot scripts"
df
cd /mnt
ls
]]>
      </source>
    </script>
  </chroot-scripts>
  <post-scripts config:type="list">
    <script>
      <filename>post.sh</filename>
      <interpreter>shell</interpreter>
      <source><![CDATA[
#!/bin/sh

echo "Running Post-install script"
systemctl start portmap
mount -a 192.168.1.1:/local /mnt
cp /mnt/test.sh /tmp
umount /mnt
]]>
```

```

        </source>
    </script>
    <script>
        <filename>post.pl</filename>
        <interpreter>perl</interpreter>
        <source><![CDATA[
#!/usr/bin/perl
print "Running Post-install script";

]]>
        </source>
    </script>
</post-scripts>
<pre-scripts config:type="list">
    <script>
        <interpreter>shell</interpreter>
        <location>http://192.168.1.1/profiles/scripts/prescripts.sh</location>
    </script>
    <script>
        <filename>pre.sh</filename>
        <interpreter>shell</interpreter>
        <source><![CDATA[
#!/bin/sh
echo "Running pre-install script"
]]>
        </source>
    </script>
</pre-scripts>
<postpartitioning-scripts config:type="list">
    <script>
        <filename>postpart.sh</filename>
        <interpreter>shell</interpreter>
        <debug config:type="boolean">>false</debug>
        <feedback config:type="boolean">>true</feedback>
        <source><![CDATA[
touch /mnt/testfile
echo Hi
]]>
        </source>
    </script>
</postpartitioning-scripts>
</scripts>
</profile>

```

After installation is finished, the scripts and the output logs can be found in the directory `/var/adm/autoinstall`. The scripts are located in the subdirectory `scripts` and the output logs in the `log` directory.

The log consists of the output produced when executing the shell scripts using the following command:

```
/bin/sh -x SCRIPT_NAME 2&>/var/adm/autoinstall/logs/SCRIPT_NAME.log
```

4.30 System Variables (Sysconfig)

Using the `sysconfig` resource, it is possible to define configuration variables in the `sysconfig` repository (`/etc/sysconfig`) directly. `Sysconfig` variables, offer the possibility to fine-tune many system components and environment variables exactly to your needs.

The following example shows how a variable can be set using the `sysconfig` resource.

EXAMPLE 4.60: SYSCONFIG CONFIGURATION

```
<sysconfig config:type="list" >
  <sysconfig_entry>
    <sysconfig_key>XNTPD_INITIAL_NTPDATE</sysconfig_key>
    <sysconfig_path>/etc/sysconfig/xntp</sysconfig_path>
    <sysconfig_value>nntp.host.com</sysconfig_value>
  </sysconfig_entry>
  <sysconfig_entry>
    <sysconfig_key>HTTP_PROXY</sysconfig_key>
    <sysconfig_path>/etc/sysconfig/proxy</sysconfig_path>
    <sysconfig_value>proxy.host.com:3128</sysconfig_value>
  </sysconfig_entry>
  <sysconfig_entry>
    <sysconfig_key>FTP_PROXY</sysconfig_key>
    <sysconfig_path>/etc/sysconfig/proxy</sysconfig_path>
    <sysconfig_value>proxy.host.com:3128</sysconfig_value>
  </sysconfig_entry>
</sysconfig>
```

Both relative and absolute paths can be provided. If no absolute path is given, it is treated as a `sysconfig` file under the `/etc/sysconfig` directory.

4.31 Adding Complete Configurations

For many applications and services you may have a configuration file which should be copied to the appropriate location on the installed system. For example, if you are installing a Web server, you may have a server configuration file (`httpd.conf`).

Using this resource, you can embed the file into the control file by specifying the final path on the installed system. YaST will copy this file to the specified location.

This feature requires the `autoyast2` package to be installed. If the package is missing, AutoYaST will automatically install the package if it is missing.

You can specify the `file_location` where the file should be retrieved from. This can also be a location on the network such as an HTTP server: `<file_location>http://my.server.site/issue</file_location>`.

You can create directories by specifying a `file_path` that ends with a slash.

EXAMPLE 4.61: DUMPING FILES INTO THE INSTALLED SYSTEM

```
<files config:type="list">
  <file>
    <file_path>/etc/apache2/httpd.conf</file_path>
    <file_contents>

<![CDATA[
some content
]]>

    </file_contents>
  </file>
  <file>
    <file_path>/mydir/a/b/c/</file_path> <!-- create directory -->
  </file>
</files>
```

A more advanced example is shown below. This configuration will create a file using the content supplied in `file_contents` and change the permissions and ownership of the file. After the file has been copied to the system, a script is executed. This can be used to modify the file and prepare it for the client's environment.

EXAMPLE 4.62: DUMPING FILES INTO THE INSTALLED SYSTEM

```
<files config:type="list">
  <file>
```



```

<file_path>/etc/someconf.conf</file_path>
<file_contents>

<![CDATA[
some content
]]>

</file_contents>
<file_owner>tux.users</file_owner>
<file_permissions>444</file_permissions>
<file_script>
  <interpreter>shell</interpreter>
  <source>

<![CDATA[
#!/bin/sh

echo "Testing file scripts" >> /etc/someconf.conf
df
cd /mnt
ls
]]>

  </source>
</file_script>
</file>
</files>

```

4.32 Ask the User for Values during Installation

You have the option to let the user decide the values of specific parts of the control file during the installation. If you use this feature, a pop-up will ask the user to enter a specific part of the control file during installation. If you want a full auto installation, but the user should set the password of the local account, you can do this via the `ask` directive in the control file.

The elements listed below must be placed within the following XML structure:

```

<general>
  <ask-list config:type="list">
    <ask>
      ...
    </ask>
  </ask-list>
</general>

```

TABLE 4.3: ASK THE USER FOR VALUES: XML REPRESENTATION

Element	Description	Comment
<u>question</u>	<p>The question you want to ask the user.</p> <pre><question>Enter the LDAP server</question></pre>	The default value is the path to the element (the path often looks strange, so we recommend entering a question).
<u>default</u>	<p>Set a preselection for the user. A text entry will be filled out with this value. A check box will be true or false and a selection will have the given value preselected.</p> <pre><default>dc=suse,dc=de</default></pre>	Optional.
<u>help</u>	<p>An optional help text that is shown on the left side of the question.</p> <pre><help>Enter the LDAP server address.</help></pre>	Optional.
<u>title</u>	<p>An optional title that is shown above the questions.</p> <pre><title>LDAP server</title></pre>	Optional.
<u>type</u>	<p>The type of the element you want to change. Possible values are <u>symbol</u>, <u>boolean</u>, <u>string</u> and <u>integer</u>. The file system in the partition section is a symbol, while</p>	Optional. The default is <u>string</u> . If type is <u>symbol</u> , you must provide the selection element too (see below).

Element	Description	Comment
	<p>the <u>encrypted</u> element in the user configuration is a boolean. You can see the type of that element if you look in your control file at the <u>config:type="..."</u> attribute. You can also use <u>static_text</u> as type. A <u>static_text</u> is a text that does not require any user input and can show information not included in the help text.</p> <pre data-bbox="600 864 991 925" style="border: 1px solid gray; padding: 2px;"><type>symbol</type></pre>	
<u>password</u>	<p>If this boolean is set to <u>true</u>, a password dialog pops up instead of a simple text entry. Setting this to <u>true</u> only makes sense if <u>type</u> is string.</p> <pre data-bbox="600 1263 991 1397" style="border: 1px solid gray; padding: 2px;"><password config:type="boolean">true</ password></pre>	Optional. The default is <u>false</u> .
<u>pathlist</u>	<p>A list of <u>path</u> elements. A path is a comma separated list of elements that describes the path to the element you want to change. For example, the LDAP server element can be found in the control file in the <u><ldap><ldap_server></u></p>	This information is optional but you should at least provide <u>path</u> or <u>file</u> .

Element	Description	Comment
	<p>section. So to change that value, you need to set the path to <code>ldap,ldap_server</code>.</p> <pre data-bbox="603 389 994 667"> <pathlist config:type="list"> <path>networking,dns,hostname</ path> <path>...</path> </pathlist> </pre> <p>To change the password of the first user in the control file, you need to set the path to <code>users,0,user_password</code>. The <code>0</code> indicates the first user in the <code><users config:type="list"></code> list of users in the control file. <code>1</code> would be the second one, and so on.</p> <pre data-bbox="603 1140 994 1834"> <users config:type="list"> <user> <username>root</ username> <user_password>password to change</user_password> <encrypted config:type="boolean">>false</ encrypted> </user> <user> <username>tux</ username> <user_password>password to change</user_password> <encrypted config:type="boolean">>false</ encrypted> </user> </pre>	

Element	Description	Comment
	<pre data-bbox="598 224 992 268"></users></pre>	
<p data-bbox="183 309 255 340"><u>file</u></p>	<p data-bbox="598 309 992 1099">You can store the answer to a question in a file, to use it in one of your scripts later. If you ask during <u>stage=initial</u> and you want to use the answer in stage 2, then you need to copy the answer-file in a chroot script that is running as <u>chroot-ed=false</u>. Use the command: <u>cp /tmp/my_answer /mnt/tmp/</u>. The reason is that <u>/tmp</u> in stage 1 is in the RAM disk and will be lost after the reboot, but the installed system is already mounted at <u>/mnt/</u>.</p> <pre data-bbox="598 1137 992 1227"><file>/tmp/answer_hostname</file></pre>	<p data-bbox="1015 309 1410 439">This information is optional, but you should at least provide <u>path</u> or <u>file</u>.</p>
<p data-bbox="183 1270 252 1301">stage</p>	<p data-bbox="598 1270 992 1816">Stage configures the installation stage in which the question pops up. You can set this value to <u>cont</u> or <u>initial</u>. <u>initial</u> means the pop-up comes up very early in the installation, shortly after the pre-script has run. <u>cont</u> means, that the dialog with the question comes after the first reboot when the system boots for the very first</p>	<p data-bbox="1015 1270 1410 1346">Optional. The default is <u>initial</u>.</p>

Element	Description	Comment
	<p>time. Questions you answer during the <u>initial</u> stage will write their answer into the control file on the hard disk. You should know that if you enter clear text passwords during <u>initial</u>. Of course it does not make sense to ask for the file system to use during the <u>cont</u> phase. The hard disk is already partitioned at that stage and the question will have no effect.</p> <pre data-bbox="600 864 991 925" style="border: 1px solid black; padding: 5px;"> <stage>cont</stage> </pre>	
<p><u>selection</u></p>	<p>The selection element contains a list of <u>entry</u> elements. Each entry represents a possible option for the user to choose. The user cannot enter a value in a text box, but they can choose from a list of values.</p> <pre data-bbox="600 1361 991 1841" style="border: 1px solid black; padding: 5px;"> <selection config:type="list"> <entry> <value> btrfs </value> <label> Btrfs File System </label> </entry> <entry> <value> ext3 </pre>	<p>Optional for <u>type=string</u>, not possible for <u>type=boolean</u> and mandatory for <u>type=symbol</u>.</p>

Element	Description	Comment
	<pre data-bbox="608 230 989 488"> </value> <label> Extended3 File System </label> </entry> </selection> </pre>	
<u>dialog</u>	<p data-bbox="600 528 997 846">You can ask more than one question per dialog. To do so, specify the dialog-id with an integer. All questions with the same dialog-id belong to the same dialog. The dialogs are sorted by the id too.</p> <pre data-bbox="608 880 989 1003"> <dialog config:type="integer">3</ dialog> </pre>	Optional.
<u>element</u>	<p data-bbox="600 1043 997 1317">You can have more than one question per dialog. To make that possible you need to specify the <u>element-id</u> with an integer. The questions in a dialog are sorted by ID.</p> <pre data-bbox="608 1350 989 1473"> <element config:type="integer">1</ element> </pre>	Optional (see dialog).
<u>width</u>	<p data-bbox="600 1514 997 1832">You can increase the default width of the dialog. If there are multiple width specifications per dialog, the largest one is used. The number is roughly equivalent to the number of characters.</p>	Optional.

Element	Description	Comment
	<pre data-bbox="603 241 986 344"><width config:type="integer">50</ width></pre>	
<u>height</u>	<p data-bbox="603 398 986 712">You can increase the default height of the dialog. If there are multiple height specifications per dialog, the largest one is used. The number is roughly equivalent to the number of lines.</p> <pre data-bbox="603 748 986 869"><height config:type="integer">15</ height></pre>	Optional.
<u>frametitle</u>	<p data-bbox="603 922 986 1326">You can have more than one question per dialog. Each question on a dialog has a frame that can have a frame title, a small caption for each question. You can put multiple elements into one frame. They need to have the same frame title.</p> <pre data-bbox="603 1361 986 1451"><frametitle>User data</ frametitle></pre>	Optional; default is no frame title.
<u>script</u>	<p data-bbox="603 1496 986 1720">You can run scripts after a question has been answered. See the table below for detailed instructions about scripts.</p> <pre data-bbox="603 1756 986 1800"><script>...</script></pre>	Optional; default is no script.

Element	Description	Comment
<u>ok_label</u>	<p>You can change the label on the <i>Ok</i> button. The last element that specifies the label for a dialog wins.</p> <pre data-bbox="600 456 991 517" style="border: 1px solid gray; padding: 5px;"><ok_label>Finish</ok_label></pre>	Optional.
<u>back_label</u>	<p>You can change the label on the <i>Back</i> button. The last element that specifies the label for a dialog wins.</p> <pre data-bbox="600 763 991 857" style="border: 1px solid gray; padding: 5px;"><back_label>change values</back_label></pre>	Optional.
<u>timeout</u>	<p>You can specify an integer here that is used as time-out in seconds. If the user does not answer the question before the timeout, the default value is taken as answer. When the user touches or changes any widget in the dialog, the timeout is turned off and the dialog needs to be confirmed via <i>Ok</i>.</p> <pre data-bbox="600 1440 991 1570" style="border: 1px solid gray; padding: 5px;"><timeout config:type="integer">30</timeout></pre>	Optional; a missing value is interpreted as <u>0</u> , which means that there is no time-out.
<u>default_value_script</u>	<p>You can run scripts to set the default value for a question (see Section 4.32.1, "Default Value Scripts" for detailed instructions about default val-</p>	Optional; default is no script.

Element	Description	Comment
	<p>ue scripts). This feature is useful if you can <u>calculate</u> a default value, especially in combination with the <u>time-out</u> option.</p> <pre><default_value_script>...</default_value_script></pre>	

4.32.1 Default Value Scripts

You can run scripts to set the default value for a question. This feature is useful if you can calculate a default value, especially in combination with the timeout option.

The elements listed below must be placed within the following XML structure:

```
<general>
  <ask-list config:type="list">
    <ask>
      <default_value_script>
        ...
      </default_value_script>
    </ask>
  </ask-list>
</general>
```

TABLE 4.4: DEFAULT VALUE SCRIPTS: XML REPRESENTATION

Element	Description	Comment
<u>source</u>	The source code of the script. Whatever you <u>echo</u> to STDOUT will be used as default value for the ask-dialog. If your script has an exit code other than 0, the normal default element is used. Take care you use <u>echo -n</u> to	This value is required, otherwise nothing would be executed.

Element	Description	Comment
	suppress the <code>\n</code> and that you echo reasonable values and not “okay” for a boolean <pre><source>...</source></pre>	
<u>interpreter</u>	The interpreter to use. <pre><interpreter>perl</interpreter></pre>	The default value is <code>shell</code> . You can also set <code>/bin/myinterpreter</code> as value.

4.32.2 Scripts

You can run scripts after a question has been answered.

The elements listed below must be placed within the following XML structure:

```
<general>
  <ask-list config:type="list">
    <ask>
      <script>
        ...
      </script>
    </ask>
  </ask-list>
</general>
```

TABLE 4.5: SCRIPTS: XML REPRESENTATION

Element	Description	Comment
<u>file name</u>	The file name of the script. <pre><filename>my_ask_script.sh</filename></pre>	The default is <code>ask_script.sh</code>
<u>source</u>	The source code of the script. Together with <code>re-run_on_error</code> activated, you check the value that	This value is required, otherwise nothing would be executed.

Element	Description	Comment
	<p>was entered for sanity. Your script can create a file <code>/tmp/next_dialog</code> with a dialog id specifying the next dialog AutoYaST will raise. A value of -1 terminates the ask sequence. If that file is not created, AutoYaST will run the dialogs in the normal order (since 11.0 only).</p> <pre data-bbox="600 719 991 779" style="border: 1px solid gray; padding: 5px;"><source>...</source></pre>	
<u>environment</u>	<p>A boolean that passes the value of the answer to the question as an environment variable to the script. The variable is named <code>VAL</code>.</p> <pre data-bbox="600 1077 991 1205" style="border: 1px solid gray; padding: 5px;"><environment config:type="boolean">true</environment></pre>	Optional. Default is <u>false</u> .
<u>feedback</u>	<p>A boolean that turns on feedback for the script execution. STDOUT will be displayed in a pop-up window that must be confirmed after the script execution.</p> <pre data-bbox="600 1550 991 1677" style="border: 1px solid gray; padding: 5px;"><feedback config:type="boolean">true</feedback></pre>	Optional, default is <u>false</u> .
<u>debug</u>	<p>A boolean that turns on debugging for the script execution.</p>	Optional, default is <u>true</u> . This value needs <u>feedback</u> to be turned on, too.

Element	Description	Comment
	<pre><debug config:type="boolean">true</ debug></pre>	
<u>rerun_on_error</u>	<p>A boolean that keeps the dialog open until the script has an exit code of 0 (zero). So you can parse and check the answers the user gave in the script and display an error with the <u>feedback</u> option.</p> <pre><rerun_on_error config:type="boolean">true</ rerun_on_error></pre>	<p>Optional, default is <u>false</u>. This value should be used together with the <u>feedback</u> option.</p>

Below you can see an example of the usage of the ask feature.

```
<general>
<ask-list config:type="list">
  <ask>
    <pathlist config:type="list">
      <path>ldap,ldap_server</path>
    </pathlist>
    <stage>cont</stage>
    <help>Choose your server depending on your department</help>
    <selection config:type="list">
      <entry>
        <value>ldap1.mydom.de</value>
        <label>LDAP for development</label>
      </entry>
      <entry>
        <value>ldap2.mydom.de</value>
        <label>LDAP for sales</label>
      </entry>
    </selection>
    <default>ldap2.mydom.de</default>
    <default_value_script>
      <source> <![CDATA[
echo -n "ldap1.mydom.de"
]]>
```

```

    </source>
  </default_value_script>
</ask>
<ask>
  <pathlist config:type="list">
    <path>networking,dns,hostname</path>
  </pathlist>
  <question>Enter Hostname</question>
  <stage>initial</stage>
  <default>enter your hostname here</default>
</ask>
<ask>
  <pathlist config:type="list">
    <path>partitioning,0,partitions,0,filesystem</path>
  </pathlist>
  <question>File System</question>
  <type>symbol</type>
  <selection config:type="list">
    <entry>
      <value config:type="symbol">ext4</value>
      <label>default File System (recommended)</label>
    </entry>
    <entry>
      <value config:type="symbol">ext3</value>
      <label>Fallback File System</label>
    </entry>
  </selection>
</ask>
</ask-list>
</general>

```

The following example shows a to choose between AutoYaST control files. AutoYaST will read the modified.xml file again after the ask-dialogs are done. This way you can fetch a complete new control file.

```

<general>
  <ask-list config:type="list">
    <ask>
      <selection config:type="list">
        <entry>
          <value>part1.xml</value>
          <label>Simple partitioning</label>
        </entry>
        <entry>
          <value>part2.xml</value>
          <label>encrypted /tmp</label>
        </entry>
      </selection>
    </ask>
  </ask-list>
</general>

```

```

    <entry>
      <value>part3.xml</value>
      <label>LVM</label>
    </entry>
  </selection>
  <title>XML Profile</title>
  <question>Choose a profile</question>
  <stage>initial</stage>
  <default>part1.xml</default>
  <script>
    <filename>fetch.sh</filename>
    <environment config:type="boolean">>true</environment>
    <source>
<![CDATA[
wget http://10.10.0.162/$VAL -O /tmp/profile/modified.xml 2>/dev/null
]]>
    </source>
    <debug config:type="boolean">>false</debug>
    <feedback config:type="boolean">>false</feedback>
  </script>
</ask>tion>
</ask-list>
</general>

```

You can verify the answer of a question with a script like this:

```

<general>
  <ask-list config:type="list">
    <ask>
      <script>
        <filename>my.sh</filename>
        <rerun_on_error config:type="boolean">>true</rerun_on_error>
        <environment config:type="boolean">>true</environment>
        <source><![CDATA[
if [ "$VAL" = "myhost" ]; then
  echo "Illegal Hostname!";
  exit 1;
fi
exit 0
]]>
        </source>
        <debug config:type="boolean">>false</debug>
        <feedback config:type="boolean">>true</feedback>
      </script>
      <dialog config:type="integer">0</dialog>
      <element config:type="integer">0</element>
      <pathlist config:type="list">
        <path>networking,dns,hostname</path>

```

```
</pathlist>
<question>Enter Hostname</question>
<default>enter your hostname here</default>
</ask>
</ask-list>
</general>
```

4.33 Kernel Dumps

With Kdump the system can create crashdump files if the whole kernel crashes. Crash dump files contain the memory contents while the system crashed. Such core files can be analyzed later by support or a (kernel) developer to find the reason for the system crash. Kdump is mostly useful for servers where you cannot easily reproduce such crashes but it is important to get the problem fixed.

There is a downside to this. Enabling Kdump requires between 64 MB and 128 MB of additional system RAM reserved for Kdump in case the system crashes and the dump needs to be generated. This section only describes how to set up Kdump with AutoYaST. It does not describe how Kdump works. For details, refer to the `kdump(7)` manual page.

The following example shows a general Kdump configuration.

EXAMPLE 4.63: KDUMP CONFIGURATION

```
<kdump>
<!-- memory reservation -->
<add_crash_kernel config:type="boolean">true</add_crash_kernel>
<crash_kernel>256M-:64M</crash_kernel>
<general>

<!-- dump target settings -->
<KDUMP_SAVEDIR>ftp://stravinsky.suse.de/incoming/dumps</KDUMP_SAVEDIR>
<KDUMP_COPY_KERNEL>true</KDUMP_COPY_KERNEL>
<KDUMP_FREE_DISK_SIZE>64</KDUMP_FREE_DISK_SIZE>
<KDUMP_KEEP_OLD_DUMPS>5</KDUMP_KEEP_OLD_DUMPS>

<!-- filtering and compression -->
<KDUMP_DUMPFORMAT>compressed</KDUMP_DUMPFORMAT>
<KDUMP_DUMPLEVEL>1</KDUMP_DUMPLEVEL>

<!-- notification -->
<KDUMP_NOTIFICATION_TO>tux@example.com</KDUMP_NOTIFICATION_TO>
<KDUMP_NOTIFICATION_CC>spam@example.com devnull@example.com</KDUMP_NOTIFICATION_CC>
```



```

<KDUMP_SMTP_SERVER>mail.example.com</KDUMP_SMTP_SERVER>
<KDUMP_SMTP_USER></KDUMP_SMTP_USER>
<KDUMP_SMTP_PASSWORD></KDUMP_SMTP_PASSWORD>

<!-- kdump kernel -->
<KDUMP_KERNELVER></KDUMP_KERNELVER>
<KDUMP_COMMANDLINE></KDUMP_COMMANDLINE>
<KDUMP_COMMANDLINE_APPEND></KDUMP_COMMANDLINE_APPEND>

<!-- expert settings -->
<KDUMP_IMMEDIATE_REBOOT>yes</KDUMP_IMMEDIATE_REBOOT>
<KDUMP_VERBOSE>15</KDUMP_VERBOSE>
<KEXEC_OPTIONS></KEXEC_OPTIONS>
</general>
</kdump>

```

4.33.1 Memory Reservation

The first step is to reserve memory for Kdump at boot-up. Because the memory must be reserved very early during the boot process, the configuration is done via a kernel command line parameter called `crashkernel`. The reserved memory will be used to load a second kernel which will be executed without rebooting if the first kernel crashes. This second kernel has a special `initrd`, which contains all programs necessary to save the dump over the network or to disk, send a notification e-mail, and finally reboot.

To reserve memory for Kdump, specify the `amount` (such as `64M` to reserve 64 MB of memory from the RAM) and the `offset`. The syntax is `crashkernel=AMOUNT@OFFSET`. The kernel can auto-detect the right offset (except for the Xen hypervisor, where you need to specify `16M` as offset). The amount of memory that needs to be reserved depends on architecture and main memory. Refer to *Book "System Analysis and Tuning Guide", Chapter 17 "Kexec and Kdump", Section 17.7.1 "Manual Kdump Configuration"* for recommendations on the amount of memory to reserve for Kdump.

You can also use the extended command line syntax to specify the amount of reserved memory depending on the System RAM. That is useful if you share one AutoYaST control file for multiple installations or if you often remove or install memory on one machine. The syntax is:

```
BEGIN_RANGE_1-END_RANGE_1:AMOUNT_1,BEGIN_RANGE_2-END_RANGE_2:AMOUNT_2@OFFSET
```

BEGIN_RANGE_1 is the start of the first memory range (for example: 0M) and END_RANGE_1 is the end of the first memory range (can be empty in case infinity should be assumed) and so on. For example, 256M-2G:64M,2G-:128M reserves 64 MB of crashkernel memory if the system has between 256 MB and 2 GB RAM and reserves 128 MB of crashkernel memory if the system has more than 2 GB RAM.

On the other hand, it is possible to specify multiple values for the crashkernel parameter. For example, when you need to reserve different segments of low and high memory, use values like 72M,low and 256M,high:

EXAMPLE 4.64: **KDUMP MEMORY RESERVATION WITH MULTIPLE VALUES**

```
<kdump>
  <!-- memory reservation (high and low) -->
  <add_crash_kernel config:type="boolean">true</add_crash_kernel>
  <crash_kernel config:type="list">
    <listentry>72M,low</listentry>
    <listentry>256M,high</listentry>
  </crash_kernel>
</kdump>
```

The following table shows the settings necessary to reserve memory:

TABLE 4.6: **KDUMP MEMORY RESERVATION SETTINGS:XML REPRESENTATION**

Element	Description	Comment
<u>add_crash_kernel</u>	Set to <u>true</u> if memory should be reserved and Kdump enabled. <pre><add_crash_kernel config:type="boolean">true</ add_crash_kernel></pre>	required
<u>crash_kernel</u>	Use the syntax of the crashkernel command line as discussed above. <pre><crash_kernel>256M:64M</ crash_kernel></pre> <p>A list of values is also supported.</p>	required

Element	Description	Comment
	<pre><crash_kernel config:type="list"> <listentry>72M,low</ listentry> <listentry>256M,high</ listentry> </crash_kernel></pre>	

4.33.2 Dump Saving

This section describes where and how crash dumps will be stored.

4.33.2.1 Target

The element `KDUMP_SAVEDIR` specifies the URL to where the dump is saved. The following methods are possible:

- `file` to save to the local disk,
- `ftp` to save to an FTP server (without encryption),
- `sftp` to save to an SSH2 SFTP server,
- `nfs` to save to an NFS location and
- `cifs` to save the dump to a CIFS/SMB export from Samba or Microsoft Windows.

For details see the `kdump(5)` manual page. Two examples are: `file:///var/crash` (which is the default location according to FHS) and `ftp://user:password@host:port/incoming/dumps`. A subdirectory, with the time stamp contained in the name, will be created and the dumps saved there.

When the dump is saved to the local disk, `KDUMP_KEEP_OLD_DUMPS` can be used to delete old dumps automatically. Set it to the number of old dumps that should be kept. If the target partition would end up with less free disk space than specified in `KDUMP_FREE_DISK_SIZE`, the dump is not saved.

To save the whole kernel and the debug information (if installed) to the same directory, set `KDUMP_COPY_KERNEL` to `true`. You will have everything you need to analyze the dump in one directory (except kernel modules and their debugging information).

4.33.2.2 Filtering and Compression

The kernel dump is uncompressed and unfiltered. It can get as large as your system RAM. To get smaller files, compress the dump file afterward. The dump needs to be decompressed before opening.

To use page compression, which compresses every page and allows dynamic decompression with the `crash(8)` debugging tool, set `KDUMP_DUMPFORMAT` to `compressed` (default).

You may not want to save all memory pages, for example those filled with zeroes. To filter the dump, set the `KDUMP_DUMPLEVEL`. 0 produces a full dump and 31 is the smallest dump. The manual pages `kdump(5)` and `makedumpfile(8)` list for each value which pages will be saved.

4.33.2.3 Summary

TABLE 4.7: DUMP TARGET SETTINGS: XML REPRESENTATION

Element	Description	Comment
<code>KDUMP_SAVEDIR</code>	A URL that specifies the target to which the dump and related files will be saved. <pre><KDUMP_SAVEDIR>file:///var/crash/</KDUMP_SAVEDIR></pre>	required
<code>KDUMP_COPY_KERNEL</code>	Set to <code>true</code> , if not only the dump should be saved to <code>KDUMP_SAVEDIR</code> but also the kernel and its debugging information (if installed). <pre><KDUMP_COPY_KERNEL>false</KDUMP_COPY_KERNEL></pre>	optional
<code>KDUMP_FREE_DISK_SIZE</code>	Disk space in megabytes that must remain free after saving the dump. If not enough space is available, the dump will not be saved.	optional

Element	Description	Comment
	<KDUMP_FREE_DISK_SIZE>64</KDUMP_FREE_DISK_SIZE>	
<u>KDUMP_KEEP_OLD_DUMPS</u>	<p>The number of dumps that are kept (not deleted) if <u>KDUMP_SAVEDIR</u> points to a local directory. Specify 0 if you do not want any dumps to be automatically deleted, specify -1 if all dumps except the current one should be deleted.</p> <p><KDUMP_KEEP_OLD_DUMPS>4</KDUMP_KEEP_OLD_DUMPS></p>	optional

4.33.3 E-Mail Notification

Configure e-mail notification to be informed when a machine crashes and a dump is saved.

Because Kdump runs in the initrd, a local mail server cannot send the notification e-mail. An SMTP server needs to be specified (see below).

You need to provide exactly one address in KDUMP_NOTIFICATION_TO. More addresses can be specified in KDUMP_NOTIFICATION_CC. Only use e-mail addresses in both cases, not a real name. Specify KDUMP_SMTP_SERVER and (if the server needs authentication) KDUMP_SMTP_USER and KDUMP_SMTP_PASSWORD. Support for TLS/SSL is not available but may be added in the future.

TABLE 4.8: E-MAIL NOTIFICATION SETTINGS: XML REPRESENTATION

Element	Description	Comment
<u>KDUMP_NOTIFICATION_TO</u>	Exactly one e-mail address to which the e-mail should be sent. Additional recipients can be specified in <u>KDUMP_NOTIFICATION_CC</u> .	optional (notification disabled if empty)

Element	Description	Comment
	<pre><KDUMP_NOTIFICATION_TO >tux@example.com</ KDUMP_NOTIFICATION_TO></pre>	
<u>KDUMP_NOTIFICATION_CC</u>	<p>Zero, one or more recipients that are in the cc line of the notification e-mail.</p> <pre><KDUMP_NOTIFICATION_CC >wilber@example.com geeko@example.com</ KDUMP_NOTIFICATION_CC></pre>	optional
<u>KDUMP_SMTP_SERVER</u>	<p>Host name of the SMTP server used for mail delivery. SMTP authentication is supported (see <u>KDUMP_SMTP_USER</u> and <u>KDUMP_SMTP_PASSWORD</u>) but TLS/SSL are not.</p> <pre><KDUMP_SMTP_SERVER>email.suse.de</ KDUMP_SMTP_SERVER></pre>	optional (notification disabled if empty)
<u>KDUMP_SMTP_USER</u>	<p>User name used together with <u>KDUMP_SMTP_PASSWORD</u> for SMTP authentication.</p> <pre><KDUMP_SMTP_USER>bwalle</ KDUMP_SMTP_USER></pre>	optional
<u>KDUMP_SMTP_PASSWORD</u>	<p>Password used together with <u>KDUMP_SMTP_USER</u> for SMTP authentication.</p> <pre><KDUMP_SMTP_PASSWORD>geheim</ KDUMP_SMTP_PASSWORD></pre>	optional

4.33.4 Kdump Kernel Settings

As already mentioned, a special kernel is booted to save the dump. If you do not want to use the auto-detection mechanism to find out which kernel is used (see the `kdump(5)` manual page that describes the algorithm which is used to find the kernel), you can specify the version of a custom kernel in `KDUMP_KERNELVER`. If you set it to `foo`, then the kernel located in `/boot/vmlinuz-foo` or `/boot/vmlinux-foo` (in that order on platforms that have a `vmlinuz` file) will be used.

You can specify the command line used to boot the Kdump kernel. Normally the boot command line is used, minus settings that are not relevant for Kdump (like the `crashkernel` parameter) plus some settings needed by Kdump (see the manual page `kdump(5)`). To specify additional parameters, use `KDUMP_COMMANDLINE_APPEND`. If you know what you are doing and you want to specify the entire command line, set `KDUMP_COMMANDLINE`.

TABLE 4.9: KERNEL SETTINGS: XML REPRESENTATION

Element	Description	Comment
<code>KDUMP_KERNELVER</code>	Version string for the kernel used for Kdump. Leave it empty to use the auto-detection mechanism (strongly recommended). <pre><KDUMP_KERNELVER >2.6.27-default</ KDUMP_KERNELVER></pre>	optional (auto-detection if empty)
<code>KDUMP_COMMANDLINE_APPEND</code>	Additional command line parameters for the Kdump kernel. <pre><KDUMP_COMMANDLINE_APPEND >console=ttyS0,57600</ KDUMP_COMMANDLINE_APPEND></pre>	optional

Element	Description	Comment
<u>KDUMP_Command Line</u>	<p>Overwrite the automatically generated Kdump command line. Use with care. Usually, <u>KDUMP_COMMANDLINE_APPEND</u> should suffice.</p> <pre><KDUMP_COMMANDLINE_APPEND >root=/dev/sda5 maxcpus=1 irqpoll</ KDUMP_COMMANDLINE></pre>	optional

4.33.5 Expert Settings

TABLE 4.10: EXPERT SETTINGS: XML REPRESENTATIONS

Element	Description	Comment
<u>KDUMP_IMMEDIATE_REBOOT</u>	<p><u>true</u> if the system should be rebooted automatically after the dump has been saved, <u>false</u> otherwise. The default is to reboot the system automatically.</p> <pre><KDUMP_IMMEDIATE_REBOOT >true</ KDUMP_IMMEDIATE_REBOOT></pre>	optional
<u>KDUMP_VERBOSE</u>	<p>Bitmask that specifies how verbose the Kdump process should be. Read <code>kdump(5)</code> for details.</p> <pre><KDUMP_VERBOSE>3</ KDUMP_VERBOSE></pre>	optional

Element	Description	Comment
<u>KEXEC_OPTIONS</u>	<p>Additional options that are passed to kexec when loading the Kdump kernel. Normally empty.</p> <pre><KEXEC_OPTIONS>--noio</KEXEC_OPTIONS></pre>	optional

4.34 DNS Server

The Bind DNS server can be configured by adding a dns-server resource. The three more straightforward properties of that resource can have a value of 1 to enable them or 0 to disable.

Attribute	Value	Description
<u>chroot</u>	0 / 1	The DNS server must be jailed in a chroot.
<u>start_service</u>	0 / 1	Bind is enabled (executed on system start).
<u>use_ldap</u>	0 / 1	Store the settings in LDAP instead of native configuration files.

EXAMPLE 4.65: BASIC DNS SERVER SETTINGS

```
<dns-server>
  <chroot>0</chroot>
  <start_service>1</start_service>
  <use_ldap>0</use_ldap>
</dns-server>
```

In addition to those basic settings, there are three properties of type list that can be used to fine-tune the service configuration.

List	Description
<u>logging</u>	Options of the DNS server logging.
<u>options</u>	Bind options like the files and directories to use, the list of forwarders and other configuration settings.
<u>zones</u>	List of DNS zones known by the server, including all the settings, records and SOA records.

EXAMPLE 4.66: CONFIGURING DNS SERVER ZONES AND ADVANCED SETTINGS

```

<dns-server>
  <logging config:type="list">
    <listentry>
      <key>channel</key>
      <value>log_syslog { syslog; }</value>
    </listentry>
  </logging>
  <options config:type="list">
    <option>
      <key>forwarders</key>
      <value>{ 10.10.0.1; }</value>
    </option>
  </options>
  <zones config:type="list">
    <listentry>
      <is_new>1</is_new>
      <modified>1</modified>
      <options config:type="list"/>
      <records config:type="list">
        <listentry>
          <key>mydom.uwe.</key>
          <type>MX</type>
          <value>0 mail.mydom.uwe.</value>
        </listentry>
        <listentry>
          <key>mydom.uwe.</key>
          <type>NS</type>
          <value>ns.mydom.uwe.</value>
        </listentry>
      </records>
    <soa>

```

```

<expiry>1w</expiry>
<mail>root.aaa.aaa.cc.</mail>
<minimum>1d</minimum>
<refresh>3h</refresh>
<retry>1h</retry>
<serial>2005082300</serial>
<server>aaa.aaa.cc.</server>
<zone>@</zone>
</soa>
<soa_modified>1</soa_modified>
<ttl>2d</ttl>
<type>master</type>
<update_actions config:type="list">
  <listentry>
    <key>mydom.uwe.</key>
    <operation>add</operation>
    <type>NS</type>
    <value>ns.mydom.uwe.</value>
  </listentry>
</update_actions>
<zone>mydom.uwe</zone>
</listentry>
</zones>
</dns-server>

```

4.35 DHCP Server

The `dhcp-server` resource makes it possible to configure all the settings of a DHCP server by means of the six following properties.

Element	Value	Description
<code>chroot</code>	0 / 1	A value of 1 means that the DHCP server must be jailed in a chroot.
<code>start_service</code>	0 / 1	Set this to 1 to enable the DHCP server (that is, run it on system startup).

Element	Value	Description
<u>use_ldap</u>	0 / 1	If set to 1, the settings will be stored in LDAP instead of native configuration files.
<u>other_options</u>	Text	String with parameters that will be passed to the DHCP server executable when started. For example, use "-p 1234" to listen on a non-standard 1234 port. For all possible options, consult the dhcpd manual page. If left blank, default values will be used.
<u>allowed_interfaces</u>	List	List of network cards in which the DHCP server will be operating. See the example below for the exact format.
<u>settings</u>	List	List of settings to configure the behavior of the DHCP server. The configuration is defined in a tree-like structure where the root represents the global options, with subnets and host nested from there. The <u>children</u> , <u>parent_id</u> and <u>parent_type</u> properties are used to represent that nesting. See the example below for the exact format.

EXAMPLE 4.67: EXAMPLE DHCP-SERVER SECTION

```
<dhcp-server>
  <allowed_interfaces config:type="list">
    <allowed_interface>eth0</allowed_interface>
  </allowed_interfaces>
  <chroot>0</chroot>
  <other_options>-p 9000</other_options>
  <start_service>1</start_service>
  <use_ldap>0</use_ldap>

  <settings config:type="list">
    <settings_entry>
      <children config:type="list"/>
      <directives config:type="list">
        <listentry>
          <key>fixed-address</key>
          <type>directive</type>
          <value>192.168.0.10</value>
        </listentry>
        <listentry>
          <key>hardware</key>
          <type>directive</type>
          <value>ethernet d4:00:00:bf:00:00</value>
        </listentry>
      </directives>
      <id>static10</id>
      <options config:type="list"/>
      <parent_id>192.168.0.0 netmask 255.255.255.0</parent_id>
      <parent_type>subnet</parent_type>
      <type>host</type>
    </settings_entry>
    <settings_entry>
      <children config:type="list">
        <child>
          <id>static10</id>
          <type>host</type>
        </child>
      </children>
      <directives config:type="list">
        <listentry>
          <key>range</key>
          <type>directive</type>
          <value>dynamic-bootp 192.168.0.100 192.168.0.150</value>
        </listentry>
        <listentry>
          <key>default-lease-time</key>
```

```

        <type>directive</type>
        <value>14400</value>
    </listentry>
    <listentry>
        <key>max-lease-time</key>
        <type>directive</type>
        <value>86400</value>
    </listentry>
</directives>
<id>192.168.0.0 netmask 255.255.255.0</id>
<options config:type="list"/>
<parent_id/>
<parent_type/>
<type>subnet</type>
</settings_entry>
<settings_entry>
    <children config:type="list">
        <child>
            <id>192.168.0.0 netmask 255.255.255.0</id>
            <type>subnet</type>
        </child>
    </children>
    <directives config:type="list">
        <listentry>
            <key>ddns-update-style</key>
            <type>directive</type>
            <value>none</value>
        </listentry>
        <listentry>
            <key>default-lease-time</key>
            <type>directive</type>
            <value>14400</value>
        </listentry>
    </directives>
</id>
<options config:type="list"/>
<parent_id/>
<parent_type/>
<type/>
</settings_entry>
</settings>
</dhcp-server>

```

4.36 Firewall Configuration

SuSEfirewall2 has been replaced by `firewalld` starting with openSUSE Leap 15.0. Profiles using SuSEfirewall2 properties will be translated to `firewalld` profiles. However, not all profile properties can be converted. For details about `firewalld`, refer to *Book “Security Guide”, Chapter 18 “Masquerading and Firewalls”, Section 18.4 “firewalld”*.



Important: Limited Backward Compatibility with SuSEFirewall2 Based Profiles

The use of SuSEFirewall2 based profiles will be only partially supported as many options are not valid in `firewalld` and some missing configuration could affect your network security.

4.36.1 General Firewall Configuration

In `firewalld` the general configuration only exposes a few properties and most of the configuration is done by zones.

Attribute	Value	Description
<code>start_firewall</code>	Boolean	Whether <code>firewalld</code> should be started right after applying the configuration.
<code>enable_firewall</code>	Boolean	Whether <code>firewalld</code> should be started on every system startup.
<code>default_zone</code>	Zone name	The default zone is used for everything that is not explicitly assigned.

Attribute	Value	Description
<u>log_denied_packets</u>	Type of dropped packages to be logged	Enable logging of dropped packages for the type selected. Values: <u>off</u> , <u>unicast</u> , <u>multicast</u> , <u>broadcast</u> , <u>all</u> .
<u>name</u>	Identifier of zone	Used to identify a zone. If the zone is not known yet, a new zone will be created.
<u>short</u>	Short summary of zone	Briefly summarizes the purpose of the zone. Ignored for already existing zones. If not specified, the name is used.
<u>description</u>	Description of zone	Describes the purpose of the zone. Ignored for already existing zones. If not specified, the name is used.
<u>target</u>	Default action	Defines the default action in the zone if no rule matches. Possible values are <u>ACCEPT</u> , <u>%%REJECT%%</u> , <u>DROP</u> and <u>default</u> . If not specified, <u>default</u> is used. For details about values, see https://firewalld.org/documentation/zone/options.html .

4.36.2 Firewall Zones Configuration

The configuration of `firewalld` is based on the existence of several zones which define the trust level for a connection, interface or source address. The behavior of each zone can be tweaked in several ways although not all the properties are exposed yet.

Attributes	Value	Description
<u>interfaces</u>	List of interface names	List of interface names assigned to this zone. Interfaces or sources can only be part of one zone.
<u>services</u>	List of services	List of services accessible in this zone.
<u>ports</u>	List of ports	List of single ports or ranges to be opened in the assigned zone.
<u>protocols</u>	List of protocols	List of protocols to be opened or be accessible in the assigned zone.
<u>masquerade</u>	Enable masquerade	It will enable or disable network address translation (NAT) in the assigned zone.

4.36.3 A Full Example

A full example of the firewall section, including general and zone specific properties could look like this.

EXAMPLE 4.68: EXAMPLE FIREWALL SECTION

```
<firewall>
  <enable_firewall>true</enable_firewall>
  <log_denied_packets>all</log_denied_packets>
  <default_zone>external</default_zone>
  <zones config:type="list">
    <zone>
      <name>public</name>
      <interfaces config:type="list">
        <interface>eth0</interface>
      </interfaces>
      <services config:type="list">
```

```

<service>ssh</service>
<service>dhcp</service>
<service>dhcpv6</service>
<service>samba</service>
<service>vnc-server</service>
</services>
<ports config:type="list">
  <port>21/udp</port>
  <port>22/udp</port>
  <port>80/tcp</port>
  <port>443/tcp</port>
  <port>8080/tcp</port>
</ports>
</zone>
<zone>
  <name>dmz</name>
  <interfaces config:type="list">
    <interface>eth1</interface>
  </interfaces>
</zone>
</zones>
</firewall>

```

4.37 Miscellaneous Hardware and System Components

In addition to the core component configuration, like network authentication and security, AutoYaST offers a wide range of hardware and system configuration options, the same as available by default on any system installed manually and in an interactive way. For example, it is possible to configure printers, sound devices, TV cards and any other hardware components which have a module within YaST.

Any new configuration options added to YaST will be automatically available in AutoYaST.

4.37.1 Printer

AutoYaST support for printing is limited to basic settings defining how CUPS is used on a client for printing via the network.

There is no AutoYaST support for setting up local print queues. Modern printers are usually connected via USB. CUPS accesses USB printers by a model-specific device URI like `usb://ACME/FunPrinter?serial=1a2b3c`. Usually it is not possible to predict the correct USB device URI in advance, because it is determined by the CUPS back-end `usb` during runtime. Therefore it is not possible to set up local print queues with AutoYaST.

Basics on how CUPS is used on a client workstation to print via network:

On client workstations application programs submit print jobs to the CUPS daemon process (`cupsd`). `cupsd` forwards the print jobs to a CUPS print server in the network where the print jobs are processed. The server sends the printer specific data to the printer device.

If there is only a single CUPS print server in the network, there is no need to have a CUPS daemon running on each client workstation. Instead it is simpler to specify the CUPS server in `/etc/cups/client.conf` and access it directly (only one CUPS server entry can be set). In this case application programs that run on client workstations submit print jobs directly to the specified CUPS print server.

Example 4.69, "Printer configuration" shows a `printer` configuration section. The `cupsd_conf_content` entry contains the whole verbatim content of the `cupsd` configuration file `/etc/cups/cupsd.conf`. The `client_conf_content` entry contains the whole verbatim content of `/etc/cups/client.conf`. The `printer` section contains the `cupsd` configuration but it does not specify whether the `cupsd` should run.

EXAMPLE 4.69: PRINTER CONFIGURATION

```
<printer>
  <client_conf_content>
    <file_contents><![CDATA[
... verbatim content of /etc/cups/client.conf ...
]]></file_contents>
  </client_conf_content>
  <cupsd_conf_content>
    <file_contents><![CDATA[
... verbatim content of /etc/cups/cupsd.conf ...
]]></file_contents>
  </cupsd_conf_content>
</printer>
```



Note: /etc/cups/cups-files.conf

With release 1.6 the CUPS configuration file has been split into two files: `cupsd.conf` and `cups-files.conf`. As of openSUSE Leap 15.2, AutoYaST only supports modifying `cupsd.conf` since the default settings in `cups-files.conf` are sufficient for usual printing setups.

4.37.2 Sound devices

An example of the sound configuration created using the configuration system is shown below.

EXAMPLE 4.70: SOUND CONFIGURATION

```
<sound>
  <autoinstall config:type="boolean">true</autoinstall>
  <modules_conf config:type="list">
    <module_conf>
      <alias>snd-card-0</alias>
      <model>M5451, ALI</model>
      <module>snd-ali5451</module>
      <options>
        <snd_enable>1</snd_enable>
        <snd_index>0</snd_index>
        <snd_pcm_channels>32</snd_pcm_channels>
      </options>
    </module_conf>
  </modules_conf>
  <volume_settings config:type="list">
    <listentry>
      <Master config:type="integer">75</Master>
    </listentry>
  </volume_settings>
</sound>
```

4.38 Importing SSH Keys and Configuration

YaST allows SSH keys and server configuration to be imported from previous installations. The behavior of this feature can also be controlled through an AutoYaST profile.

EXAMPLE 4.71: IMPORTING SSH KEYS AND CONFIGURATION FROM /DEV/SDA2

```
<ssh_import>
```

```

<import config:type="boolean">true</import>
<copy_config config:type="boolean">true</copy_config>
<device>/dev/sda2</device>
</ssh_import>

```

Attributes	Value	Description
<u>import</u>	true / false	SSH keys will be imported. If set to <u>false</u> , nothing will be imported.
<u>copy_config</u>	true / false	Additionally, SSH server configuration will be imported. This setting will not have effect if <u>import</u> is set to <u>false</u> .
<u>device</u>	Partition	Partition to import keys and configuration from. If it is not set, the partition which contains the most recently accessed key is used.

4.39 Configuration Management

AutoYaST allows delegating part of the configuration to a *configuration management tool* like Salt. AutoYaST takes care of the basic system installation (partitioning, network setup, etc.) and the remaining configuration tasks can be delegated.



Note: Only Salt is Officially Supported

Although Puppet is mentioned in this document, only Salt is officially supported. Nevertheless, feel free to report any problem you might find with Puppet.

AutoYaST supports two different approaches:

- Using a configuration management server. In this case, AutoYaST sets up a configuration management tool. It connects to a master server to get the instructions to configure the system.
- Getting the configuration from elsewhere (for example, an HTTP server or a flash disk like a USB stick) and running the configuration management tool in stand-alone mode.

4.39.1 Connecting to a Configuration Management Server

This approach is especially useful when a configuration management server (a *master* in Salt and Puppet jargon) is already in place. In this case, the hardest part might be to set up a proper authentication mechanism.

Both Salt and Puppet support the following authentication methods:

- Manual authentication on the fly. When AutoYaST starts the client, a new authentication request is generated. The administrator can manually accept this request on the server. AutoYaST will retry the connection. If the key was accepted meanwhile, AutoYaST continues the installation.
- Using a preseed key. Refer to the documentation of your configuration management system of choice to find out how to generate them. Use the `keys_url` option to tell AutoYaST where to look for them.

With the configuration example below, AutoYaST will launch the client to generate the authentication request. It will try to connect up to three times, waiting 15 seconds between each try.

EXAMPLE 4.72: CLIENT/SERVER WITH MANUAL AUTHENTICATION

```
<configuration_management>
  <type>salt</type>
  <master>my-salt-server.example.net</master>
  <auth_attempts config:type="integer">3</auth_attempts>
  <auth_time_out config:type="integer">15</auth_time_out>
</configuration_management>
```

However, with the following example, AutoYaST will retrieve the keys from a flash disk (for example, USB stick) and will use them to connect to the master server.

EXAMPLE 4.73: CLIENT/SERVER WITH PRESEED KEYS

```
<configuration_management>
  <type>salt</type>
  <master>my-salt-server.example.net</master>
  <keys_url>usb:/</keys_url>
</configuration_management>
```

The table below summarizes the supported options for these scenarios.

Attributes	Value	Description
<u>type</u>	String	Configuration management name. Currently only <u>salt</u> is officially supported.
<u>master</u>	String	Host name or IP address of the configuration management server.
<u>auth_attempts</u>	Integer	Maximum attempts to connect to the server. The default is three attempts.
<u>auth_time_out</u>	Integer	Time (in seconds) between attempts to connect to the server. The default is 15 seconds.
<u>keys_url</u>	URL of used key	Path to an HTTP server, hard disk, flash drive or similar with the files <u>default.key</u> and <u>default.pub</u> . This key must be known to the configuration management master.
<u>enable_services</u>	True/False	Enables the configuration management services on the client side after the installation. The default is <u>true</u> .

4.39.2 Running in Stand-alone Mode

For simple scenarios, deploying a configuration management server is unnecessary. Instead, use Salt or Puppet in *stand-alone* (or *masterless*) mode.

As there is no server, AutoYaST needs to know where to get the configuration from. Put the configuration into a TAR archive and store it anywhere (for example, on a flash drive, an HTTP/HTTPS server, an NFS/SMB share).

The TAR archive must have the same layout that is expected under `/srv` in a Salt server. It means that you need to place your Salt states in a `salt` directory and your formulas in a separate `formulas` directory.

Additionally, you can have a `pillar` directory containing the pillar data. Alternatively, you can provide that data in a separate TAR archive by using the `pillar_url` option.

EXAMPLE 4.74: STANDALONE MODE

```
<configuration_management>
  <type>salt</type>
  <states_url>my-salt-server.example.net</states_url>
  <pillar_url>my-salt-server.example.net</pillar_url>
</configuration_management>
```

Attributes	Value	Description
<code>type</code>	String	Configuration management name. Currently only <code>salt</code> is officially supported.
<code>states_url</code>	URL	Location of the Salt states TAR archive. It may include formulas and pillars. Files must be located in a <code>salt</code> directory.
<code>pillar_url</code>	URL	Location of the TAR archive that contains the pillars.
<code>modules_url</code>	URL	Location of Puppet modules.

4.39.3 SUSE Manager Salt Formulas Support

AutoYaST offers support for SUSE Manager Salt Formulas when running in stand-alone mode. In case a formula is found in the states TAR archive, AutoYaST displays a screen which allows the user to select and configure the formulas to apply.

Bear in mind that this feature defeats the AutoYaST purpose of performing an unattended installation, as AutoYaST will wait for the user's input.

III Managing Mass Installations with Rules and Classes

5 Rules and Classes 196

5 Rules and Classes

Rules and classes allow customizing installations for sets of machines in different ways:

- Rules allow configuring a system depending on its attributes.
- Classes represent configurations for groups of target systems. Classes can be assigned to systems.



Note: Use `autoyast` Boot Option Only

Rules and classes are only supported by the boot parameter `autoyast=URL`.

`autoyast2=URL` is not supported, because this option downloads a single AutoYaST control file only.

5.1 Rule-based Automatic Installation

Rules offer the possibility to configure a system depending on system attributes by merging multiple control files during installation. The rule-based installation is controlled by a rules file. For example, this could be useful to install systems in two departments in one go. Assume a scenario where machines in department A need to be installed as office desktops, whereas machines in department B need to be installed as developer workstations. You would create a rules file with two different rules. For each rule, you could use different system parameters to distinguish the installations from one another. Each rule would also contain a link to an appropriate profile for each department.

The rules file is an XML file containing rules for each group of systems (or single systems) that you want to automatically install. A set of rules distinguish a group of systems based on one or more system attributes. After passing all rules, each group of systems is linked to a control file. Both the rules file and the control files must be located in a pre-defined and accessible location. The rules file is retrieved only if no specific control file is supplied using the `autoyast` keyword. For example, if the following is used, the rules file will not be evaluated:

```
autoyast=http://10.10.0.1/profile/myprofile.xml
autoyast=http://10.10.0.1/profile/rules/rules.xml
```

Instead use:

```
autoyast=http://10.10.0.1/profile/
```

which will load `http://10.10.0.1/profile/rules/rules.xml` (the slash at the end of the directory name is important).

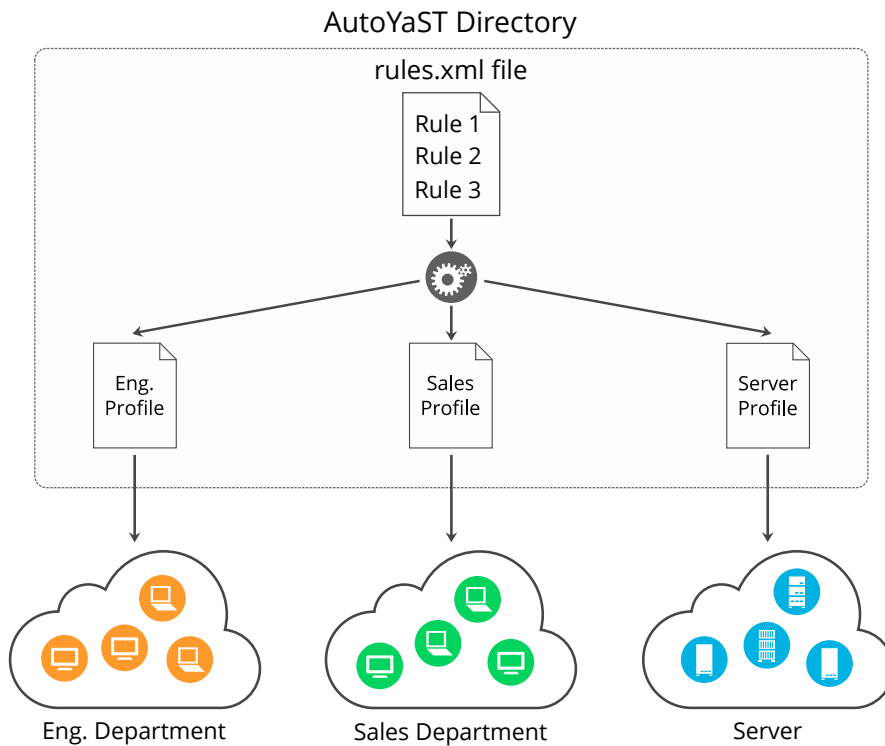


FIGURE 5.1: RULES

If more than one rule applies, the final control file for each group is generated on the fly using a merge script. The merging process is based on the order of the rules and later rules override configuration data in earlier rules. Note that the names of the top sections in the merged XML files need to be in alphabetical order for the merge to succeed.

The use of a rules file is optional. If the rules file is not found, system installation proceeds in the standard way by using the supplied control file or by searching for the control file depending on the MAC or the IP address of the system.

5.1.1 Rules File Explained

EXAMPLE 5.1: SIMPLE RULES FILE

The following simple example illustrates how the rules file is used to retrieve the configuration for a client with known hardware.

```
<?xml version="1.0"?>
```

```

<!DOCTYPE autoinstall>
<autoinstall xmlns="http://www.suse.com/1.0/yast2ns" xmlns:config="http://
www.suse.com/1.0/configs">
  <rules config:type="list">
    <rule>
      <disksize>
        <match>/dev/sdc 1000</match>
        <match_type>greater</match_type>
      </disksize>
      <result>
        <profile>department_a.xml</profile>
        <continue config:type="boolean">>false</continue>
      </result>
    </rule>
    <rule>
      <disksize>
        <match>/dev/sda 1000</match>
        <match_type>greater</match_type>
      </disksize>
      <result>
        <profile>department_b.xml</profile>
        <continue config:type="boolean">>false</continue>
      </result>
    </rule>
  </rules>
</autoinstall>

```

The last example defines two rules and provides a different control file for every rule. The rule used in this case is disksize. After parsing the rules file, YaST attempts to match the target system with the rules in the rules.xml file. A rule match occurs when the target system matches all system attributes defined in the rule. When the system matches a rule, the respective resource is added to the stack of control files AutoYaST will use to create the final control file. The continue property tells AutoYaST whether it should continue with other rules after a match has been found.

If the first rule does not match, the next rule in the list is examined until a match is found.

Using the disksize attribute, you can provide different configurations for systems with hard disks of different sizes. The first rule checks if the device /dev/sdc is available and if it is greater than 1 GB in size using the match property.

A rule must have at least one attribute to be matched. If you need to check more attributes, such as memory or architectures, you can add more attributes in the rule resource as shown in the next example.

EXAMPLE 5.2: SIMPLE RULES FILE

The following example illustrates how the rules file is used to retrieve the configuration for a client with known hardware.

```
<?xml version="1.0"?>
<!DOCTYPE autoinstall>
<autoinstall xmlns="http://www.suse.com/1.0/yast2ns" xmlns:config="http://
www.suse.com/1.0/configs">
  <rules config:type="list">
    <rule>
      <disksize>
        <match>/dev/sdc 1000</match>
        <match_type>greater</match_type>
      </disksize>
      <memsize>
        <match>1000</match>
        <match_type>greater</match_type>
      </memsize>
      <result>
        <profile>department_a.xml</profile>
        <continue config:type="boolean">>false</continue>
      </result>
    </rule>
    <rule>
      <disksize>
        <match>/dev/shda 1000</match>
        <match_type>greater</match_type>
      </disksize>
      <memsize>
        <match>256</match>
        <match_type>greater</match_type>
      </memsize>
      <result>
        <profile>department_b.xml</profile>
        <continue config:type="boolean">>false</continue>
      </result>
    </rule>
  </rules>
</autoinstall>
```

The rules directory must be located in the same directory specified via the `autoyast` keyword at boot time. If the client was booted using `autoyast=http://10.10.0.1/profiles/`, AutoYaST will search for the rules file at `http://10.10.0.1/profiles/rules/rules.xml`.

5.1.2 Custom Rules

If the attributes AutoYaST provides for rules are not enough for your purposes, use custom rules. Custom rules contain a shell script. The output of the script (STDOUT, STDERR is ignored) can be evaluated.

Here is an example for the use of custom rules:

```
<rule>
  <custom1>
    <script>
if grep -i intel /proc/cpuinfo > /dev/null; then
echo -n "intel"
else
echo -n "non_intel"
fi;
    </script>
    <match>*</match>
    <match_type>exact</match_type>
  </custom1>
  <result>
    <profile>@custom1.xml</profile>
    <continue config:type="boolean">>true</continue>
  </result>
</rule>
```

The script in this rule can echo either `intel` or `non_intel` to STDOUT (the output of the `grep` command must be directed to `/dev/null` in this case). The output of the rule script will be filled between the two '@' characters, to determine the file name of the control file to fetch. AutoYaST will read the output and fetch a file with the name `intel.xml` or `non_intel.xml`. This file can contain the AutoYaST profile part for the software selection; for example, in case you want a different software selection on Intel hardware than on others.

The number of custom rules is limited to five. So you can use `custom1` to `custom5`.

5.1.3 Match Types for Rules

You can use five different `match_types`:

- `exact` (default)
- `greater`
- `lower`

- range
- regex (a simple == operator like in Bash)

If using exact, the string must match exactly as specified. regex can be used to match substrings like ntel will match Intel, intel and intelligent. greater and lower can be used for memsize or totaldisk for example. They can match only with rules that return an integer value. A range is only possible for integer values too and has the form of value1-value2, for example 512-1024.

5.1.4 Combine Attributes

Multiple attributes can be combined via a logical operator. It is possible to let a rule match if disksize is greater than 1GB or memsize is exactly 512MB.

You can do this with the operator element in the rules.xml file. and and or are possible operators, and being the default. Here is an example:

```
<rule>
  <disksize>
    <match>/dev/sda 1000</match>
    <match_type>greater</match_type>
  </disksize>
  <memsize>
    <match>256</match>
    <match_type>greater</match_type>
  </memsize>
  <result>
    <profile>machine2.xml</profile>
    <continue config:type="boolean">>false</continue>
  </result>
  <operator>or</operator>
</rule>
```

5.1.5 Rules File Structure

The rules.xml file needs to:

- have at least one rule,
- have the name rules.xml,

- be located in the directory `rules` in the profile repository,
- have at least one attribute to match in the rule.

5.1.6 Predefined System Attributes

The following table lists the predefined system attributes you can match in the rules file.

If you are unsure about a value on your system, run `/usr/lib/YaST/bin/y2base ayast_probe ncurses`. The text box displaying the detected values can be scrolled. Note that this command will not work while another YaST process that requires a lock (for example the installer) is running. Therefore you cannot run it during the installation.

TABLE 5.1: SYSTEM ATTRIBUTES

Attribute	Values	Description
<code>hostaddress</code>	IP address of the host	This attribute must always match exactly.
<code>host name</code>	The name of the host	This attribute must always match exactly.
<code>domain</code>	Domain name of host	This attribute must always match exactly.
<code>installed_product</code>	The name of the product to be installed.	This attribute must always match exactly.
<code>installed_product_version</code>	The version of the product to be installed.	This attribute must always match exactly.
<code>network</code>	network address of host	This attribute must always match exactly.
<code>mac</code>	MAC address of host	This attribute must always match exactly (the MAC addresses should have the form <code>0080c8f6484c</code>).

Attribute	Values	Description
<u>linux</u>	Number of installed Linux partitions on the system	This attribute can be 0 or more.
<u>others</u>	Number of installed non-Linux partitions on the system	This attribute can be 0 or more.
<u>xserver</u>	X Server needed for graphic adapter	This attribute must always match exactly.
<u>memsize</u>	Memory available on host in megabytes	All match types are available.
<u>totaldisk</u>	Total disk space available on host in megabytes	All match types are available.
<u>hostid</u>	Hex representation of the IP address	Exact match required
<u>arch</u>	Architecture of host	Exact match required
<u>karch</u>	Kernel Architecture of host (for example SMP kernel, Xen kernel)	Exact match required
<u>disksize</u>	Drive device and size	All match types are available.
<u>product</u>	The hardware product name as specified in SMBIOS	Exact match required
<u>product_vendor</u>	The hardware vendor as specified in SMBIOS	Exact match required
<u>board</u>	The system board name as specified in SMBIOS	Exact match required
<u>board_vendor</u>	The system board vendor as specified in SMBIOS	Exact match required

Attribute	Values	Description
<u>custom1-5</u>	Custom rules using shell scripts	All match types are available.

5.1.7 Rules with Dialogs

You can use dialog pop-ups with check boxes to select rules you want matched.

The elements listed below must be placed within the following XML structure in the rules.xml file:

```
<rules config:type="list">
  <rule>
    <dialog>
      ...
    </dialog>
  </rule>
</rules>
```

Attribute	Values	Description
<u>dialog_nr</u>	All rules with the same <u>dialog_nr</u> are presented in the same pop-up dialog. The same <u>dialog_nr</u> can appear in multiple rules. <pre><dialog_nr config:type="integer">3</ dialog_nr></pre>	This element is optional and the default for a missing <u>dialog_nr</u> is always <u>0</u> . To use one pop-up for all rules, you do not need to specify the <u>dialog_nr</u> .
<u>element</u>	Specify a unique ID. Even if you have more than one dialog, you must not use the same id twice. Using id <u>1</u> on dialog 1 and id <u>1</u> on dialog 2 is not supported. (This behavior is contrary to the <u>ask</u>	Optional. If left out, AutoY-aST adds its own ids internally. Then you cannot specify conflicting rules (see below).

Attribute	Values	Description
	<p>dialog, where you can have the same ID for multiple dialogs.)</p> <pre data-bbox="600 389 991 521"><element config:type="integer">3</ element></pre>	
<u>title</u>	<p>Caption of the pop-up dialog</p> <pre data-bbox="600 629 991 723"><title>Desktop Selection</ title></pre>	Optional
<u>question</u>	<p>Question shown in the pop-up behind the check box.</p> <pre data-bbox="600 875 991 969"><question>GNOME Desktop</ question></pre>	Optional. If you do not configure a text here, the name of the XML file that is triggered by this rule will be shown instead.
<u>timeout</u>	<p>Timeout in seconds after which the dialog will automatically “press” the okay button. Useful for a non-blocking installation in combination with rules dialogs.</p> <pre data-bbox="600 1335 991 1464"><timeout config:type="integer">30</ timeout></pre>	Optional. A missing timeout will stop the installation process until the dialog is confirmed by the user.
<u>conflicts</u>	<p>A list of element ids (rules) that conflict with this rule. If this rule matches or is selected by the user, all conflicting rules are deselected and</p>	<u>optional</u>

Attribute	Values	Description
	<p>disabled in the pop-up. Take care that you do not create deadlocks.</p> <pre><conflicts config:type="list"> <element config:type="integer">1</ element> <element config:type="integer">5</ element> ... </conflicts></pre>	

Here is an example of how to use dialogs with rules:

```
<rules config:type="list">
  <rule>
    <custom1>
      <script>
echo -n 100
      </script>
      <match>100</match>
      <match_type>exact</match_type>
    </custom1>
    <result>
      <profile>rules/gnome.xml</profile>
      <continue config:type="boolean">>true</continue>
    </result>
    <dialog>
      <element config:type="integer">0</element>
      <question>GNOME Desktop</question>
      <title>Desktop Selection</title>
      <conflicts config:type="list">
        <element config:type="integer">1</element>
      </conflicts>
      <dialog_nr config:type="integer">0</dialog_nr>
    </dialog>
  </rule>
  <rule>
    <custom1>
      <script>
echo -n 100
```

```

    </script>
    <match>101</match>
    <match_type>exact</match_type>
</custom1>
<result>
  <profile>rules/gnome.xml</profile>
  <continue config:type="boolean">>true</continue>
</result>
<dialog>
  <element config:type="integer">1</element>
  <dialog_nr config:type="integer">0</dialog_nr>
  <question>Gnome Desktop</question>
  <conflicts config:type="list">
    <element config:type="integer">0</element>
  </conflicts>
</dialog>
</rule>
<rule>
  <custom1>
    <script>
echo -n 100
    </script>
    <match>100</match>
    <match_type>exact</match_type>
  </custom1>
  <result>
    <profile>rules/all_the_rest.xml</profile>
    <continue config:type="boolean">>false</continue>
  </result>
</rule>
</rules>

```

5.2 Classes

Classes represent configurations for groups of target systems. Unlike rules, classes need to be configured in the control file. Then classes can be assigned to target systems.

Here is an example of a class definition:

```

<classes config:type="list">
  <class>
    <class_name>TrainingRoom</class_name>
    <configuration>Software.xml</configuration>
  </class>
</classes>

```

In the example above, the file `Software.xml` must be placed in the subdirectory `classes/TrainingRoom/`. It will be fetched from the same place the AutoYaST control file and rules were fetched from.

If you have multiple control files and those control files share parts, better use classes for common parts. You can also use XIncludes.

Using the configuration management system, you can define a set of classes. A class definition consists of the following variables:

- Name: class name
- Description:
- Order: order (or priority) of the class in the stack of migration

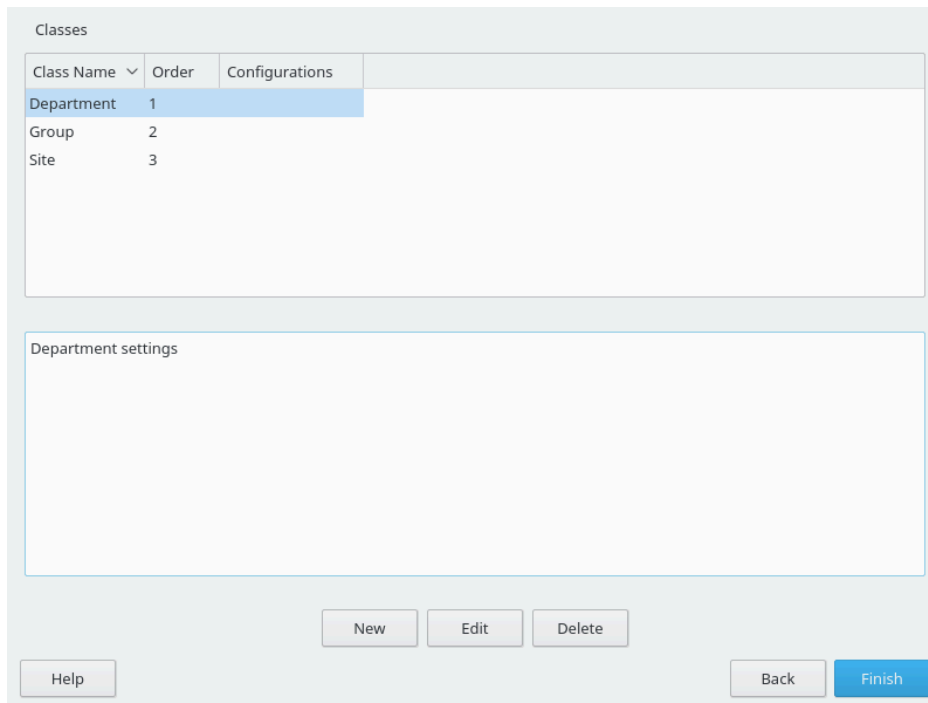


FIGURE 5.2: DEFINING CLASSES

You can create as many classes as you need, however it is recommended to keep the set of classes as small as possible to keep the configuration system concise. For example, the following sets of classes can be used:

- site: classes describing a physical location or site,
- machine: classes describing a type of machine,

- role: classes describing the function of the machine,
- group: classes describing a department or a group within a site or a location.

A file saved in a class directory can have the same syntax and format as a regular control file but represents a subset of the configuration. For example, to create a new control file for a computer with a specific network interface, you only need the control file resource that controls the configuration of the network. Having multiple network types, you can merge the one needed for a special type of hardware with other class files and create a new control file which suits the system being installed.

5.3 Mixing Rules and Classes

It is possible to mix rules and classes during an auto-installation session. For example you can identify a system using rules which contain class definitions in them. The process is described in the figure *Figure A.1, "Rules Retrieval Process"*.

After retrieving the rules and merging them, the generated control file is parsed and checked for class definitions. If classes are defined, then the class files are retrieved from the original repository and a new merge process is initiated.

5.4 Merging of Rules and Classes

With classes and with rules, multiple XML files get merged into one resulting XML file. This merging process is often confusing for people, because it behaves different than one would expect. First of all, it is important to note that the names of the top sections in the merged XML files must be in alphabetical order for the merge to succeed.

For example, the following two XML parts should be merged:

```
<partitioning config:type="list">
  <drive>
    <partitions config:type="list">
      <partition>
        <filesystem config:type="symbol">swap</filesystem>
        <format config:type="boolean">>true</format>
        <mount>swap</mount>
        <partition_id config:type="integer">130</partition_id>
        <size>2000mb</size>
      </partition>
    </partitions>
  </drive>
</partitioning>
```



```

    </partition>
  </partition>
  <filesystem config:type="symbol">xf</filesystem>
  <partition_type>primary</partition_type>
  <size>4Gb</size>
  <mount>/data</mount>
</partition>
</partitions>
</drive>
</partitioning>

```

```

<partitioning config:type="list">
  <drive>
    <initialize config:type="boolean">>false</initialize>
    <partitions config:type="list">
      <partition>
        <format config:type="boolean">>true</format>
        <filesystem config:type="symbol">xf</filesystem>
        <mount>/</mount>
        <partition_id config:type="integer">131</partition_id>
        <partition_type>primary</partition_type>
        <size>max</size>
      </partition>
    </partitions>
    <use>all</use>
  </drive>
</partitioning>

```

You might expect the control file to contain three partitions. This is not the case. You will end up with two partitions and the first partition is a mix up of the swap and the root partition. Settings configured in both partitions, like `mount` or `size`, will be used from the second file. Settings that only exist in the first or second partition, will be copied to the merged partition too.

In this example, you do not want a second `drive`. The two drives should be merged into one. With regard to partitions, three separate ones should be defined. Using the `dont_merge` method solves the merging problem:

```

<classes config:type="list">
  <class>
    <class_name>swap</class_name>
    <configuration>largeswap.xml</configuration>
    <dont_merge config:type="list">
      <element>partition</element>
    </dont_merge>
  </class>
</classes>

```

```
<rule>
  <board_vendor>
    <match>ntel</match>
    <match_type>regex</match_type>
  </board_vendor>
  <result>
    <profile>classes/largeswap.xml</profile>
    <continue config:type="boolean">true</continue>
    <dont_merge config:type="list">
      <element>partition</element>
    </dont_merge>
  </result>
  <board_vendor>
    <match>PowerEdge [12]850</match>
    <match_type>regex</match_type>
  </board_vendor>
  <result>
    <profile>classes/smallswap.xml</profile>
    <continue config:type="boolean">true</continue>
    <dont_merge config:type="list">
      <element>partition</element>
    </dont_merge>
  </result>
</rule>
```

IV Understanding the Auto-Installation Process

6 The Auto-Installation Process **213**

6 The Auto-Installation Process

6.1 Introduction

After the system has booted into an automatic installation and the control file has been retrieved, YaST configures the system according to the information provided in the control file. All configuration settings are summarized in a window that is shown by default and should be deactivated if a fully automatic installation is needed.

By the time YaST displays the summary of the configuration, YaST has only probed hardware and prepared the system for auto-installation. Nothing has been changed in the system yet. In case of any error, you can still abort the process.

A system should be automatically installable without the need to have any graphic adapter or monitor. Having a monitor attached to the client machine is nevertheless recommended so you can supervise the process and to get feedback in case of errors. Choose between the graphical and the text-based Ncurses interfaces. For headless clients, system messages can be monitored using the serial console.

6.1.1 X11 Interface (graphical)

This is the default interface while auto-installing. No special variables are required to activate it.

6.1.2 Serial Console

Start installing a system using the serial console by adding the keyword `console` (for example `console=ttyS0`) to the command line of the kernel. This starts `linuxrc` in console mode and later YaST in serial console mode.

6.1.3 Text-based YaST Installation

This option can also be activated on the command line. To start YaST in text mode, add `textmode=1` on the command line.

Starting YaST in the text mode is recommended when installing a client with less than 64 MB or when X11 should not be configured, especially on headless machines.

6.2 Choosing the Right Boot Medium

There are different methods for booting the client. The computer can boot from its network interface card (NIC) to receive the boot images via DHCP or TFTP. Alternatively a suitable kernel and initrd image can be loaded from a flash disk (for example, USB stick) or a bootable DVD-ROM.

YaST will check for `autoinst.xml` in the root directory of the boot medium or the initrd upon start-up and switch to an automated installation if it was found. In case the control file is named differently or located elsewhere, specify its location on the kernel command line with the parameter `AutoYaST=URL`.

6.2.1 Booting from a Flash Disk (for example, USB stick)

For testing/rescue purposes or because the NIC does not have a PROM or PXE you can build a bootable flash disk to use with AutoYaST. Flash disks can also store the control file.



Tip: Creating a Bootable Flash Disk

To create a bootable flash disk, copy either the openSUSE Leap ISO image of DVD1 or the Mini CD ISO image to the disk using the `dd` command (the flash disk must not be mounted, all data on the device will be erased):

```
tux > sudo dd if=PATH_TO_ISO_IMAGE of=USB_STORAGE_DEVICE bs=4M
```

6.2.2 Booting from DVD-ROM

You can use the original openSUSE Leap DVD-ROM number one in combination with other media. For example, the control file can be provided via a flash disk or a specified location on the network. Alternatively, create a customized DVD-ROM that includes the control file.

6.2.3 Booting via PXE over the Network

Booting via PXE requires a DHCP and a TFTP server in your network. The computer will then boot without a physical medium.

If you install via PXE, the installation will run in an endless loop. This happens because after the first reboot, the machine performs the PXE boot again and restarts the installation instead of booting from the hard disk for the second stage of the installation.

There are several ways to solve this problem. You can use an HTTP server to provide the AutoYaST control file. Alternatively, instead of a static control file, run a CGI script on the Web server that provides the control file and changes the TFTP server configuration for your target host. This way, the next PXE boot of the machine will be from the hard disk by default.

Another way is to use AutoYaST to upload a new PXE boot configuration for the target host via the control file:

```
<pxe>
  <pxe_localboot config:type="boolean">true</pxe_localboot>
  <pxelinux-config>
    DEFAULT linux
    LABEL linux
    localboot 0
  </pxelinux-config>
  <tftp-server>192.168.1.115</tftp-server>
  <pxelinux-dir>/pxelinux.cfg</pxelinux-dir>
  <filename>__MAC__</filename>
</pxe>
```

This entry will upload a new configuration for the target host to the TFTP server shortly before the first reboot happens. In most installations the TFTP daemon runs as user `nobody`. You need to make sure this user has write permissions to the `pxelinux.cfg` directory. You can also configure the file name that will be uploaded. If you use the “magic” `__MAC__` file name, the file name will be the MAC address of your machine like, for example `01-08-00-27-79-49-ee`. If the file name setting is missing, the IP address will be used for the file name.

To do another auto-installation on the same machine, you need to remove the file from the TFTP server.

6.3 Invoking the Auto-Installation Process

6.3.1 Command Line Options

Adding the command line variable `autoyast` causes `linuxrc` to start in automated mode. The `linuxrc` program searches for a configuration file, which should be distinguished from the main control file, in the following places:

- in the root directory of the initial RAM disk used for booting the system;
- in the root directory of the boot medium.

The `linuxrc` configuration file supports multiple keywords. For a detailed description of how `linuxrc` works and other keywords, see [Appendix C, Advanced linuxrc Options](#). Some of the more common ones are:

TABLE 6.1: KEYWORDS FOR `linuxrc`

Keyword	Value
<code>autoupgrade</code>	Initiate an automatic upgrade using AutoYaST; see Section 4.9, "Upgrade" .
<code>autoyast</code>	Location of the control file for automatic installation; see AutoYaST Control File Locations for details.
<code>ifcfg</code>	Configure and start the network. Required if the AutoYaST is to be fetched from a remote location. See Section C.3, "Advanced Network Setup" for details.
<code>insmod</code>	Kernel modules to load
<code>install</code>	Location of the installation directory, for example <code>install=nfs://192.168.2.1/CDs/</code> .
<code>instmode</code>	Installation mode, for example <code>nfs</code> , <code>http</code> etc. (not needed if <code>install</code> is set).
<code>rootpassword</code>	Password for root user if not specified in AutoYaST profile
<code>server</code>	Server (NFS) to contact for source directory

Keyword	Value
<u>serverdir</u>	Directory on NFS Server
<u>y2confirm</u>	Even with <code>< confirm > no </confirm ></code> in the control file, the confirm proposal comes up.

These variables and keywords will bring the system up to the point where YaST can take over with the main control file. Currently, the source medium is automatically discovered, which in some cases makes it possible to initiate the auto-install process without giving any instructions to **linuxrc**.

The traditional **linuxrc** configuration file (info) has the function of giving the client enough information about the installation server and the location of the sources. Usually, this file is not required, but it is needed in special network environments where DHCP and BOOTP are not used or when special kernel modules need to be loaded.

You can pass keywords to **linuxrc** using the kernel command line. This can be done in several ways. You can specify **linuxrc** keywords along with other kernel parameters interactively at boot time, in the usual way. You can also insert kernel parameters into custom network-bootable disk images. It is also possible to configure a DHCP server to pass kernel parameters in combination with Etherboot or PXE.



Note: Using autoyast2 Boot Option instead of autoyast

The autoyast2 option is similar to the autoyast option, but **linuxrc** parses the provided value and, for example, tries to configure a network when needed. This option is not described in this documentation. For information about differences between the AutoYaST and **linuxrc** URI syntax, see the **linuxrc** appendix: *Appendix C, Advanced linuxrc Options*. AutoYaST's rules and classes are *not* supported.

The command line variable autoyast can be used in the format described in the following list.

AUTOYAST CONTROL FILE LOCATIONS

Format of URIs

The autoyast syntax for the URIs for your control file locations can be confusing. The format is SCHEMA://HOST/PATH-TO-FILE. The number of forward slashes to use varies. For remote locations of your control file, the URI looks like this example for an NFS server, with two slashes: autoyast=nfs://SERVER/PATH.

It is different when your control file is on a local file system. For example, `autoyast=usb:///profile.xml` is the same as `autoyast=usb://localhost/profile.xml`. You may omit the local host name, but you must keep the third slash. `autoyast=usb://profile.xml` will fail because `profile.xml` is interpreted as the host name.

When no control file specification is needed

For upgrades, no `autoyast` variable is needed for an automated offline upgrade, see *Procedure 4.1, "Starting AutoYaST in Offline Upgrade Mode"*.

For new installations, `autoyast` will be started if a file named `autoinst.xml` is in one of the following three locations:

1. The root directory of the installation flash disk (for example, USB stick)
2. The root directory of the installation medium
3. Or the root directory of the initial RAM disk used to boot the system

autoyast=file:///PATH

Looks for control file in the specified path, relative to the source root directory, for example `file:///autoinst.xml` when the control file is in the top-level directory of any local file system, including mounted external devices such as a CD or USB drive. (This is the same as `file://localhost/autoinst.xml`.)

autoyast=device://DEVICE/FILENAME

Looks for the control file on a storage device. Do not specify the full path to the device, but the device name only (for example, `device://vda1/autoyast.xml`). You may also omit specifying the device and trigger `autoyast` to search all devices, for example. `autoyast=device://localhost/autoinst.xml`, or `autoyast=device:///autoinst.xml`.

autoyast=nfs://SERVER/PATH

Looks for the control file on an NFS server.

autoyast=http://[user:password@]SERVER/PATH

Retrieves the control file from a Web server using the HTTP protocol. Specifying a user name and a password is optional.

autoyast=https://[user:password@]SERVER/PATH

Retrieves the control file from a Web server using HTTPS. Specifying a user name and a password is optional.

autoyast=tftp://SERVER/PATH

Retrieve the control file via TFTP.

autoyast=ftp://[user:password@]SERVER/PATH

Retrieve the control file via FTP. Specifying a user name and a password is optional.

autoyast=usb:///PATH

Retrieve the control file from USB devices (autoyast will search all connected USB devices).

autoyast=relurl://PATH

Retrieve the control file from the installation source. Either from the default installation source or from the installation source defined in install=INSTALLATION_SOURCE_PATH.

autoyast=cifs://SERVER/PATH

Looks for the control file on a CIFS server.

autoyast=label://LABEL/PATH

Searches for a control file on a device with the specified label.

Several scenarios for auto-installation are possible using different types of infrastructure and source media. The simplest way is to use the source media (DVD number one) of openSUSE Leap. But to initiate the auto-installation process, the auto-installation command line variable should be entered at system boot-up and the control file should be accessible for YaST.

In a scripting context, you can use a serial console for your virtual machine, that allows you to work in text mode. Then you can pass the needed parameters from an expect script or equivalent.

The following list of scenarios explains how the control file can be supplied:

Using the Original openSUSE Leap DVD-ROM

When using the original DVD-ROM (DVD #1 is needed), the control file needs to be accessible via flash disk (for example, USB stick) or network:

Flash Disk (for example, USB stick). Access the control file via the autoyast=usb:///PATH option.

Network. Access the control file via the following commands: autoyast=nfs://.., autoyast=ftp://.., autoyast=http://.., autoyast=https://.., autoyast=tftp://.., or autoyast=cifs://... Network access needs to be defined using the boot options in `linuxrc`. This can be done via DHCP: **netsetup=dhcp autoyast=http://163.122.3.5/autoyast.xml**

Using a Custom DVD-ROM

In this case, you can include the control file directly on the DVD-ROM. When placing it in the root directory and naming it `autoinst.xml`, it will automatically be found and used for the installation. Otherwise use `autoyast=file:///PATH` to specify the path to the control file.

Using a Network Installation Source

This option is the most important one because installations of multiple machines are usually done using SLP or NFS servers and other network services like BOOTP and DHCP. The easiest way to make the control file available is to place it in the root directory of the installation source, naming it `autoinst.xml`. In this case, it will automatically be found and used for the installation. The control file can also reside in the following places:

Flash Disk (for example, USB stick). Access the control file via the `autoyast=usb:///PATH` option.

Network. Access the control file via the following commands: `autoyast=nfs://..`, `autoyast=ftp://..`, `autoyast=http://..`, `autoyast=https://..`, `autoyast=tftp://..`, or `autoyast=cifs://...`



Note: Disabling Network and DHCP

To disable the network during installations where it is not needed or unavailable, for example when auto-installing from DVD-ROMs, use the `linuxrc` option `netsetup=0` to disable the network setup.

With all AutoYaST invocation options it is possible to specify the location of the control file in the following ways:

1. Specify the exact location of the control file:

```
autoyast=http://192.168.1.1/control-files/client01.xml
```

2. Specify a directory where several control files are located:

```
autoyast=http://192.168.1.1/control-files/
```

In this case the relevant control file is retrieved using the hex digit representation of the IP as described below.

The path of this directory needs to end with a `/`.

The files in the directory must not have any extension, for example `.xml`. So the file name needs to be the IP or MAC address only.

```
tux > ls -r control-files
C00002 0080C8F6484C default
```

If only the path prefix variable is defined, YaST will fetch the control file from the specified location in the following way:

1. First, it will search for the control file using its own IP address in uppercase hexadecimal, for example `192.0.2.91 -> C000025B`.
2. If this file is not found, YaST will remove one hex digit and try again. This action is repeated until the file with the correct name is found. Ultimately, it will try looking for a file with the MAC address of the client as the file name (mac should have the following syntax: `0080C8F6484C`) and if not found a file named `default` (in lowercase).

As an example, for 192.0.2.91, the HTTP client will try:

```
C000025B
C000025
C00002
C0000
C000
C000
C00
C0
C
0080C8F6484C
default
```

in that order.

To determine the hex representation of the IP address of the client, use the utility called `/usr/bin/gethostip` available with the `syslinux` package.

EXAMPLE 6.1: DETERMINE HEX CODE FOR AN IP ADDRESS

```
tux > /usr/bin/gethostip 10.10.0.1
10.10.0.1 10.10.0.1 0A0A0001
```

6.3.2 Auto-installing a Single System

The easiest way to auto-install a system without any network connection is to use the original openSUSE Leap DVD-ROMs and a flash disk (for example, USB stick). You do not need to set up an installation server nor the network environment.

Create the control file and name it `autoinst.xml`. Copy the file `autoinst.xml` to the flash disk.

6.3.3 Combining the `linuxrc info` File with the AutoYaST Control File

If you choose to pass information to `linuxrc` using the `info` file or as boot options, you may integrate the keywords into the AutoYaST control file. Add an `info_file` section as shown in the example below. This section contains keyword—value pairs, separated by colons, one pair per line.

EXAMPLE 6.2: `linuxrc` OPTIONS IN THE AUTOYAST CONTROL FILE

```
....
<install>
....
  <init>
    <info_file>

install: nfs://192.168.1.1/CDs/full-x86_64
dud: https://example.com/driver_updates/filename.dud
upgrade: 1
textmode: 1
  </info_file>
  </init>
.....
</install>
....
```

Note that the `autoyast2` keyword must point to the same file. If it is on a flash disk (for example, USB stick), then the option `usb://` needs to be used. If the `info` file is stored in the initial RAM disk, the `file:///` option needs to be used.

6.4 System Configuration

The system configuration during auto-installation is the most important part of the whole process. As you have seen in the previous chapters, almost anything can be configured automatically on the target system. In addition to the pre-defined directives, you can always use post-scripts to change other things in the system. Additionally you can change any system variables, and if required, copy complete configuration files into the target system.

6.4.1 Post-Install and System Configuration

The post-installation and system configuration are initiated directly after the last package is installed on the target system and continue after the system has booted for the first time.

Before the system is booted for the first time, AutoYaST writes all data collected during installation and writes the boot loader in the specified location. In addition to these regular tasks, AutoYaST executes the chroot-scripts as specified in the control file. Note that these scripts are executed while the system is not yet mounted.

If a different kernel than the default is installed, a hard reboot will be required. A hard reboot can also be forced during auto-installation, independent of the installed kernel. Use the `reboot` property of the `general` resource (see [Section 4.1, "General Options"](#)).

6.4.2 System Customization

Most of the system customization is done in the second stage of the installation. If you require customization that cannot be done using AutoYaST resources, use post-install scripts for further modifications.

You can define an unlimited number of custom scripts in the control file, either by editing the control file or by using the configuration system.

V Uses for AutoYaST on Installed Systems

- 7 Running AutoYaST in an Installed System [225](#)

7 Running AutoYaST in an Installed System

In some cases it is useful to run AutoYaST in a running system.

In the following example, an additional software package (`foo`) is going to be installed. To run this software, a user needs to be added and an NTP client needs to be configured.

The respective AutoYaST profile needs to include a section for the package installation ([Section 4.8.7, "Installing Packages in Stage 2"](#)), a user ([Section 4.28.1, "Users"](#)) section and an NTP-client ([Section 4.19, "NTP Client"](#)) section:

```
<?xml version="1.0"?>
<!DOCTYPE profile>
<profile xmlns="http://www.suse.com/1.0/yast2ns" xmlns:config="http://www.suse.com/1.0/
configns">
  <ntp-client>
    <peers config:type="list">
      <peer>
        <address>us.pool.ntp.org</address>
        <comment/>
        <options> iburst</options>
        <type>server</type>
      </peer>
    </peers>
    <start_at_boot config:type="boolean">true</start_at_boot>
    <start_in_chroot config:type="boolean">>false</start_in_chroot>
    <sync_interval config:type="integer">5</sync_interval>
    <synchronize_time config:type="boolean">>false</synchronize_time>
  </ntp-client>
  <software>
    <post-packages config:type="list">
      <package>ntp</package>
      <package>yast2-ntp-client</package>
      <package>foo</package>
    </post-packages>
  </software>
  <users config:type="list">
    <user>
      <encrypted config:type="boolean">>false</encrypted>
      <fullname>Foo user</fullname>
      <gid>100</gid>
      <home>/home/foo</home>
      <password_settings>
        <expire/>
        <flag/>
        <inact/>
      </password_settings>
    </user>
  </users>
</profile>
```



```
<max>99999</max>
<min>0</min>
<warn>7</warn>
</password_settings>
<shell>/bin/bash</shell>
<uid>1001</uid>
<user_password>linux</user_password>
<username>foo</username>
</user>
</users>
</profile>
```

Store this file as /tmp/install_foo.xml and start the AutoYaST installation process by calling:

```
tux > sudo yast2 ayast_setup setup filename=/tmp/install_foo.xml dopackages="yes"
```

For more information, run yast2 ayast_setup longhelp

VI Appendices

- A Handling Rules [228](#)
- B AutoYaST FAQ - Frequently Asked Questions [229](#)
- C Advanced **linuxrc** Options [233](#)
- D Main Differences Between openSUSE Leap 42.3 and 15 Profiles [238](#)

A Handling Rules

The following figure illustrates how rules are handled and the processes of retrieval and merge.

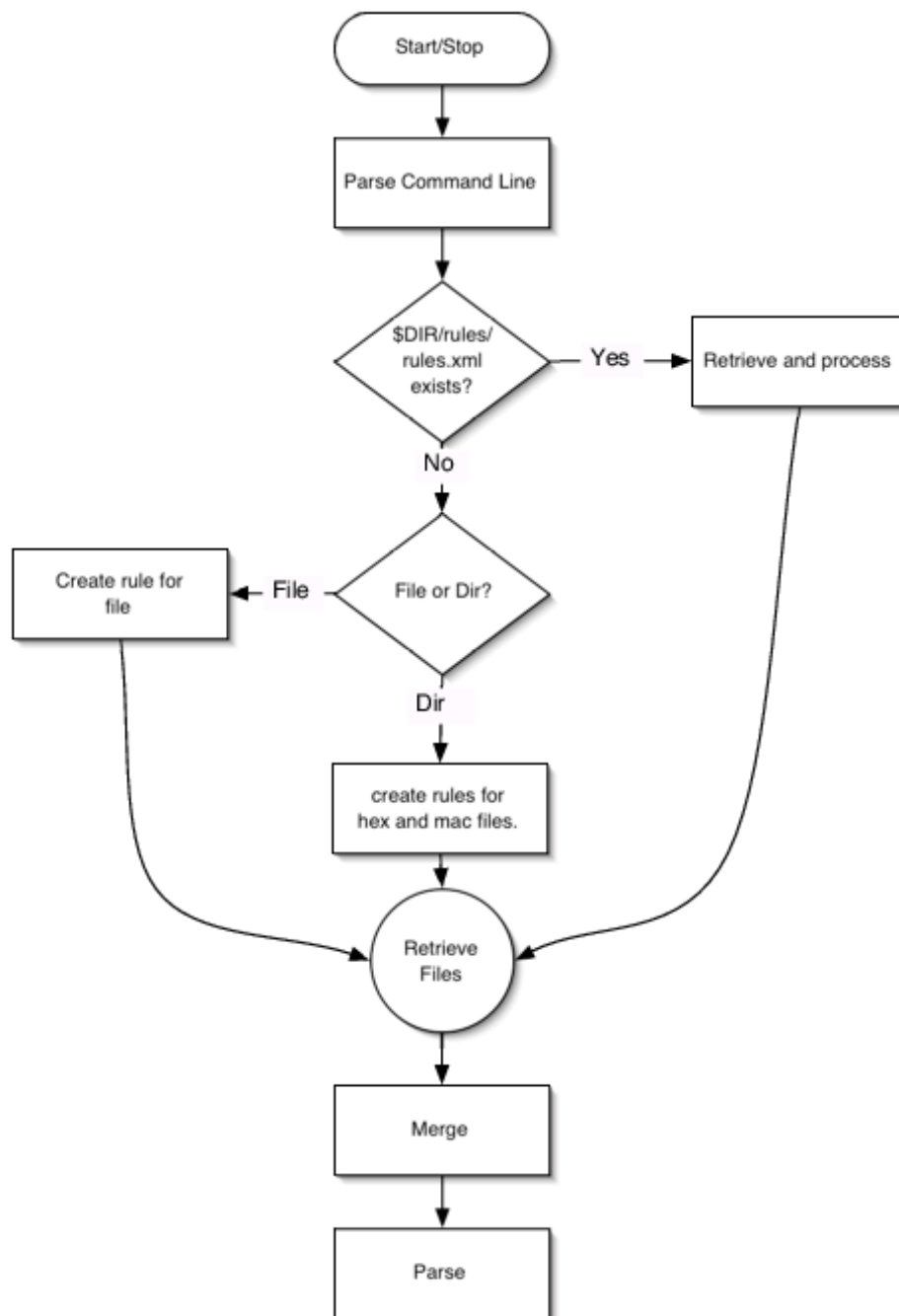


FIGURE A.1: RULES RETRIEVAL PROCESS

B AutoYaST FAQ - Frequently Asked Questions

Q: 1. How do I invoke an AutoYaST installation?

On all openSUSE Leap versions, the automatic installation gets invoked by adding `autoyast=<PATH_TO_PROFILE>` to the kernel parameter list. So for example adding `autoyast=http://MYSERVER/MYCONFIG.xml` will start an automatic installation where the profile with the AutoYaST configuration gets fetched from the Web server `myserver`. See [Section 6.3, "Invoking the Auto-Installation Process"](#) for more information.

Q: 2. What is an AutoYaST profile?

A profile is the AutoYaST configuration file. The content of the AutoYaST profile determines how the system will be configured and which packages will get installed. This includes partitioning, network setup, and software sources, to name but a few. Almost everything that can be configured with YaST in a running system can also be configured in an AutoYaST profile. The profile format is an ASCII XML file.

Q: 3. How do I create an AutoYaST profile?

The easiest way to create an AutoYaST profile is to use an existing openSUSE Leap system as a template. On an already installed system, start `YaST > Miscellaneous > Autoinstallation`. Now select `Tools > Create Reference Profile` from the menu. Choose the system components you want to include in the profile. Alternatively, create a profile containing the complete system configuration by running `sudo yast clone_system` from the command line.

Both methods will create the file `/root/autoinst.xml`. The version created on the command line can be used to set up an identical clone of the system on which the profile was created. However, usually you will want to adjust the file to make it possible to install several machines that are very similar, but not identical. This can be done by adjusting the profile using your favorite text/XML editor.

Q: 4. How can I check the syntax of a created AutoYaST profile?

The most efficient way to check your created AutoYaST profile is by using `jing` or `xmllint`.

See [Section 3.3, "Creating/Editing a Control File Manually"](#) for details.

Q: 5. What is smallest AutoYaST profile that makes sense?

If a section has not been defined in the AutoYaST profile the settings of the general YaST installation proposal will be used. However, you need to specify at least the `root` password to be able to log in to the machine after the installation.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE profile>
<profile xmlns="http://www.suse.com/1.0/yast2ns" xmlns:config="http://
www.suse.com/1.0/configs">
  <users config:type="list">
    <user>
      <encrypted config:type="boolean">>false</encrypted>
      <user_password>linux</user_password>
      <username>root</username>
    </user>
  </users>
</profile>
```

Q: 6. *How do I do an automatic installation with autodetection of my sound card?*

Use the following `sound` section in your profile:

```
<sound>
  <autoinstall config:type="boolean">>true</autoinstall>
  <configure_detected config:type="boolean">>true</configure_detected>
</sound>
```

Q: 7. *I want to install from DVD only. Where do I put the AutoYaST profile?*

Put the profile in the root of the DVD. Refer to it with `file:///PROFILE.xml`.

Q: 8. *How can I test a merging process on the command line?*

To merge two profiles, `a.xml` with `base.xml`, run the following command:

```
tux > /usr/bin/xsltproc --novalid --param replace "'false'" \
--param dontmerge1 "'package'" --param with "'a.xml'" --output out.xml \
/usr/share/autoinstall/xslt/merge.xslt base.xml
```

This requires sections in both profiles to be in alphabetical order (`<software>`, for example, needs to be listed after `<add-on>`). If you have created the profile with YaST, profiles are automatically sorted correctly.

The `dontmerge1` parameter is optional and an example of what to do when you use the `dont_merge` element in your profile. See [Section 5.4, "Merging of Rules and Classes"](#) for more information.

Q: 9. *Can I call Zypper from scripts?*

Zypper can only be called from AutoYaST init scripts, because during the post-script phase, YaST still has an exclusive lock on the RPM database.

If you really need to use other script types (for example a post-script) you will need to break the lock at your own risk:

```
<post-scripts config:type="list">
  <script>
    <filename>yast_clone.sh</filename>
    <interpreter>shell</interpreter>
    <location/>
    <feedback config:type="boolean">>false</feedback>
    <source><![CDATA[#!/bin/sh
mv /var/run/zypp.pid /var/run/zypp.sav
zypper in foo
mv /var/run/zypp.sav /var/run/zypp.pid
]]></source>
  </script>
</post-scripts>
```

Q: 10. *Is the order of sections in an AutoYaST profile important?*

Actually the order is not important. The order of sections in the profile has no influence on the AutoYaST workflow. However, to *merge* different profiles, sections need to be in alphabetical order.

Q: 11. linuxrc blocks the installation with File not signed. I need to manually interact.

linuxrc found an unsigned file, such as a driver update. To use an unsigned file, you can suppress that message by passing insecure=1 to the linuxrc parameter list (together with the autoyast=... parameter).

Q: 12. *I want to install from DVD/USB/HD but fetch the XML file from the network.*

You need to pass ifcfg to linuxrc. This is required to set up the network, otherwise AutoYaST cannot download the profile from the remote host. See [Section C.3, "Advanced Network Setup"](#) for more information.


Q: 13. *Is installation onto an NFS root (/) possible?*

Yes, but it is more complex than other methods. The environment (DHCP, TFTP, etc.) must be set up very carefully. The AutoYaST profile must look like the following:

```
<?xml version="1.0"?>
<!DOCTYPE profile>
<profile xmlns="http://www.suse.com/1.0/yast2ns" xmlns:config="http://
www.suse.com/1.0/configs">
  <partitioning config:type="list">
    <drive>
```

```
<device>/dev/nfs</device>
<initialize config:type="boolean">>false</initialize>
<type config:type="symbol">CT_NFS</type>
<partitions config:type="list">
  <partition>
    <filesystem config:type="symbol">nfs</filesystem>
    <fstopt>nolock</fstopt>
    <device>10.10.1.53:/tmp/m4</device>
    <mount>/</mount>
  </partition>
</partitions>
<use>all</use>
</drive>
</partitioning>
</profile>
```

Q: 14. *Where can I ask questions which have not been answered here?*

There is an AutoYaST mailing list where you can post your questions. Join us at <http://lists.opensuse.org/opensuse-autoinstall/> .

C Advanced **linuxrc** Options

linuxrc is a small program that runs after the kernel has loaded, but before AutoYaST or other stages. It prepares the system for installation. It allows the user to load modules, start an installed system or a rescue system, and to guide the operation of YaST.



Note: AutoYaST and **linuxrc** Settings Are Not Identical

Some **linuxrc** settings coincidentally have the same names as settings used by AutoYaST in its `autoyast.xml` file. This does *not* mean that they take the same parameters or function in the same way. For example, AutoYaST takes a `self_update` setting. If this value is set to `1`, another setting, `self_update_url` will be read and followed. Although **linuxrc** also has a `self_update` setting, **linuxrc**'s setting takes values of either `0` or a URL.

Do not pass AutoYaST parameters to **linuxrc**, as this will almost certainly not give the desired results.

If **linuxrc** is installed on a machine, information about it can be found in the directory `/usr/share/doc/packages/linuxrc/`. Alternatively, its documentation can be found online at: <https://en.opensuse.org/SDB:Linuxrc>.



Note: Running **linuxrc** on an Installed System

If you run **linuxrc** on an installed system, it will work slightly differently so as not to destroy your installation. As a consequence, you cannot test all features this way.

To keep the **linuxrc** binary file as small as possible, all its libraries and other supplemental files are linked directly into the main program binary file. This means that there is no need for any shared libraries in the initial RAM disk, `initrd`.

C.1 Passing Parameters to **linuxrc**

Unless **linuxrc** is in manual mode, it will look for an `info` file in these locations: first `/info` on the flash disk (for example, USB stick) and if that does not exist, for `/info` in the `initrd`. After that, it parses the kernel command line for parameters. You may change the `info` file

`linuxrc` reads by setting the `info` command line parameter. If you do not want `linuxrc` to read the kernel command line (for example, because you need to specify a kernel parameter that `linuxrc` recognizes as well), use `linuxrc=nocmdline`.

`linuxrc` will always look for and parse a file called `/linuxrc.config`. Use this file to change default values if you need to. In general, it is better to use the `info` file instead. Note that `/linuxrc.config` is read before any `info` file, even in manual mode.

C.2 info File Format

Lines starting with `#` are comments. Valid entries are of the form:

```
key: value
```

Note that `value` extends to the end of the line and therefore may contain spaces. The matching of `key` is on a case-insensitive basis.

You can use the same key-value pairs on the kernel command line using the syntax `key=value`. Lines that do not have the form described above will be ignored.

The table below lists important keys and example values. For a complete list of `linuxrc` parameters, refer to <https://en.opensuse.org/SDB:Linuxrc>.

TABLE C.1: ADVANCED `linuxrc` KEYWORDS

Keyword: Example Value	Description
<code>addswap: 0 3 /dev/sda5</code>	If 0, never ask for swap; if the argument is a positive number <code>n</code> , activate the swap partition; if the argument is a partition name, activate this swap partition.
<code>autoyast: ftp://AU-TOYASTFILE</code>	Location of the auto installation file; activates auto installation mode. See <i>AutoYaST Control File Locations</i> for details.
<code>bootptimeout: 10</code>	10 seconds timeout for BOOTP requests.
<code>bootpwait: 5</code>	Sleep 5 seconds between network activation and starting bootp.
<code>display: color mono alt</code>	Set the menu color scheme.

Keyword: Example Value	Description
<u>exec: COMMAND</u>	Run <i>command</i> .
<u>forceinsmod: 0 1</u>	Use the <u>-f</u> option (force) when running <u>insmod</u> commands.
<u>forcerootime: 0 1</u>	Load the installation system into RAM disk.
<u>ifcfg: NETWORK_CONFIGURATION</u>	Set up and start the network. See <i>Section C.3, "Advanced Network Setup"</i> for more information.
<u>insmod: MODULE</u>	Load <u>MODULE</u> .
<u>install: URL</u>	Install from the repository specified with <u>URL</u> . For the syntax of <u>URL</u> refer to https://en.opensuse.org/SDB:Linuxrc#url_descr .
<u>keytable: de-lat1-nd</u>	Virtual console keyboard map to load.
<u>language: de_DE</u>	Language preselected for the installation.
<u>loghost: 10.10.0.22</u>	Enable remote logging via syslog.
<u>memloadimage: 50000</u>	Load installation system into RAM disk if free memory is above 50000 KB.
<u>memlimit: 10000</u>	Ask for swap if free memory drops below 10000 KB.
<u>memYaST: 20000</u>	Run YaST in text mode if free memory is below 20000 KB.
<u>memYaSTText: 10000</u>	Ask for swap before starting YaST if free memory is below 10000 KB.
<u>proxy: 10.10.0.1</u>	Proxy (either FTP or HTTP).
<u>rescue: 1 nfs://server/dir</u>	Load the rescue system; the URL variant specifies the location of the rescue image explicitly.
<u>rescueimage: /suse/images/rescue</u>	Location of the rescue system image.

Keyword: Example Value	Description
<u>rootimage</u> : /suse/images/root	Location of the installation system image.
<u>textmode</u> : 1	Start YaST in text mode.
<u>usbwait</u> : 4	Wait four seconds after loading the USB modules.
<u>y2confirm</u>	Overrides the confirm parameter in a control file and requests confirmation of installation proposal.

C.3 Advanced Network Setup

Even if parameters like hostip, nameserver, and gateway are passed to **linuxrc**, the network is only started when it is needed (for example, when installing via SSH or VNC). Because autoyast is not a **linuxrc** parameter (this parameter is ignored by **linuxrc** and is only passed to YaST), the network will *not* be started automatically when specifying a remote location for the AutoYaST profile.

Therefore, the network needs to be started explicitly. This used to be done with the **linuxrc** parameter netsetup. Starting with openSUSE Leap 13.2, the parameter ifcfg is available. It offers more configuration options, for example configuring more than one interface. ifcfg directly controls the content of the /etc/sysconfig/network/ifcfg-* files.

DHCP Network Configuration

The general syntax to configure DHCP is

```
ifcfg=INTERFACE=DHCP*,OPTION1=VALUE1,OPTION2=VALUE2
```

where INTERFACE is the interface name, for example eth0, or eth* for all interfaces. DHCP* can either be dhcp (IPv4 and IPv6), dhcp4, or dhcp6.

To set up DHCP for eth0 use:

```
ifcfg=eth0=dhcp
```

To set up DHCP on all interfaces use:

```
ifcfg=eth*=dhcp
```

Static Network Configuration

The general syntax to configure a static network is

```
ifcfg=INTERFACE=IP_LIST,GATEWAY_LIST,NAME_SERVER_LIST,DOMAINSEARCH_LIST,\
OPTION1=value1,...
```

where *INTERFACE* is the interface name, for example `eth0`. If using `eth*`, the first device available will be used. The other parameters need to be replaced with the respective values in the given order. Example:

```
ifcfg=eth0=192.168.2.100/24,192.168.5.1,192.168.1.116,example.com
```

When specifying multiple addresses for a parameter, use spaces to separate them and quote the complete string. The following example uses two name servers and a search list containing two domains.

```
ifcfg="eth0=192.168.2.100/24,192.168.5.1,192.168.1.116 192.168.1.117,example.com
example.net"
```

For more information refer to https://en.opensuse.org/SDB:Linuxrc#Network_Configuration.

D Main Differences Between openSUSE Leap 42.3 and 15 Profiles

Significant changes in openSUSE Leap 15, like the new modules concept or replacing SuSEfirewall2 with `firewalld`, required changes in AutoYaST. If you want to reuse existing openSUSE Leap 42.3 profiles with openSUSE Leap 15, you need to adjust them as documented here.

D.1 Partitioning

The partitioning back-end previously used by YaST, `libstorage`, has been replaced by `libstorage-ng` which is designed to allow new capabilities that were not possible before. Despite the back-end change, the XML syntax for profiles has *not* changed. However, openSUSE Leap 15 comes with some general changes, which are explained below.

D.1.1 GPT Becomes the Default Partition Type on AMD64/Intel 64

On AMD64/Intel 64 systems, GPT is now the preferred partition type. However, if you would like to retain the old behavior, you can explicitly indicate this in the profile by setting the `disklabel` element to `msdos`.

D.1.2 Setting Partition Numbers

AutoYaST will no longer support forcing partition numbers, as it might not work in some situations. Moreover, GPT is now the preferred partition table type, so partition numbers are less relevant.

However, the `partition_nr` tag is still available to specify a partition to be reused. Refer to [Section 4.4.3.2, "Partition Configuration"](#) for more information.

D.1.3 Forcing Primary Partitions

It is still possible to force a partition as primary (only on MS-DOS partition tables) by setting the primary_type to primary. However, any other value, like logical, will be ignored by AutoYaST, which will automatically determine the partition type.

D.1.4 Btrfs: Default Subvolume Name

The new storage layer allows the user to set different default subvolumes (or none) for every Btrfs file system. As shown in the example below, a prefix name can be specified for each partition using the subvolumes_prefix tag:

EXAMPLE D.1: SPECIFYING THE BTRFS DEFAULT SUBVOLUME NAME

```
<partition>
  <mount>/</mount>
  <filesystem config:type="symbol">btrfs</filesystem>
  <size>max</size>
  <subvolumes_prefix>@</subvolumes_prefix>
</partition>
```

To omit the subvolume prefix, set the subvolumes_prefix tag:

EXAMPLE D.2: DISABLING BTRFS SUBVOLUMES

```
<partition>
  <mount>/</mount>
  <filesystem config:type="symbol">btrfs</filesystem>
  <size>max</size>
  <subvolumes_prefix>@</subvolumes_prefix>
</partition>
```

As a consequence of the new behavior, the old btrfs_set_default_subvolume_name tag is not needed and, therefore, it is not supported anymore.

D.1.5 Btrfs: Disabling Subvolumes

Btrfs subvolumes can be disabled by setting create_subvolumes to false. To skip the default @ subvolume, specify subvolumes_prefix.

```

<partition>
  <create_subvolumes config:type="boolean">false</create_subvolumes>
  <subvolumes_prefix><![CDATA[]]></subvolumes_prefix>
</partition>]]>

```

D.1.6 Reading an Existing `/etc/fstab` Is No Longer Supported

On openSUSE Leap 15 the ability to read an existing `/etc/fstab` from a previous installation when trying to determine the partitioning layout, is no longer supported.

D.1.7 Setting for Aligning Partitions Has Been Dropped

As cylinders have become obsolete, the `partition_alignment >` tag makes no sense and it is no longer available. AutoYaST will always try to align partitions in an optimal way.

D.2 Firewall Configuration

In openSUSE Leap 15, SuSEfirewall2 has been replaced by `firewalld` as the default firewall. The configuration of these two firewalls differs significantly, and therefore the respective AutoYaST profile syntax has changed.

Old profiles will continue working, but the supported configuration will be very limited. It is recommended to update profiles for Leap 15 as outlined below. If keeping Leap 42.3 profiles, we recommend to check the final configuration to avoid unexpected behavior or network security threats.

TABLE D.1: AUTOYAST FIREWALL CONFIGURATION IN LEAP 15: BACKWARD COMPATIBILITY

Supported (but deprecated)	Unsupported
<code>FW_CONFIGURATIONS_{DMZ, EXT, INT}</code>	<code>FW_ALLOW_FW_BROADCAST_{DMZ, EXT, INT}</code>
<code>FW_DEV_{DMZ, EXT, INT}</code>	<code>FW_IGNORE_FW_BROADCAST_{DMZ, EXT, INT}</code>
<code>FW_LOG_DROP_ALL</code>	<code>FW_IPSECT_TRUST</code>

Supported (but deprecated)	Unsupported
<u>FW_LOG_DROP_CRIT</u>	<u>FW_LOAD_MODULES</u>
<u>FW_MASQUERADE</u>	<u>FW_LOG_ACCEPT_ALL</u>
<u>FW_SERVICES_{DMZ, INT, EXT}_{TCP, UDP, IP}</u>	<u>FW_LOG_ACCEPT_CRIT</u>
	<u>FW_PROTECT_FROM_INT</u>
	<u>FW_ROUTE</u>
	<u>FW_SERVICES_{DMZ, EXT, INT}_RPC</u>
	<u>FW_SERVICES_ACCEPT_RELATED_{DMZ, EXT, INT}</u>

Configuration options from SuSEfirewall2 that are no longer available either have no equivalent mapping in `firewalld` or will be supported in future releases of openSUSE Leap. Some `firewalld` features are not yet supported by YaST and AutoYaST—you can use them with post installation scripts in your AutoYaST profile. See [Section 4.29, “Custom User Scripts”](#) for more information.



Note: Enabling and Starting the Firewall

Enabling and starting the `systemd` service for `firewalld` is done with the same syntax as in Leap 42.3. This is the only part of the firewall configuration syntax in AutoYaST that has not changed:

```
<firewall>
  <enable_firewall>true</enable_firewall>
  <start_firewall>true</start_firewall>
  ...
</firewall>
```


The following examples show how to convert deprecated (but still supported) profiles to the Leap 15 syntax:

D.2.1 Assigning Interfaces to Zones

Both SuSEfirewall2 and `firewalld` are zone-based, but have a different set of predefined rules and a different level of trust for network connections.

TABLE D.2: MAPPING OF SUSEFIREWALL2 AND `firewalld` ZONES

<code>firewalld</code> (Leap 15)	SuSEfirewall2 (Leap 42.3)
dmz	DMZ
external	EXT with <code>FW_MASQUERADE</code> set to <code>yes</code>
public	EXT with <code>FW_MASQUERADE</code> set to <code>no</code>
internal	INT with <code>FW_PROTECT_FROM_INT</code> set to <code>yes</code>
trusted	INT with <code>FW_PROTECT_FROM_INT</code> set to <code>no</code>
block	N/A
drop	N/A
home	N/A
work	N/A

In SuSEfirewall2 the default zone is the external one (EXT) but it also allows the use of the special keyword `any` to assign all the interfaces that are not listed anywhere to a specified zone.

D.2.1.1 Default Configuration

The following two examples show the default configuration that is applied for the interfaces `eth0`, `eth1`, `wlan0` and `wlan1`.

EXAMPLE D.3: ASSIGNING ZONES: DEFAULT CONFIGURATION (DEPRECATED SYNTAX)

```
<firewall>
```

```
<FW_DEV_DMZ>any eth0</FW_DEV_DMZ>
<FW_DEV_EXT>eth1 wlan0</FW_DEV_EXT>
<FW_DEV_INT>wlan1</FW_DEV_INT>
</firewall>
```

EXAMPLE D.4: ASSIGNING ZONES: DEFAULT CONFIGURATION (LEAP 15 SYNTAX)

```
<firewall>
<default_zone>dmz</default_zone>
<zones config:type="list">
  <zone>
    <name>dmz</name>
    <interfaces config:type="list">
      <interface>eth0</interface>
    </interfaces>
  </zone>
  <zone>
    <name>public</name>
    <interfaces config:type="list">
      <interface>eth1</interface>
    </interfaces>
  </zone>
  <zone>
    <name>trusted</name>
    <interfaces config:type="list">
      <interface>wlan1</interface>
    </interfaces>
  </zone>
</zones>
</firewall>
```

D.2.1.2 Masquerading and Protecting Internal Zones

The following two examples show how to configure the interfaces eth0, eth1, wlan0 and wlan1 with masquerading and protected internal zones.

EXAMPLE D.5: MASQUERADING AND PROTECTING INTERNAL ZONES (DEPRECATED SYNTAX)

```
<firewall>
<FW_DEV_DMZ>any eth0</FW_DEV_DMZ>
<FW_DEV_EXT>eth1 wlan0</FW_DEV_EXT>
<FW_DEV_INT>wlan1</FW_DEV_INT>
<FW_MASQUERADE>yes</FW_MASQUERADE>
<FW_PROTECT_FROM_INT>yes</FW_PROTECT_FROM_INT>
</firewall>
```

EXAMPLE D.6: MASQUERADING AND PROTECTING INTERNAL ZONES (LEAP 15 SYNTAX)

```
<firewall>
<default_zone>dmz</default_zone>
<zones config:type="list">
  <zone>
    <name>dmz</name>
    <interfaces config:type="list">
      <interface>eth0</interface>
    </interfaces>
  </zone>
  <zone>
    <name>external</name>
    <interfaces config:type="list">
      <interface>eth1</interface>
    </interfaces>
  </zone>
  <zone>
    <name>internal</name>
    <interfaces config:type="list">
      <interface>wlan1</interface>
    </interfaces>
  </zone>
</zones>
</firewall>
```

D.2.2 Opening Ports

In SuSEfirewall2 the `FW_SERVICES_{DMZ,EXT,INT}_{TCP,UDP,IP,RPC}` tags were used to open ports in different zones.

For `TCP` or `UDP`, SuSEfirewall2 supported a port number or range, or a service name from `/etc/services` with a single tag for the respective zone and service. For IP services a port number or range, or a protocol name from `/etc/protocols` could be specified with `FW_SERVICES_ZONE_IP`.

For `firewalld` each port, port range, and service requires a separate entry in the `port` section for the respective zone. IP services need separate entries in the `protocol` section.

RPC services, which were supported by SuSEfirewall2, are no longer supported with `firewalld`.

EXAMPLE D.7: OPENING PORTS (DEPRECATED SYNTAX)

```
<firewall>
<FW_SERVICES_DMZ_TCP>ftp ssh 80 5900:5999</FW_SERVICES_DMZ_TCP>
```

```

<FW_SERVICES_EXT_UDP>1723 ipsec-nat-t</FW_SERVICES_EXT_UDP>
<FW_SERVICES_EXT_IP>esp icmp gre</FW_SERVICES_EXT_IP>
<FW_MASQUERADE>yes</FW_MASQUERADE>
</firewall>

```

EXAMPLE D.8: OPENING PORTS (LEAP 15 SYNTAX)

```

<firewall>
<zones config:type="list">
  <zone>
    <name>dmz</name>
    <ports config:type="list">
      <port>ftp/tcp</port>
      <port>ssh/tcp</port>
      <port>80/tcp</port>
      <port>5900-5999/tcp</port>
    </ports>
  </zone>
  <zone>
    <name>external</name>
    <ports config:type="list">
      <port>1723/udp</port>
      <port>ipsec-nat-t/udp</port>
    </ports>
    <protocols config:type="list">
      <protocol>esp</protocol>
      <protocol>icmp</protocol>
      <protocol>gre</protocol>
    </protocols>
  </zone>
</zones>
</firewall>

```

D.2.3 Opening firewalld Services

For opening a combination of ports and/or protocols, SuSEfirewall2 provides the `FW_CONFIGURATIONS_{EXT, DMZ, INT}` tags which are equivalent to services in `firewalld`.

EXAMPLE D.9: OPENING SERVICES (DEPRECATED SYNTAX)

```

<firewall>
<FW_CONFIGURATIONS_EXT>dhcp dhcpv6 samba vnc-server</FW_CONFIGURATIONS_EXT>
<FW_CONFIGURATIONS_DMZ>ssh</FW_CONFIGURATIONS_DMZ>
</firewall>

```

EXAMPLE D.10: OPENING SERVICES (LEAP 15 SYNTAX)

```
<firewall>
  <zones config:type="list">
    <zone>
      <name>dmz</name>
      <services config:type="list">
        <service>ssh</service>
      </services>
    </zone>
    <zone>
      <name>public</name>
      <services config:type="list">
        <service>dhcp</service>
        <service>dhcpv6</service>
        <service>samba</service>
        <service>vnc-server</service>
      </services>
    </zone>
  </zones>
</firewall>
```

The services definition can be added via packages in both cases:

- SuSEfirewall2 Service Definitions: https://en.opensuse.org/SuSEfirewall2/Service_Definitions_Added_via_Packages ↗
- `firewalld` RPM Packaging https://en.opensuse.org/firewalld/RPM_Packaging ↗
`firewalld` already provides support for the majority of important services in `/usr/lib/firewalld/services`. Check this directory for an existing configuration before defining a new one.

D.2.4 For More Information

- Official `firewalld` Documentation (<http://www.firewalld.org/documentation/>) ↗

D.3 NTP Configuration

The time server synchronization daemon `ntpd` has been replaced with the more modern daemon `chrony`. Therefore the configuration syntax for the time-keeping daemon in AutoYaST has changed. AutoYaST profiles from Leap 42.3 that contain a section with `ntp:client` need to be updated.

Instead of containing low level configuration options, NTP is now configured by a set of high level options that are applied on top of the default settings:

EXAMPLE D.11: NTP CONFIGURATION (LEAP 15 SYNTAX)

```
<ntp-client>
<ntp_policy>auto</ntp_policy>
<ntp_servers config:type="list">
  <ntp_server>
    <iburst config:type="boolean">>false</iburst>
    <address>cz.pool.ntp.org</address>
    <offline config:type="boolean">>true</offline>
  </ntp_server>
</ntp_servers>
<ntp_sync>systemd</ntp_sync>
</ntp-client>
```

D.4 AutoYaST Packages Are Needed for the Second Stage

A regular installation is performed in a single stage, while an installation performed via AutoYaST usually needs two stages. In order to perform the second stage of the installation AutoYaST requires a few additional packages, for example [autoyast2-installation](#) and [autoyast2](#). If these are missing, a warning will be shown.

D.5 The CA Management Module Has Been Dropped

The module for CA Management (`yast2-ca-management >`) has been removed from openSUSE Leap 15, and for the time being there is no replacement available. In case you are reusing a Leap 42.3 profile, make sure it does not contain a `ca_mgm` section.

D.6 Upgrade

D.6.1 Software

Leap 42.3 has two modes of evaluating which packages need to be upgraded. In openSUSE Leap 15.2, upgrades are always determined by the dependency solver, equivalent to using zypper dup.

This makes the option only_installed_packages in the software section obsolete.