siduction manual

siduction team

September 21, 2024

Contents

1   Welcome

1.1   The siduction GNU-Linux operating system

The name **siduction™** is a play on two words: the word **sid**, meaning the code name of Debian Unstable, and **seduction**.

siduction is an operating system based on the Linux kernel and the GNU project. In addition there are application programs from Debian. siduction is committed to the core values of the Debian Social Contract and the following "*Debian Free Software Guidelines*".
See also DFSG

1.1.1   General

For those who want to get started quickly, here is the Quick Start Guide

The siduction operating system manual is a reference for getting to know the system as well as for refreshing your knowledge of the system. It not only provides basic knowledge, but also covers complex topics and supports the work as an administrator of siduction systems.

It is divided into similar topics: Everything concerning partitioning, for example, is in the "Installation/Partitioning" chapter, and topics concerning WiFi are in the "Network" chapter.

The current manual is delivered with the ISOs. Since changes are made to the manual on an ongoing basis, it is worth taking a look at the online version from time to time.

Printing manual pages:
Linux commands can be more than 120 characters long. In order to optimize the display on a screen, automatic line breaks are not applied. Our manual in PDF format, on the other hand, contains line breaks for the long commands. The PDF manual is available on all ISOs and on the system after installation.
To print manual pages, please use the PDF and print only the pages you need.

To get help for a specific pre-installed or self-installed application program (also called a package), it is best to consult the FAQs, online manuals or forums on the home page, or the application's help menu.

Almost all application programs offer assistance by means of an associated "*Manual-Page*" (short manpage). It can be called in the terminal using the command `man <package_name>`. You can also check for documentation in the `/usr`↙ `/share/doc/<packagename>` directory.

1.1.2 Copyright Legal and License Notices

All rights © 2006-2023 of the siduction manual are licensed under the GNU Free Documentation License.
This permits copying, distribution, and/or modification of the document under the terms of the GNU Free Document License version 1.3 or later (as published by the Free Software Foundation); without invariant sections and without cover texts (front cover texts, back cover texts).

The rights of protected trademarks or copyrights belong to the respective owners, regardless of whether this is noted or not.

Errors excepted (E&OE)

1.1.3 Disclaimer

This is experimental software. Use at your own risk. The siduction project, its developers, and team members cannot be held liable under any circumstances for any damage to hardware or software, loss of data, or any other direct or indirect damage caused by the use of this software.

If you do not agree with these terms, you are not allowed to further use or distribute this software.

Last edited: 2023/08/15

%release notes

1.2 release notes 2023.1.0

"Standing on the Shoulders of Giants"

The siduction team is very proud to present an unscheduled release for a special occasion. Debian GNU/Linux, whose unstable branch some of us have been following for over 20 years, celebrates its thirtieth birthday on 8/16/2023 and we think that is worthy of all honor.

Debian is the second oldest distribution after Slackware, and is solely supported by the people involved, without a company standing in the background or someone at the top deciding where things go. Debian is considered the "universal operating system" because of the many architectures supported to this day, and the stability of their releases is legendary.

1.2.1 Changing history

The story began on August 16, 1993, when Ian Murdock introduced his self-designed system "Debian Linux Release". The Debian Manifesto followed that same year. The first stable version, 1.1, was released in 1996, then compiled by under 100 developers. The release was codenamed "Buzz", a first reference to the characters from the movie franchise Toy Story, from which the codenames are still borrowed today.

So far, another 16 releases have been added, the current of which is codenamed "Bookworm" and was released on June 10, 2023. The motto is still: It will be released when it is ready.

In 1997 already about 400 developers agreed on the Debian social contract as an important document, which has been edited only twice since its existence and which states under clause 1: **Debian will remain 100% free**. At the same time, the "Debian Free Software Guidelines" (DFSG) were adopted as part of this contract.

1.2.2 Self-managed

Debian administers itself in the form of a do-ocracy. Today, the approximately 1,000 developers determine their tasks independently and responsibly. That this is not always easy among a thousand creative people should be clear to everyone. This is reminded, among other things, of upheavals such as the introduction of systemd, which brought Debian to its breaking point. The inclusion of non-free firmware on the distribution's installation media was also hotly debated in 2022. The discussions about this dragged on for years.

To the outside world, Debian often appears to be at odds with one another, as upcoming decisions are often the subject of heated debate. In the end, however, the project seems to emerge stronger from such phases. This reliability may also be a reason why more derivatives are based on Debian than on any other distribution. Currently 122 active distributions use Debian as a base. Among them are such pioneering distributions as Knoppix or Ubuntu. If you add distributions that are no longer active, the number rises to 414. Impressive!

It all started with Knoppix

Our own timeline starts with Knoppix and went from Kanotix to sidux and aptosid to siduction, which we have been shipping for 12 years now. Debian has never failed us, but the decision making process has sometimes been a bit too slow. For example, we introduced systemd before Debian, and we have been shipping firmware as an option for some time to support current hardware on the installation media.

But otherwise siduction is probably 98% Debian Unstable. We thank you for that and wish the Projekt at least 30 more years. Standing on the Shoulders of Giants.

1.2.3 What can you expect from siduction 2023.1.0

User and password for the live session are siducer/live.

First of all, a new wallpaper by artist Angevere (Ona Kristensen) awaits you, which we may use with kind permission.

1.2.4   The Flavours

The flavours we offer for siduction 2023.1.0 are KDE Plasma 5.27.7.1, LXQt 1.3.0, Xfce 4.18, Xorg and noX. GNOME, MATE and Cinnamon did not make it again this time because there is no maintainer within siduction for them. If you are interested, please get in touch. They may or may not come back at some point. Of course they are still installable from the repository.

The released images of siduction 2023.1.0 are a snapshot of Debian Unstable, also named Sid, from Aug. 14, 2023, enriched with some useful packages and scripts, a Calamares-based installer and a customized version of the Linux kernel 6.4-10, while systemd is at 254.1-2.

1.2.5   Non-free and Contrib

The following non-free and contrib packages are installed by default:

Nonfree:

- amd64-microcode - Processor microcode firmware for AMD CPUs

- firmware-amd-graphics - Binary firmware for AMD/ATI graphics chips

- firmware-atheros - Binary firmware for Atheros wireless cards

- firmware-bnx2 - Binary firmware for Broadcom NetXtremeII

- firmware-bnx2x - Binary firmware for Broadcom NetXtreme II 10Gb

- firmware-brcm80211 - Binary firmware for Broadcom 802.11 wireless card

- firmware-crystalhd - Crystal HD Video Decoder (firmware)

- firmware-intelwimax - Binary firmware for Intel WiMAX Connection

- firmware-iwlwifi - Binary firmware for Intel Wireless cards

- firmware-libertas - Binary firmware for Marvell Libertas 8xxx wireless car

- firmware-linux-nonfree - Binary firmware for various drivers in the Linux kernel

- firmware-misc-nonfree - Binary firmware for various drivers in the Linux kernel

- firmware-myricom - Binary firmware for Myri-10G Ethernet adapters

- firmware-netxen - Binary firmware for QLogic Intelligent Ethernet (3000)

- firmware-qlogic - Binary firmware for QLogic HBAs

- firmware-realtek - Binary firmware for Realtek wired/wifi/BT adapters

- firmware-ti-connectivity - Binary firmware for TI Connectivity wireless network

- firmware-zd1211 - binary firmware for the zd1211rw wireless driver

- intel-microcode - Processor microcode firmware for Intel CPUs

Contrib:

- b43-fwcutter - utility for extracting Broadcom 43xx firmware

- firmware-b43-installer - firmware installer for the b43 driver

- firmware-b43legacy-installer - firmware installer for the b43legacy driver

- iucode-tool - Intel processor microcode

Remove Non-Free Content

Currently, the installer does not offer the possibility to deselect packages that do not comply with the DFSG, the Debian Free Software Guidelines. This means that packages such as non-free firmware are installed on the system by default. The vrms command will list these packages for you. You can manually uninstall unwanted packages or remove them all by typing apt purge $(vrms -s) before or after installation. Otherwise, our remove-nonfree script can do this for you later.

1.2.6   Installation notes and known issues

If you want to reuse an existing home partition (or any other data partition), you should do this after installation and not in the Calamares installer. While it will work, it is more robust to add it afterwards.

With some Intel graphics processors on some devices, the system may freeze shortly after booting into Live. To fix this, you must set the kernel parameter `intel_iommu=igfx_off` before rebooting.

1.2.7   Many thanks to all involved

We would like to thank you, all the testers and all the people who have supported us over the years. This release is also thanks to you. We would also like to thank the birthday boy Debian, as this means we stand on the shoulders of giants.

And now have fun!

On behalf of the siduction team:

Ferdinand Thommes

Last edited: 2023/12/15

1.3 Credits

1.3.1   Credits for siduction 2023.1.0

Core Team:

Torsten Wohlfarth (towo)
Hendrik Lehmbruch (hendrikL)
Ferdinand Thommes (devil)
Vincent Vietzke (vinzv)
Axel Konrad (akli)

Past contributors:

Alf Gaida (agaida) (eaten by the cat)
Axel Beu 2021†

Code, ideas and support:

Markus Meyer (coruja)
der_bud
se7en
davydych
tuxnix

Artwork:

The artwork is by Angevere (Ona Kristensen). We use it with kind permission.

1.4   Credits for the manual

Core Team

Torsten Wohlfarth (towo)
Hendrik Lehmbruch (hendrikL)
Ferdinand Thommes (devil)
Vincent Vietzke (vinzv)
Axel Konrad (akli)

Credits for the original manual team

Trevor Walkley (bluewater)

Jose Tadeu Barros (ceti)

Alpha Mohamed Diakite (alphad)

Stefan R. Eissens (eislon)

Roland Engert (RoEn)

Alessio Giustini (alessiog75)

Markus Huber (hubi)

Luis_P

Janusz Martyniak (wiarus_old)

Philippe Masson (LjanA)

Mutsumu Nomura (muchan)

Rasmus Güllich Pørksen (ragupo)

Dawid Staropietka (DaVidoSS)

Bruno Torremans (btorrem)

Robert Ulatowski (quidam77)

Dorin Vatavu (dorin)

Bram Verdoodt (Bram0s)

Petr Vorel (pumrel)

zenren

1.4.1   Many thanks to all involved

We would like to thank you, all the testers and all the people who have supported us over the years. siduction is also thanks to you.

Last edited: 2023/12/15

1.5   siduction help

Quick help can save you a lot of tears and gives you the opportunity to work on the important things in life. This section is organized by areas where the siduction distribution offers help.

1.5.1   The siduction forum

The siduction forum offers the possibility to ask questions and get answers to them. Before creating a new post, use the forum search, as there is a good chance that this or a similar question has been asked before. The forum is available in English and German.

1.5.2   IRC - interactive live support.

The IRC should never be entered as "root" but only as a normal user.
If you are unsure, please announce this immediately in the IRC channel so that help can be given.

Rules of conduct in IRC

- A friendly tone is obligatory because we all do the support on a voluntary basis.
- It is helpful to make a request that is accurate to the best of your knowledge and to search for solutions in the siduction wiki beforehand if possible.
- Please never post a request in IRC and the forum at the same time. At best, we rub our eyes in amazement.

Reach siduction

- Just click on the **"IRC Chat #siduction" icon** on the desktop or use the *kmenu* entry of `konversation`.
  If you prefer another chat client, you need to enter these server details:

  ```
  irc.oftc.net
  port 6697
  ```

- [With this link you can start the IRC immediately in your browser](#): Enter a free nickname and join the channel #siduction-en.

1.5.3 Useful helpers in text mode

Normally, you should use text mode runlevel 3 (`init 3` or `journalctl isolate`↙ `multi-user.target`) if you want to perform a dist-upgrade or if you are forced to beceause of a serious system error.

gpm

is a useful program in text mode. It allows you to use the mouse for copying and pasting in the terminal.

gpm is preconfigured in siduction. In case it is not:

```
$ gpm -t imps2 -m /dev/input/mice
```

After that, you should check if the service is active:

```
$ systemctl status gpm.service
```

If successful, you will also find a line similar to the following in the output:

```
   Active: active (running) since Thu 2020-04-09 12:17:14 CEST↙
      ; 5min ago
```

Now you should be able to use your mouse in text mode (tty).

File manager and text editing

Midnight Commander is an easy to use text mode (tty) file manager and text editor preinstalled in siduction.
Apart from normal keyboard input, the mouse can also be used due to gpm.
`mc` shows the file system, and with `mcedit` you can edit an existing file or create a new one.
This is how to open an existing file (a backup copy is created first):

```
$ cp /etc/apt/sources.list.d/debian.list /etc/apt/sources.↙
   list.d/debian.list_$(date +%F)


then


$ mcedit /etc/apt/sources.list.d/debian.list
```

Now the file can be edited and saved. The changes will take effect immediately.

See the man page for more information:

```
$ man mc
```

1.5.4   siduction IRC support in text mode

Rules of conduct in IRC

The IRC should never be entered as "root" but only as a normal user.
If you are unsure, please announce this immediately in the IRC channel so that
help can be given.

IRC in text mode

The program **irssi** provides an IRC client in text mode or console and is activated
in siduction.
With the key combination **ALT**+**F2** or **F3** etc., you can switch from one terminal/TTY
to another and log in with your user account:

```
$ siductionbox login: <username> <password> (not as root)
```

After that you enter

```
$ siduction-irc
```

to start irssi.

Instructions for using a different client (weechat in the example):
First, make sure that weechat is installed by looking for the weechat entry in the
menu. If this is not available:

```
# apt update
# apt install weechat-curses


  and then start the program


$ weechat-curses
```

Now you can connect to irc.oftc.net on port 6697. After successful connection, the pseudonym (the "nickname") will be changed:

/nick 'Your_new_nick'.

You can enter the siduction channel with the following input:

/join #siduction-en

If you want to change the server, enter a command with the following syntax:

/server server.name

In the bottom menu, you can see numbers if the channels are active. In order to connect to a channel, you can use **ALT**+**1**, **ALT**+**2**, **ALT**+**3**, **ALT**+**4**, and so on.

To exit a channel use

/exit

If a dist-upgrade is performed at the same time, you can switch to the terminal to monitor the upgrade progress as follows:

key combination **ALT**+**F3**
To return to the IRC, you can use the
key combination **ALT**+**F2**.

The following links provide more information:
Documentation page of irssi
Documentation page of WeeChat

1.5.5   Surfing the Internet in text mode

The command line browser **w3m** allows you to surf the internet in a terminal, console, or in text mode.
If neither w3m nor elinks are installed, proceed as follows:

```
# apt update
# apt install w3m
# apt install elinks
```

Now you can use the command line browser w3m. For this purpose, it is useful to switch to another terminal and log in with your user account:

key combination **ALT**+**F2**

```
$ siductionbox login: <username> <password> (not root!)
```

The program call is **w3m URL** or **w3m ?**.
Example: https://siduction.org is called like this (https:// is omitted):

```
$ w3m siduction.org
```

A new URL is called using the key combination **Shift**+**U**.

After that, you will see a line like Goto URL: https://siduction.org. With the backspace key you delete the last selected URL and enter the desired one.
Exit w3m with:

SHIFT+**Q**

More information can be found on the documentation page of w3m.

It is advisable to familiarize yourself with **elinks/w3m, irssi/weechat, midnight commander**. Print this file to have the information handy in case of an emergency.

1.5.6   inxi

inxi is a system information script that works independently of individual IRC clients. This script outputs various information about the hardware and software

being used, so that other users in #siduction can better help with troubleshooting. Alternatively, run it in a console to get information about your own system yourself.

To use inxi in `konversation`, type this into the chat box:

/cmd inxi -v2

To use inxi in `weechat`, enter this into the chat box:

/shell -o inxi -v2

This requires the shell extension to be installed.

See: https://www.weechat.org/scripts/

To use inxi in other clients, type this into the chat box:

/exec -o inxi -v2
or
/inxi -v2

Type the following command into a console:

```
$ inxi -v2
```

Help for inxi:

```
$ inxi --help
```

1.5.7   Useful links

Debian reference card - to print on a single sheet
HOWTOs from the Debian site (automatically in your language if browser is localized)
Debian Reference: Basics and System Administration (documents available as HTML, text, PDF, and PS)
Common Unix Printing System CUPS (In KDE, the KDE Help Center provides information about CUPS.)
LibreOffice (There is a wide choice in the "Help" menu.)

Last edited: 2024/04/23

1.6   Scripts for siduction

1.6.1   User executable

siduction provides some scripts that support administrative tasks and help with troubleshooting.

- **chroothelper**
  Change to a chroot environment

  chroothelper simplifies switching to a chroot environment on the live system. It searches the hard disks for installed Linux operating systems and offers to chroot to them.

- **sshactivate - sshdeactivate**
  Starts and enable sshd or stop and disable sshd

  sshactivate generates sshd system keys, starts and enable the ssh daemon and asks for user password, to allow remote access to the (live-)system. *sshdeactivate* stop and disable the ssh daemon.

- **fw-detect**
  Check system firmware status

  fw-detect scans the loaded modules on your system, checks if they require firmware and checks if the firmware is present in the correct location. By default it will only output information about any modules which appear to require firmware.

- **siduction-irc**
  Starts an IRC session to #siduction

  siduction-irc starts an IRC session with following clients depends on the environment, HEXCHAT, IRSSI, KONVERSATION, KVIRC, WEECHAT, XCHAT and connects to our IRC-channel #siduction on OFTC.

- **remove-nonfree**
  DFSG compliant installation

remove-nonfree makes the install DFSG compliant. Running it might on the other hand lead to a not working wifi or problems with your graphics card.

- **kernel-remover**

removes siduction kernels

kernel-remover removes unused kernels from the running system. It offers a selection of kernels, removes the selected ones and updates the boot menu.

1.6.2   Active in the background

- **siduction-btrfs**

Checks and updates the boot menu

siduction-btrfs is a compilation of scripts and systemd units that become active on an installation in the Btrfs file system. After a snapshot or rollback, the boot menu is checked and updated if necessary. *siduction-btrfs* works in the background without user input. As of version 0.2.0-5, *siduction-btrfs* also supports the boot manager *systemd-boot* and generates boot entries after a rollback for the rollback target. More information can be found in our systemd-boot manual page.

Last edited: 2024/08/30

2 Quickstart

For quick users, here is a brief summary of important information on using the Live
Medium and installing siduction.

- Essential chapters

- siduction-kernel and software packages

- Configuration of networks

- Quick and easy installation

Last edited: 2023/11/15

2.1 Introduction

siduction strives to be 100% compatible with Debian Sid. Nevertheless, siduction may offer packages that temporarily replace faulty Debian packages. The siduction apt repository contains siduction specific packages such as the siduction kernel, scripts, utilities, and documentation.
The distribution siduction is aimed at users who already have some experience with Linux and are not afraid of using the command line.

On the stability of Debian Sid

"Sid" is the name of the Debian unstable repository. Debian Sid is regularly updated with new software packages, which means that this Debian distribution contains the latest versions of the respective programs very promptly. However, this also means that there is less time between a release in the upstream (by the software developers) and the distribution in Debian Sid to test the packages.

2.1.1 Essential chapters

For users who are new to siduction, we strongly recommend reading the following chapters. In addition to basic information on Linux, they contain important, siduction-specific settings and procedures.

- Terminal/Console - describes how to use a terminal and the `su` command.

- Partitioning the hard disk - describes how to partition a hard disk.

- Downloading siduction ISO - describes how to download and check a siduction ISO file.

- ISO to USB stick - memory card - Describes how to create a siduction life media.

- Installation on a hard disk - describes how to install siduction onto a hard disk.

- Non-free drivers, firmware, and sources - describes how software sources can be adapted and non-free firmwares can be installed.

- Internet connection - describes how to connect to the Internet.

- Package Manager and System Update - describes how to install new software and update the system.

Last edited: 2023/11/15

2.2   Kernel and software packages

The Linux kernel of siduction is optimized to achieve the following goals: problem solving, enhanced and updated features, performance optimization, higher stability. The basis is always the latest kernel from http://www.kernel.org/.
siduction is based on Debian Sid, which can lead to Debian packages containing errors in rare cases. Therefore, we may offer packages that temporarily replace faulty Debian packages.

2.2.1   The management of software packages

siduction follows Debian rules regarding package structure and uses `apt` as well as `dpkg` for software package management. The Debian and siduction repositories are located in `/etc/sources.list.d/*`.

Debian Sid contains more than 20,000 program packages, so the chances of finding a program suitable for a task are very good. Information on how to search for program packages can be found here:
Search program packages .

A program package is installed with this command:

```
apt install <package_name>
```

See also: Install new packages.

New and updated software packages are pushed to Debian Sid Repositories four times a day. Quick package management is achieved by using a local database. The command

```
apt update
```

is necessary before each installation of a new software package to synchronize the local database with the repositories' software supply.

The use of other Debian based repositories, sources, and RPMs.
Installations from source code are not supported. It is recommended to compile as user (not root) and to place the application in the home directory without installing it

onto the system. The use of `checkinstall` to generate DEB packages should be limited to purely private use. Conversion programs for RPM packages like `alien` are not recommended either.

Other well-known (and lesser-known) Debian based distributions create new packages with a structure different from Debian. They often use other directories for programs, scripts, and files during installation, which can lead to unstable systems. Some packages cannot be installed at all because of unresolvable dependencies, different naming conventions, or different versioning. For example, a different version of glibc may result in the inability to execute any program at all.

For this reason, Debian's repositories should be used to install the required software packages. Other software sources may be difficult or impossible to support by siduction. This includes packages and PPAs from Ubuntu.

2.2.2   Updating the system - upgrade

An upgrade can only be performed when X graphics server is stopped. To stop the graphics server, the following command can be entered into a console as **root**:

```
init 3
```

After that, system updates can be performed safely. First, refresh the local package database with

```
apt update
```

Then update the system with one of the two alternatives

```
apt upgrade
apt full-upgrade
```

Afterwards, start the graphical user interface with the following command:

```
init 5
```

apt full-upgrade is the recommended procedure to upgrade a siduction installation to the latest version. It is described in more detail here:

Updating an installed system - full-upgrade.

Last edited: 2023/11/15

2.3   Configuration of networks

The `NetworkManager` integrated in all graphical user interfaces of siduction offers a quick configuration of network cards (WLAN and Ethernet). In the terminal, the script `nmcli` provides access to the functionality of the NetworkManager.

Wireless networks are scanned. You can connect to the networks found and make settings for the encryption method, the IPv4 or IPv6 Internet protocol and a proxy server. The backend is `iwd`. The Ethernet configuration takes place automatically when using a DHCP server on the router (dynamic assignment of an IP address), but there is also the option of a manual setup (from netmasks to name servers).

On the live medium and after installation, the NetworkManager with its backend iwd is already configured and ready for use.

2.3.1   NetworkManager

In the graphical user interface, the NetworkManager is located in the taskbar. It is largely self-explanatory.

With **nmcli** or **nmtui** a powerful command line client is available for the daily use of the NetworkManager. *nmtui* offers a user-friendly ncurses interface within the terminal. Here too, the function is largely self-explanatory. If the script is not available, install it with :

```
apt install network-manager
```

The start command in the console is **nmcli**.
More information at network - nmcli.

If you have all the necessary information to hand, a single command line is sufficient.
Example:
Device Name = wlan0
Device Type = wifi
SSID = HOME_WLAN
Passwodt = s3Hrg3he!m

Command to set up the WLAN connection:

```
nmcli device wifi connect "HOME_WLAN" password "s3Hrg3he!m"
```

Disconnect with:

```
nmcli device down wlan0
```

NetworkManager saves the connection data entered once. If you are within range of this router again, NetworkManager automatically re-establishes the connection. To change the behavior, please read the man pages
man NetworkManager
and
man nm-settings

Last edited: 2023/11/15

2.4 Quick installation

2.4.1 Five steps to the goal

1) Select the preferred flavor on the siduction download page and download the ISO file and the file with the associated checksum.
   Then check the download as described here.

2) Transfer the ISO file to a bootable medium.
   The instructions for a USB stick or an SD card or for burning to a DVD offer help.

3) Boot the siduction Live medium created in this way.
   Important:
   During the boot process, call up the EFI menu (depending on the manufacturer, press one of the keys `Del`, `F2`, `F11`…).
   In the EFI menu, deactivate both the *CSM (Compatibility Support Module)* mode and *Secure Boot*.
   Also select the Live Medium for booting in the EFI menu.

4) Set the desired language in the boot menu of the live medium and boot siduction.
   The user in the live medium is **siducer** and his password is **live**.
   No password is set for the user **root** (system administrator) in the live medium. In the terminal, either execute `sudo <command>` or become root by entering `su`.

5) Double-click on the `Install system` icon to start the installation on the hard disk.
   If partitioning manually, make sure that the first partition is the EFI system partition (esp) and that the *boot* and *esp* marks are set. It must be mounted under `/boot/efi/`.

Last edited: 2023/11/15

3   ISO images-en

This section contains information and notes on

- The Content of the live ISO,

  available variants, system requirements, applications, utilities, and the disclaimer.

- Using the live medium,

  the available users and their passwords, working with root privileges, and software installation during the live session.

- The boot options (cheat codes),

  in tabular form for both the live ISO and installed systems.

- Downloading and integrity check of ISOs,

  the mirror servers and the files on them, and the integrity check of the download.

- Write the ISOs onto a flash drive, SD or SDHC card,

  methods to write a siduction ISO image file as live media to a USB stick, SD card or SDHC card.

- Burning the ISOs onto a DVD with GUI,

  using programs on a graphical user interface on the operating systems Linux and Windows™.

- Burning the ISOs without GUI,

  by means of a whole set of directly usable terminal commands, which can also be used to determine available devices.

Last edited: 2023/11/09

3.1   Contents of the Live-ISO

3.1.1   Note about the software on the Live-ISO

siduction provides DFSG-free software on the Live-ISO as well as non-free firmware. To uninstall proprietary software, use the command **`apt purge $(`**↙ **`vrms -s)`** or our script **`remove-nonfree`** after installation.

The ISO is based exclusively on the latest Debian Sid at the time of release, enriched and stabilized with custom packages and scripts from the siduction repositories. The kernel we use is a patched version of the latest vanilla mainline kernel. ACPI and DMA are enabled.

A complete manifest file with a list of all installed programs for each release variant of siduction can be found on each download mirror.

3.1.2   Variants of the ISO

siduction offers seven current images-en in 64-bit as live ISO to get started with Debian Sid. Five of the images come with a preinstalled desktop environment. Typically, an installation takes between 1 and 10 minutes, depending on the hardware. The alternatives are:

1. **KDE Plasma 64-bit**, live-ISO with about 2.8 GByte:
   - Qt based Plasma Desktop and KDE frameworks; with a representative selection of KDE Applications
   - installation of additional applications easily possible via apt
2. **Xfce 64-bit**, live-ISO with about 2.3 GByte:
   - includes a GTK based desktop environment with all features (no minimal version!) and all productivity applications right away
   - resource requirements lower than for KDE
   - installation of additional applications easily possible via apt
3. **LXQt 64-bit**, live ISO with about 2.2 GByte:
   - includes desktop environment with a selection of Qt applications
   - footprint somewhat smaller than with Xfce
   - installation of additional applications easily possible via apt

4. **Xorg 64-bit**, live ISO with about 1.8 GByte:

- ISO image with an Xorg stack and the spartan window manager Fluxbox
- for users who want to build their system according to their own ideas

5. **NoX 64-bit**, live ISO with about 800 MByte:

- as the name implies, no pre-installed Xorg stack

32-bit ISOs are no longer offered by default.

If a 32bit ISO is desired, we will gladly create one on request in IRC. Unfortunately, we cannot test such an ISO.

3.1.3   Minimum system requirements

for: KDE-Plasma, Mate, Xfce, LXQt, LXDE, Cinnamon, Xorg, and NoX

Processor requirements: 64Bit CPU

AMD64

Intel Core2

Intel Atom 330

any x86-64/ EM64T capable CPU or newer

newer 64-bit capable AMD or Intel CPUs

(look for the "lm" flag in /proc/cpuinfo or use inxi -v3)

Memory requirements

```
KDE Plasma at least    4 GByte RAM
Xfce       at least    4 GByte RAM
LXQt       at least 512 MByte RAM
Xorg       at least 512 MByte RAM
NoX        at least 256 MByte RAM
```

At least 5 GByte hard disk space for NoX

At least 15 GByte of disk space for all the others.

At least 50 GBytes of disk space when installing on a partition formatted with Btrfs.

Other

VGA graphics card with at least 640x480 pixel resolution and optical drive or USB media.

3.1.4   Applications and utilities

As web browser, Firefox or Chromium is included (depending on the variant).

LibreOffice is pre-installed as office software. Dolphin, Thunar, and PCManFM are available as file managers.

Network Manager, Connman or iwd is available for network and internet configuration. The WLAN daemon used is `iwd`.

For disk partitioning, cfdisk, gdisk and cgdisk, and GParted are supplied. Gparted also provides the ability to resize NTFS partitions.

System analysis tools such as Memtest86+ (a tool for comprehensive memory analysis) are included, too.

Each ISO variant contains an extensive selection of applications for the command line. A complete manifest file with the installed programs for each release variant of siduction can be found on each download mirror.

3.1.5   Disclaimer

siduction is experimental software. Use at your own risk. The siduction project, its developers, and team members cannot be held liable under any circumstances for damage to hardware or software, lost data, or any other direct or indirect damage to the user by using this software. Anyone who does not agree to these terms may not use or distribute this software.

Last edited: 2023-11-07

3.2 How to use the live medium

3.2.1 Users set up on the live medium

The users **siducer** and **root** (the system administrator) are set up on the live medium.

The password for the user **siducer** is **live**.
No password is set for **root** (system administrator).

The live session will be locked after some time without any input. To unlock, please enter the username **siducer** and the password **live**.

3.2.2 chroot helper

A very helpful tool on the live medium is the *chroot helper*. Who can execute the commands to start a chroot off the cuff, most of us cannot. So the next time we need to repair our system, we use the *chroot helper* icon. A terminal window opens and we are asked to select a partition. After a security warning, the *chroot helper* redirects us to the corresponding installation.

3.2.3 root privileges on the live medium

Several ways of how to run a program with root priviliges are described below.

> **Caution**
> Whenever you work with root privileges, you should know exactly what you are doing. For web browsing and similar actions, root privileges are not necessary.

1. The easiest way is to open a terminal and get root privileges by typing `su`. To start a program that works with a graphical user interface, just enter the program name.

   ```
   root@siduction:~# gparted &
   ```

Now Gparted will be executed with root privileges. The *"&"* at the end of the command puts the process into the background so that the terminal remains usable.

2. Open a command prompt window:
Use the key combination **Alt**+**F2** to get a program launcher and enter the following command:

```
sudo <application>
```

A terminal window will open, asking you for the root password. Now simply press the **Enter** key, unless a temporary root password has been set as described below. In the latter case, the corresponding password must be entered.

3. Enter the following command into a terminal without root privileges:

```
sudo <application> &
```

Please note:
sudo is not preconfigured on hard disk installations. We recommend to directly use the real **root** account.
See why sudo is not configured.

3.2.4   How to set a new password

Remember: The livesession's standard user is **siducer** with the password **live**. If you want to change the password, open a terminal and enter the following commands:

```
siducer@siduction:~$ passwd
Enter a new password:
Re-enter the new password:
passwd: Password successfully changed
siducer@siduction:~$
```

This new password for **siducer** can be used for the rest of the live session.

The same procedure can be used to set a password for **root** in any terminal, but you have to become root via **su** first. Afterwards, a login on a virtual console as **root** is possible.

3.2.5   Software installation during live session

The command sequence for installing software during a live session is similar to that on a hard disk installation. The prerequisite is a root terminal:

```
apt update
apt install <the-package-you-want>
```

Otherwise, type `sudo` before the commands.

```
sudo apt update
sudo apt install <the-package-you-want>
```

However, if you shut down the live medium, no changes will be kept.

Last edited: 2023/11/10

3.3   Boot options and cheat codes

Info

This manual page contains tables of

1. siduction specific parameters (live medium only)
2. boot options for the graphics server X
3. general parameters of the Linux kernel
4. values for the general parameter **vga**

If the *"value"* field is non-empty, one of the possible values must be appended to the corresponding boot option with a = character. For example, if "1280x1024" is the desired value for the boot option `screen`, enter `screen=1280x1024` into the Grub command line. For language selection (here German), type `lang=de`. The Grub command line can be accessed by pressing the **e** key as soon as the Grub menu appears. After that, you are in edit mode. Now you can navigate to the kernel line with the arrow keys and insert the desired cheatcode(s) at the end. The space character serves as separator. The boot process can be continued with the key combination **Ctrl**+**X** or **F10**.

Detailed reference list for kernel boot codes from kernel.org

3.3.1   siduction specific parameters

These boot options apply only to the live medium.

boot option	value	description
blacklist	module_name	temporary deactivation of modules before udev becomes active
desktop	kde, gnome, fluxbox	select desktop environment
fromiso	Please read "booting 'fromiso' ".	

boot option	value	description
hostname	myhostname	changes the network name (hostname) of the live CD system
lang	be, bg, cz, da, de, de_CH, el, en, en_AU, en_GB, en_IE, es, fr, fr_BE, ga, hr, hu, it, ja, nl, nl_BE, pl, pt (pt_BR), pt_PT, ro, ru, zh	sets the language preference, the basic localization settings (locales), the keyboard layout (in the console and in X), the timezone, and the Debian mirror. In the long form `lang=ll_cc` or `lang=ll-cc`, *"ll"* refers to the language selection and *"cc"* to the keyboard layout, mirror server, and time zone selection (e.g. `lang=fr-be`). The default setting for English is en_US with UTC as the time zone and for German de with Europe/Berlin as the time zone. Example for a self-selected setting: `lang=pt_PT tz=Pacific/Auckland`
md5sum		tests the checksum of the live medium
noaptlang		prevents the installation of localization packages of the selected language
nocpufreq		does not enable speedstep/powernow
nodhcp		no DHCP (DHCP automatically tries to establish Ethernet connections)
noeject		does not remove CD/DVD from drive
nofstab		prevents writing a new fstab
nointro		skips the output of `index.html` when starting the live medium
nomodeset	radeon.modeset=0	together with `xmodule=vesa` allows a clean boot to X for Radeon cards in live mode
nonetwork		prevents automatic configuration of network interfaces at boot time
noswap		no activation of the swap partition

boot option	value	description
smouse		searches for serial mouse input devices using hwinfo
tz	tz=Europe/Dublin	sets the time zone. If the bios or hardware clock is set to UTC, `utc=yes` is specified. A list of all supported time zones can be viewed by copying & pasting `file:///usr/share/zoneinfo/` into the browser.
toram	copies the medium into RAM and boots from the RAM copy	

3.3.2   Boot options for the graphics server X

Either the `xandr` or `xmodule` boot option should also be used when applying boot options for the X graphics server for Radeon, Intel, or MGA graphics cards.

boot option	value	description
dpi	auto *or* DPI count	sets the desired pixels per inch for the monitor. The DPI is obtained by dividing the number of pixels of the monitor width by the diagonal (in inch) and multiplying the result by one of the following values: 1.25 for a 4:3 screen, 1.18 for a 16:10 screen, or 1.147 for a 16:9 screen. For a 24" screen with 1920x1080 resolution this results in 1.147x1920/24 dpi=92, or for a 15" screen with 1600x1200 resolution this results in 1.25x1600/15 dpi=133.

boot option	value	description
hsync	80	sets the horizontal frequency of the monitor (in kilohertz)
noml		prevents the X.org configuration from containing a list of modelines, thus causing the correct mode to be detected automatically
noxrandr		prevents the new X.org drivers from using the extensions of RandR 1.2 and uses the old techniques to query monitor properties
screen	1280x1024	sets custom resolution for X (1280x1024 or other screen resolutions)
vsync	(e.g.) 60	sets the vertical frequency of the monitor (in hertz)
xdepth	values: 8 15 16 24	set the color depth used by X.org (not all drivers support `1` and `4`)
keytable	(e.g.) us, de, gb	keyboard layout used by X.org
xkbmodel	(e.g.) pc105	keyboard type used by X.org (the number indicates the number of keys)
xkboptions	(e.g.) grp:alt_shift_toggle	assignment variant of the keyboard used by X.org
xkbvariant	(e.g.) nodeadkeys	set a layout variant of the keyboard
xmode	800x600	set the screen resolution according to the given value (`1024x768`, `1600x1200` etc.)
xmodule or xdriver	ati, fbdev, i810, intel, mga, nouveau, radeon, savage, vesa	uses the selected X module
xrandr		forces X.org configuration using the new RandR 1.2 extensions of the X.org drivers

boot option	value	description
xrate	XX	forces a preferred retry frequency for drivers supported by RandR 1.2. This option must be used in conjunction with the `xmode` boot option. Detailed documentation can be found here.
xhrefresh	(e.g.) 75	sets the horizontal frequency of the monitor for X (in kilohertz)
xvrefresh	(e.g.) 60	sets the vertical frequency of the monitor for X (in hertz)

3.3.3   General parameters of the Linux kernel

boot option	value	description
apm	off	disables Advanced Power Managment
1, 3, 5	(e.g.) 3	boot targets or runlevels which can be entered manually in the Grub boot line. See also the manual page Runlevel - target unit.
irqpoll		uses IRQ polling
mem	(e.g.) 128M, 1G	uses the specified memory size
noagp		no AGP support (Accelerated Graphics Port)
noapic		no APIC query (Advanced Programmable Interrupt Controller)
nodma		no support for DMA (Direct Memory Access)
noisapnpbios		does not perform an ISA "Plug and Play" query at startup
nomce		disables the kernel option "Machine Check Exception"
nosmp		does not use Symmetric Multi-Processor (multiple CPUs or CPUs with Hyper-Threading)
pci	noacpi	no ACPI for PCI devices

boot option	value	description
quiet		no output on screen
vga	normal	more about vga codes in the next paragraph
video	(e.g.)	for graphics cards with KMS enabled;
	DVI-0:800x600	applies to Intel and ATI graphics cards (the latter with Radeon driver); DVI-X/LVDS-X refers to video output shown by `xrandr`

3.3.4   VGA codes

The following tables list the values that can be specified with the general parameter `vga`.

An example of use is `vga=791` (VESA code, resolution 1024x768 with 64000 colors).

Problems with netbooks or other screen resolutions can be solved by entering `vga=0` in the grub line.

Decimal

colors	640x480	800x600	1024x768	1280x1024
256	257	259	261	263
32k	272	275	278	281
64k	273	276	279	282
16M	274	277	280	

hexadecimal

colors	640x480	800x600	1024x768	1280x1024
256	0x101	0x103	0x105	0x107
32k	0x110	0x113	0x116	0x119
64k	0x111	0x114	0x117	0x11A
16M	0x112	0x115	0x118	

VESA

colors	640x480	800x600	1024x768	1280x1024	1600x1200
256	769	771	773	775	796
32k	784	787	790	793	797
64k	785	788	791	794	798
16M	786	789	792	795	

Last edited: 2023/11/10

3.4   Downloading the ISO

Please use the closest mirror. Mirror servers listed below, with details for the entry
in
`/etc/apt/sources.list.d/siduction.list`, are updated in a timely manner.

Europe

- Office Vienna, Vienna, Austria
  https://siduction.office-vienna.at/

- Freie Universität Berlin/spline (Student Project LInux NEtwork), Germany
  http://ftp.spline.de/pub/siduction/
  https://ftp.spline.de/pub/siduction/
  ftp://ftp.spline.de/pub/siduction/

- University of Stuttgart, Germany
  https://ftp.uni-stuttgart.de/siduction/
  ftp://ftp.uni-stuttgart.de/siduction/

- Academic Computer Club, Umeå University, Sweden
  http://ftp.acc.umu.se/mirror/siduction.org/
  https://ftp.acc.umu.se/mirror/siduction.org/
  rsync://ftp.acc.umu.se/mirror/siduction.org/

- Dotsrc.org, Aalborg University, Denmark
  http://mirrors.dotsrc.org/siduction/
  https://mirrors.dotsrc.org/siduction/
  ftp://mirrors.dotsrc.org/siduction/
  rsync://mirrors.dotsrc.org/siduction/

- Yandex, Moscow, Russia
  https://mirror.yandex.ru/mirrors/siduction/
  http://mirror.yandex.ru/mirrors/siduction/
  ftp://mirror.yandex.ru/mirrors/siduction/
  rsync://mirror.yandex.ru/mirrors/siduction/

- GARR Consortium, Italy
  http://siduction.mirror.garr.it/
  https://siduction.mirror.garr.it/

- Quantum Mirror, Hungary
  http://quantum-mirror.hu/mirrors/pub/siduction/
  https://quantum-mirror.hu/mirrors/pub/siduction/
  rsync://quantum-mirror.hu/siduction/

- Belnet, Brussels, Belgium
  http://ftp.belnet.be/mirror/siduction/
  https://ftp.belnet.be/mirror/siduction/
  ftp://ftp.belnet.be/mirror/siduction/
  rsync://ftp.belnet.be/siduction/

- Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen, Germany
  https://ftp.gwdg.de/pub/linux/siduction/
  ftp://ftp.gwdg.de/pub/linux/siduction/
  rsync://ftp.gwdg.de/pub/linux/siduction/

- RWTH Aachen, Germany
  https://ftp.halifax.rwth-aachen.de/siduction/
  rsync://ftp.halifax.rwth-aachen.de/siduction/
  ftp://ftp.halifax.rwth-aachen.de/siduction/
  http://ftp.halifax.rwth-aachen.de/siduction/

- Studenten Net Twente, Netherlands
  http://ftp.snt.utwente.nl/pub/linux/siduction/
  https://ftp.snt.utwente.nl/pub/linux/siduction/
  ftp://ftp.snt.utwente.nl/pub/linux/siduction/
  rsync://ftp.snt.utwente.nl/siduction/

Asia

- KoDDOS, Amarutu Technology, Hong Kong
  https://mirror-hk.koddos.net/siduction/

http://mirror-hk.koddos.net/siduction/

rsync://mirror-hk.koddos.net/siduction/

South America

- Corporación Ecuatoriana para el Desarrollo de la Investigación y la Academia, Cuenca
  https://mirror.cedia.org.ec/siduction/
  http://mirror.cedia.org.ec/siduction/
  rsync://mirror.cedia.org.ec/siduction/

North America

- Department of Mathematics, Princeton University, United States
  http://mirror.math.princeton.edu/pub/siduction/
  https://mirror.math.princeton.edu/pub/siduction/

- Georgia Tech Software Library (GTlib), Atlanta, United States
  http://www.gtlib.gatech.edu/pub/siduction/
  ftp://ftp.gtlib.gatech.edu/pub/siduction/
  rsync://rsync.gtlib.gatech.edu/siduction/

- Liquorix.net, United States
  https://liquorix.net/siduction/

3.4.1   Files on the siduction mirrors

Each mirror includes the following files:

siduction-20xx-xx-release-name-window-manager-arch-
datetimestamp.arch.manifest
siduction-20xx-xx-release-name-window-manager-arch-datetimestamp.iso
MD5SUM
SHA256SUM
SOURCES

The `xxx.manifest` file lists all packages of the respective ISO.

`xxx.iso` is the image file provided for download.

The `xxx.md5` and `xxx.sha256` files are used to verify the integrity of the ISO. `xxx.sources` contains the download links to the source code files of the packages used.

Download links and mirrors can be found at siduction.org.

The tar archive with the sources is interesting for those who want to redistribute siduction. Here, the source code must be published to comply with the license. More information can be found in the tar archive.

If someone can provide an FTP server with appropriate traffic, we are always available in the siduction forums or in IRC irc.oftc.net:6697 #siduction-en.

3.4.2   Integrity check

md5sum

An md5sum is the checksum of a file and is used to check the integrity of the associated file. The siduction ISO file and its respective m5sum files can be downloaded from the same directory. For example:

```
siduction-21.3.0-wintersky-kde-amd64-202112231751.iso
siduction-21.3.0-wintersky-kde-amd64-202112231751.iso.md5
```

During the integrity check, a md5sum is created for the downloaded ISO file and then compared to a sum in the file with the `.md5` extension that we have created in advance. If the check sums deviate, the has been changed or damaged. This test protects you from using a manipulated ISO file and saves you a lot of time for debugging in case of a non functioning DVD.

On Linux, use the terminal and navigate to the directory containing both the ISO file and the `.md5` file. Then you can get the ISO file's checksum by entering **md5sum**↙ **siduction-*.iso** and the the `.md5` file's content with **cat siduction-*.iso**↙ **.md5**. If you combine the two commands, the output is given one upon the other and is thus easy to compare.

```
$ md5sum siduction-*.iso && cat siduction-*.iso.md5
```

```
358369ebc617613e3c58afc1af716827  siduction-21.3.0-wintersky-↲
   kde-amd64-202112231751.iso
358369ebc617613e3c58afc1af716827 *siduction-21.3.0-wintersky-↲
   kde-amd64-202112231751.iso
```

The check is made even easier on Linux with the `md5sum -c`. Note that you need to specify the .md5 file in this command.

```
    (command and output in case of success)
$ md5sum -c siduction-21.3.0-wintersky-kde-amd64↲
   -202112231751.iso.md5
siduction-21.3.0-wintersky-kde-amd64-202112231751.iso: OK


    (command and output in case of error)
$ md5sum -c siduction-21.3.0-wintersky-kde-amd64↲
   -202112231751.iso.md5
siduction-21.3.0-wintersky-kde-amd64-202112231751.iso: ↲
   FEHLSCHLAG
md5sum: WARNUNG: 1 berechnete Prüfsumme passte NICHT
```

sha256sum

A check using the sha256sum works exactly like the one with md5sum. The major difference is the increased security due to a 256 Bit check sum (md5sum: 128 Bit).

Windows

If you have downloaded the siduction ISO file on Windows 7 or later, the Powershell provides the preinstalled `CertUtil` helper program to create check sums. You can call it like this:

```
CertUtil -hashfile C:\TEMP\<my_ISO_file.img> MD5
    or
CertUtil -hashfile C:\TEMP\<my_ISO_file.img> SHA256
```

On older Windows versions you can use the `md5summer` program (486 kB) published under the General Public License.

Last edited: 2024/04/23

3.5 ISO to USB stick - memory card

Below we describe methods to write a siduction ISO image file as live media to a USB stick, SD card, or SHDC card.

Prerequisites

- The PC's BIOS must allow booting from a USB stick or SD card. Normally this is the case if the BIOS offers this boot option.
- A USB stick or SD card with a recommended capacity of at least 4 GB.
- Back up all your data on the devices you want to use for making the siduction live media in advance as well as the currently used operating system and your private data. A small typo or a hasty click can destroy all your data!

> Important information
> The following methods will overwrite existing partition tables on the target media, causing all data to be lost. Take extreme care when selecting the target media and its drive label.

3.5.1 GUI application

For Linux™, RasPi™, MS Windows™, or Mac OS X™

The small tool USBImager is available for all the above operating systems and is used to backup data and create the live medium. The program is open source and licensed under the MIT license. Download the necessary file for your operating system and install the program according to the instructions on the download page.

The handling is very simple thanks to the no-frills interface.

Write the image file to the device: 1. Select an image by clicking on `...` in the first line. 2. Select a device by clicking on the 3rd line. 3. Click on the `Write` button in the 2nd line.

Detailed information can be found in the Readme of the project page.

3.5.2 Linux command line

We recommend using the command line. There is no need to install additional programs, since all the tools you need are already available. A single, easy-to-understand command line is sufficient to transfer the siduction ISO image file to the storage medium.

Before we write the siduction ISO image file to the storage medium, we need to determine its drive label. The easiest way is to use journald. The command **`journalctl -f`** executed in a terminal shows continuously the messages of systemd. Now we plug in the storage medium and watch the messages in the terminal. Lines of the following type contain the information we are looking for.

```
kernel: usb 2-3.3: new high-speed USB device number 7 ...
[...]
kernel: scsi 1:0:0:0: Direct-Access Intenso Alu Line ...
kernel: sd 1:0:0:0: Attached scsi generic sg1 type 0
kernel: sd 1:0:0:0: [sdb] 7866368 512-byte logical blocks: ↙
    (4.03 GB/3.75 GiB)
[...]
kernel: sd 1:0:0:0: [sdb] Write cache: disabled, read ...
```

This is an Intenso USB flash drive with 4 GB storage capacity and a sector size of 512 bytes. The drive name is `sdb`. It follows that `/dev/sdb` is the path to use for the target medium.

Assuming the siduction ISO image file is stored in the `/home` directory of user **tux**, we can use the `dd` or `cat` commands to write to the target medium. The commands require root privileges. Therefore, depending on the system, either prepend **`sudo`** or **`doas`**, or use a terminal and become **root** with **`su`**.

```
dd bs=1M oflag=sync status=progress if=/home/tux/siduction↙
    -21.3.0-wintersky-kde-amd64-202112231751.iso of=/dev/sdb
     (or)
cat /home/tux/siduction-21.3.0-wintersky-kde-amd64↙
    -202112231751.iso > /dev/sdb
```

The copying process may take 15 minutes or longer for an ISO image file of about 3 GB. Please wait relaxed until the prompt returns.

3.5.2.1   Additional data partition   Usually the storage medium is much larger than the ISO image file. The methods shown so far all use the entire storage medium, although the ISO image file only occupies 2.9 GiB. This cannot be changed afterwards. It is a good idea to take advantage of the command line and set up two partitions in advance. The first partition will later contain the live system and the second one the otherwise unused space. This allows us to take data on the media to the live session and store it there during the live session.

We use as root the command `cgdisk /dev/sdb` to create a new GUID partition table (see the manual page Partitioning with gdisk) and use the following data:

1st partition:
Start sector: 64 (default)
Size: 3G (3 GB, slightly larger than the ISO image file)
Type hex code: 0700 (Microsoft basic data)
Name: siduction
2nd partition:
Start sector: xxxxxxxx (default, 1st sector after the previous partition).
Size: xxxxxx (default, the maximum possible size)
type hex code: 8300 (Linux)
Name: data

We write the partition table to the medium and exit `cgdisk`, but still stay in the root console, because the second partition still needs a file system and a meaningful label to make it easier to find in the file manager during the life session after mounting. The commands are:

```
mkfs.ext4 -L LifeData /dev/sdb2
```

With the storage medium prepared in this way, we write the ISO image file to the **1st partition**.

```
dd if=/home/tux/siduction-21.3.0-wintersky-kde-amd64↙
    -202112231751.iso of=/dev/sdb1
```

Please pay attention to `/dev/sdb1`. If only `/dev/sdb` is used, the dd command will mercilessly overwrite our newly created partition table.

3.5.3   Mac OS X command line

The copy process is very similar to the procedure for a Linux operating system. Connect your USB device, Mac OS X should mount it automatically. In the Terminal (under `Applications` > `Utilities`), run this command:

```
diskutil list
```

Determine the name of the USB device and unmount the partitions. In our example the name is `/dev/disk1`:

```
diskutil unmountDisk /dev/disk1
```

Assuming the siduction ISO image file is stored in the `/home` directory of user **steve**, and the USB device is named `disk1`, execute the following command:

```
dd if=/Users/steve/siduction-21.3.0-wintersky-kde-amd64↙
    -202112231751.iso of=/dev/disk1
```

Last edited: 2022/04/11

3.6 Burn ISO

Before burning the ISO image file to a DVD, you should always check it using the md5sum or sha256sum provided by siduction. This may save a lot of time troubleshooting a changed or corrupted file.

Detailed instructions can be found in the manual chapter ISO Download, Integrity Check.

> **IMPORTANT INFORMATION:**
> siduction, as a Linux LIVE DVD, is very heavily compressed. For this reason, special attention must be paid to the burning method of the image. Please use high quality media, burning in DAO mode (disk-at-once), and not faster than eight times (8x).

We recommend, however, if the hardware supports booting from USB, to put the image on a USB stick or SD memory card. Instructions for this can be found on the manual page ISO to USB stick / memory card.

3.6.1 Burn DVD with Linux

If you already have Linux on your computer, you can create the DVD with any installed burning program. Depending on the desktop environment, these are the programs

+ **K3b** for KDE

+ **Brasero** for Gnome

+ **Xfburn** for XFCE, LXQt, and Gnome

The burning programs are largely self-explanatory in their operation.

In K3b you select `More actions...` -> `Write image....`

In Xfburn and Brasero you should click `Burn image`.

Then select the ISO file to be burned (e.g. siduction-21.3.0-wintersky-kde-amd64-202112231751.iso) and set the burning mode `DAO` (Disk At Once) or `Automatic` and start the burning process.

Occasional problems when burning the live DVD are mostly caused by the graphical frontend applications. This can be worked around by using the very easy to

use script `burniso` on the console. The manual page Burn DVD without GUI explains the use of burniso briefly and exactly, as well as other commands to detect available hardware, compile data, and burn CD/DVDs.

3.6.2   Burn DVD with Windows

Of course, you can also burn the DVD on Windows. The downloaded file must be burned to a DVD as an ISO image and not from Windows Explorer as a file.
There are several good programs that extend the built-in CD and DVD burning feature introduced with Windows Vista to burn ISO files. Here are just two examples.

- The current version of the open source software cdrtfe is compatible with Windows Vista, 7, 8, 10, and 11. The program can be used to burn ISO images-en, create data discs (CD, DVD, BD), and audio as well as video CD/DVDs. You can install it on Windows or download the zip archive and run `cdrtfe` after unpacking it without any further installation.

- The closed-source software `CDBurnerXP` is a free program that can create data and audio CD/DVDs in addition to burning ISO images-en, and erases rewritable media if necessary. Available from CDBurnerXP.

Last edited: 2023/11/10

3.7 Burn Live-DVD without GUI

> **IMPORTANT INFORMATION:**
> siduction, as a Linux LIVE DVD, is heavily compressed. For this reason, special attention must be paid to the burning method of the image. Please use high quality media, burning in DAO mode (Disk-At-Once), and not faster than eight times (8x).

You don't necessarily need a graphical user interface (GUI) to burn a CD/DVD. Problems that occur during burning are usually caused by frontends like K3b, not so often by backends like growisofs, wodim, or cdrdao.

Before burning the ISO image file to a DVD, you should always check it using the md5sum or sha256sum offered by siduction. This may save a lot of time troubleshooting a changed or corrupted file.
Detailed instructions can be found in the manual chapter ISO Download, Integrity Check.

3.7.1 burniso

siduction provides a script called `burniso`.
It burns ISO image files, using wodim in Disk-At-Once mode with a fixed burning speed of 8x. First burniso tests if the necessary hardware is available and then lists all recognized ISO image files.

As **user**, change to the directory with the ISO image files and call `burniso`:

```
$ cd /path/to/ISO
$ burniso
Using device /dev/sr0.
Choose an ISO to burn:
1) siduction-21.3.0-wintersky-kde-amd64-202112231751.iso
2) siduction-21.3.0-wintersky-lxqt-amd64-202112231805.iso
3) siduction-21.3.0-wintersky-xfce-amd64-202112231826.iso
#? _
```

After entering the number for the desired ISO image file, `burniso` checks the integrity if there is an associated `.md5` file in the same directory. If successful, the burning process starts immediately afterwards. Therefore you should make sure that the medium to be burned to is already inserted before starting the script.

Burniso perfects and simplifies one single function for the user, namely burning ISO images-en. In addition, the command line programs offer all the possibilities to create media with data of various types on CD, DVD, and BD. In the following chapter we show some examples that are often used.

3.7.2   Burning with cdrdao wodim growisofs

The command line programs are the basis for the popular GUI programs like `K3b`, `Brasero`, or `Xfburn`. Those who prefer the full range of options offered by the programs `cdrdao`, `wodim`, `growisofs`, etc. use the command line. We present only a minimal part of the possibilities here. Studying the manpages should be self-evident and is a bit easier with the examples. In addition, tips for your own project can be found on the Internet with the search engine of choice.

3.7.3   Available devices

If the available hardware for burning is not exactly known, the programs wodim and cdrdao analyze the device data and output the information. First `wodim` for an external DVD writer to USB:

```
$ wodim -checkdrive
Device was not specified. Trying to find an [...] drive...
Detected CD-R drive: /dev/sr0
[...]
Vendor_info    : 'HL-DT-ST'
Identification : 'DVDRAM GP50NB40 '
Revision       : 'RB00'
Device seems to be: Generic mmc2 DVD-R/DVD-RW.
Using generic SCSI-3/mmc DVD-R(W) driver (mmc_mdvd).
Driver flags   : SWABAUDIO BURNFREE
Supported modes: PACKET SAO
```

The output for the same device with `cdrdao`:

```
$ cdrdao scanbus
Cdrdao version 1.2.4 - (C) Andreas Mueller
/dev/sr0 : HL-DT-ST, DVDRAM GP50NB40 , RB00
```

Another example with `wodim` on another PC with two IDE/ATAPI devices:

```
$ wodim --devices
wodim: Overview of accessible drives (2 found) :
-----------------------------------------------------------
0 dev='/dev/scd0' rwrw-- : 'AOPEN' 'CD-RW CRW2440'
1 dev='/dev/scd1' rwrw-- : '_NEC' 'DVD_RW ND-3540A'
-----------------------------------------------------------
```

To use the correct recorder, we first of all need the exact name for of the device file (*"/dev/sr0"* or *"/devscd1"*).

3.7.4 Examples for CD DVD BD

In the examples, we do not provide extensive explanations of the options used. Please consult the man pages for detailed information.

Burning a CD/DVD from an ISO image

Wodim recognizes by the filename extension `*.iso` and the option `-dao` that an image is to be burned.

```
$ wodim dev=/dev/scd0 driveropts=burnfree,noforcespeed fs=14M↙
    speed=8 -dao -eject -v <image.iso>
```

If you get an error message concerning *"driveropts"*, this is because burnfree is not possible on some burners. This is solved by removing the driveropts from the command.

```
$ wodim dev=/dev/sr0 fs=14M speed=8 -dao -eject -v <image.iso↙
    >
```

With `genisoimage` and `growisofs` you can create an ISO image file from a folder and all subfolders and burn it afterwards.

```
    (create ISO)
$ genisoimage -o <my-image.iso> -r -J -l <directory>
    (burn ISO)
$ growisofs -dvd-compat -Z /dev/dvd=<my-image.iso>
```

Burn a CD using a bin/cue image:

```
$ cdrdao write --speed 24 --device ATA:1,0,0 --eject filename↙
    .cue
```

Erase a rewritable blank disk

In order to add new data to rewritable media, it must first be erased. The commands for deleting the tables of contents are:

```
$ wodim -blank=fast -v dev=/dev/scd0
    (or)
$ cdrdao blank --device ATA:1,0,0 --blank-mode minimal
```

If you want to overburn the entire data, use `-blank=all` for wodim and `-blank↙-mode full` for cdrdao.

Copy CD/DVD

It is possible to copy even if there is only one drive. After reading, the source media is ejected and you have to insert the blank blank media into the same drive to continue.

```
$ cdrdao copy --fast-toc --device ATA:1,0,0 --buffers 256 -v2
```

You can copy a CD on the fly if two drives are available.

```
$ cdrdao copy --fast-toc --source-device ATA:1,1,0 --device ↙
    ATA:1,0,0 --on-the-fly --buffers 256 --eject -v2
```

Burn an audio CD

Burn all wav files in the current folder at 12x speed.

```
$ wodim -v -eject -pad -dao speed=12 dev=/dev/scd0 defpregap↵
   =0 -audio *.wav
```

Burn files to DVD

```
$ growisofs -Z /dev/dvd -R -J file1 file2 file3 ...
```

If there is still space on the DVD, you can add files using the `-M` option.

```
$ growisofs -M /dev/dvd -R -J file8 file9
```

This command fills the remaining free space on the DVD with zeros and closes the media.

```
$ growisofs -M /dev/dvd=/dev/zero
```

Last edited: 2023/11/10

4 Installation

This section contains informations and notes on

- Installation from live media to HDD, the necessary preparations, partitioning, and instructions for the installation program Calamares.

- Booting without installation from an ISO file.

- Partitioning of installation media, with examples of different disk sizes and single or dual boot.

- Naming of block devices (UUID), the different types of naming, the use of labels, the adjustment of the `fstab`, and the creation of new mount points.

- Partitioning with GParted on the graphical user interface.

- Partitioning with gdisk according to the UEFI-GPT standard in the terminal.

- Partitioning with fdisk based on the conventional BIOS with MBR partition tables (should only be used on old hardware).

- LVM partitioning (Logical Volume Manager) - in six steps to the goal, and the management of *logical volume*.

- moving private data from /home directory, for example to make one data partition available for multiple operating systems on parallel installations.

Last edited: 2022/04/01

4.1 Installation on HDD

4.1.1 Data backup

> **IMPORTANT: ALWAYS CREATE A DATA BACKUP!**
> If the installation target is already home to an operating system or data is to be preserved, please always create a backup before installing siduction.

4.1.2 Installation preparations

First, change the boot order so that the medium to be booted (DVD, flashcard, or USB stick) is at the top of the list. On most computers, pressing the `F2` or `Del`↵ key during the boot process takes you to the UEFI or BIOS setup. Alternatively, pressing `F12`, `F11`, `F7`, or `F8` (depending on the hardware manufacturer's specifications) during the boot process will take you directly to the boot menu where you can select the live media as the boot drive.

siduction usually starts without problems now. If this is not the case, boot options (cheat codes), which can be passed to the boot manager, are helpful. The manual page Cheatcodes explains the possible options.
At the start screen, use the arrow keys to navigate to *"From CD/DVD/ISO: …"* or *"From Stick/HDD: …"* (according to the used live medium) and press `e`. This takes you to the kernel command line where you can add the cheatcodes. Pressing `F10` will continue the boot process.

Before the installation, please remove all USB sticks, cameras, etc.

If siduction is not to be installed from, but **to a USB medium**, a different procedure is necessary. See the manual page Installation to a USB medium.

HDD, RAM, and Swap

The minimum requirements for installing the siduction variants are described on the manual page Live ISO content.
With 15 GB hard disk space and 2 GB RAM you are currently on the safe side. When installing on a partition formatted with Btrfs, we advise 50 GBytes of disk space.

A swap partition should be created on PCs with 1 GByte RAM or less. More than 2 GByte swap is rarely required and only useful for suspend to disk and server systems.

4.1.3   Partitioning

The partitioning of the drives depends on many factors:

- the chosen siduction variant
- size of the available drives and RAM
- single-boot or dual-boot with an already installed system (Windows, Linux, MAC)
- sharing of data for the installed systems

Examples and sizes for different installation situations are described on the manual page Partitioning.
We recommend leaving the `/home` directory on the root partition. The `/home` directory should be the place where individual configurations are stored, and only those. For all other private data, including `.ssh`, `.gnupg`, and the mail archives, a separate data partition should be created and linked to the `/home` directory if necessary. The advantages for data stability, data backup, and also in case of data recovery are almost immeasurable.

The partitioning can be done during installation or already in advance during the live session with the following programs:
Gparted, a graphical user interface program for GTK desktops
KDE Partition Manager, another graphical user interface program for Qt desktops
gdisk, recommended for UEFI hardware with GTP partition tables
cfdisk, only for older hardware with traditional BIOS and MBR partition tables

4.1.4   File systems

We recommend the **ext4** file system, which is used as the default file system on siduction. This applies to all partitions if only Linux operating systems are used.

For a dual-boot installation with *Windows*, a separate data partition with the **NTFS** file system makes sense. Linux can read and write to it; on Windows it is the default file system.

For a dual-boot installation with *MAC*, it also makes sense to have a separate data partition, but with the **HFS** or **HFS+** file system. Linux and MAC can have read and write access to it.

4.1.5   Duplication to another computer

The following console command creates a list of installed software packages. This list can be used to install an identical software selection on another computer or in the event of a new installation:

```
~# dpkg -l|awk '/^ii/{ print $2 }'|grep -v -e ^lib -e -dev -e↙
    $(uname -r) >/home/username/installed.txt
```

We recommend to copy this text file to a USB drive or a disk of your choice.
The text file can then be copied to the target systems `$HOME` directory and be used as a reference to install the required program packages. You can install the complete package list via

```
~# apt install $(/home/username/installed.txt)
```

4.1.6   The Calamares installer

During the installation, the computer should preferably be connected to the Internet because Calamares uses the GeoIP service to determine default settings for localization and time.

1. The installation program can be started comfortably via the icon on the desktop or in the menu: *"System" > "Install system"*.

2. After a double click on the icon, Calamares starts and we see the "*Welcome*" window.
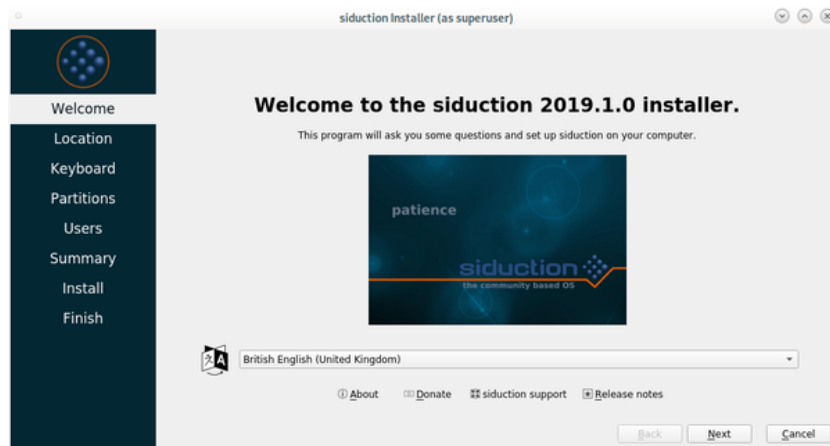
If an internet connection is provided, the correct language should already be set here.

3. In the next window ("*Location*"), you have the possibility to make changes to *region*, *timezone*, and *system language*, as well as the date and number *format*.
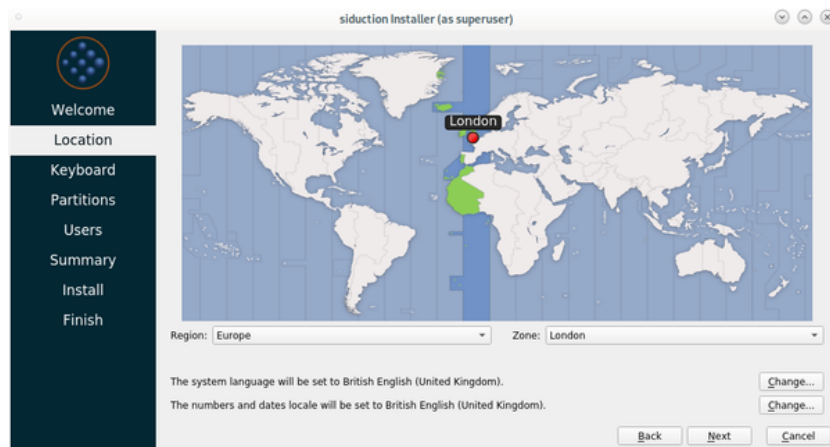
4. Next, you can set up the keyboard.
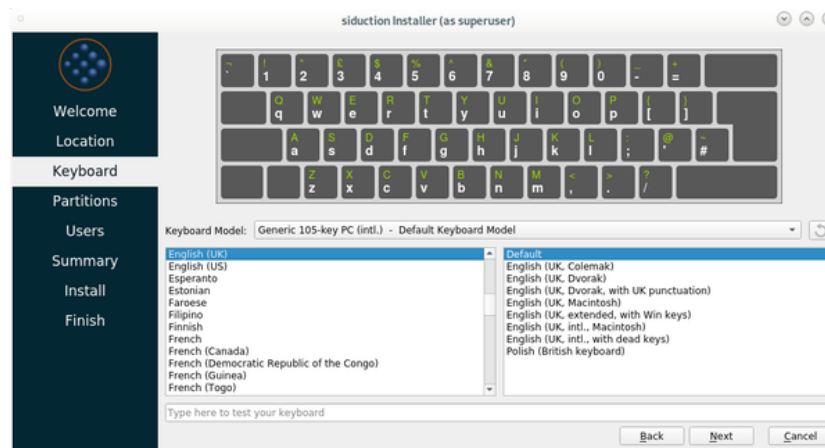
In the upper section, the keyboard is displayed graphically and the changes are visible immediately. At the bottom, there is an input line to test the keyboard layout.

5. Then we reach the already mentioned partitioning, which determines the parts of the harddisk(s) siduction uses.
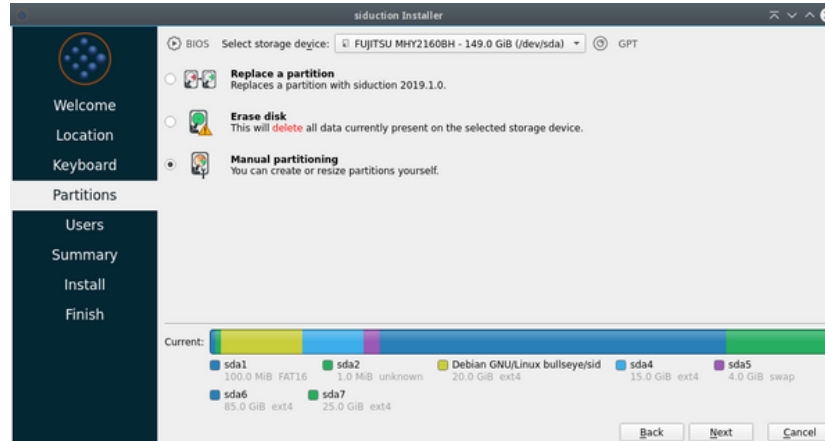
In our example, we use *"Manual partitioning"* because the partitions have already been created in advance and we only need to select the correct installation target. After clicking *"Next"*, the following window appears where we can select and edit the individual partitions.
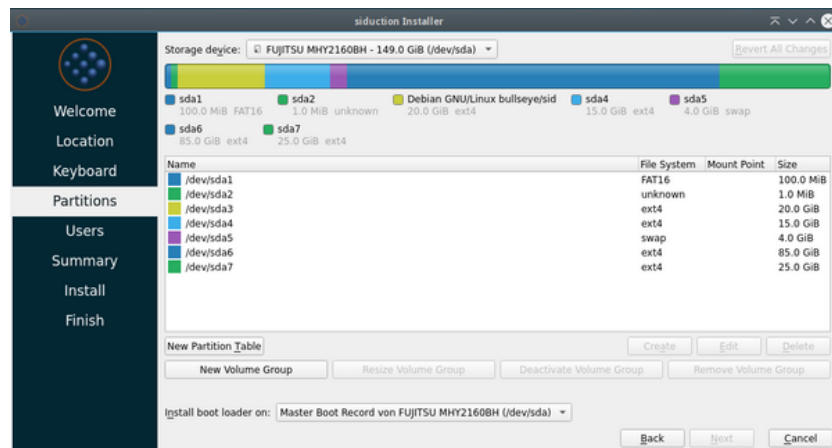
We use the partitions:

sda7 for `/` (root)

sda6 for `/data` together with the Linux system already present on sda3 and sda4
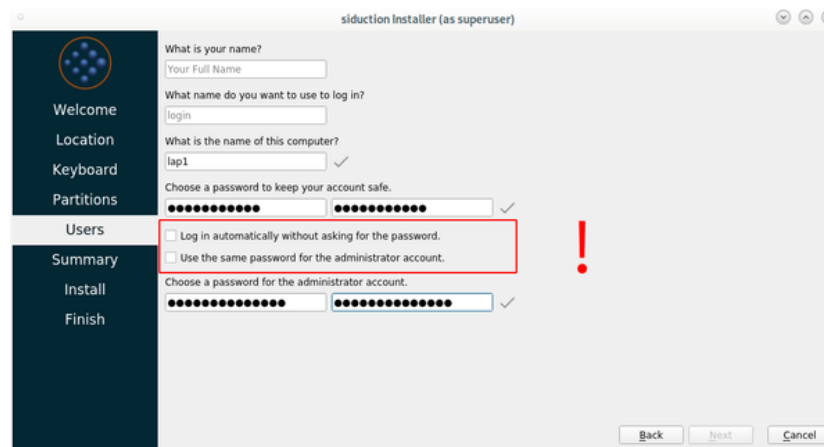
After selecting the partition in question and pressing the *"Change"* button, a window opens where we enter the above mountpoint and also format sda7 with the **ext4** file system. The partition sda6 is not formatted because we want to use the data already stored there together with the existing Linux system.

We do not need to edit the swap partition (sda5) since it will be automatically detected and integrated during the installation.

We can see the result of our efforts in the next image.
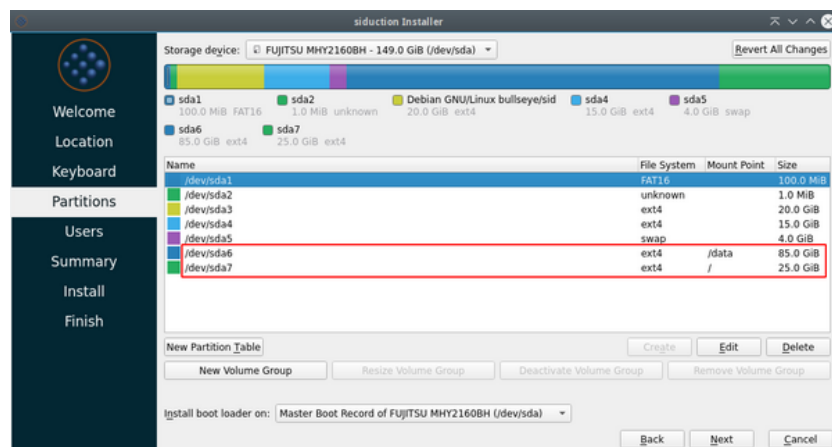
6. Next, we set username, login name, computer name, user password, and root password (remember them well!). The passwords should not be too simple for security reasons. Additional users can be added after installation in a terminal with adduser.

We explicitly recommend not to use the options
"*Log in automatically without password prompt*" and
"*Use the same password for the administrator account*".
They both represent a security risk on their own (see also sudo). If both options are enabled, entering passwords is just a farce!

7. After pressing the *"Next"* button, a summary of all previously made entries appears. Now you still have the possibility to make changes via *"Back"*. If you are satisfied with the result, a click on *"Install"* opens the small warning window in which you have to confirm the installation.
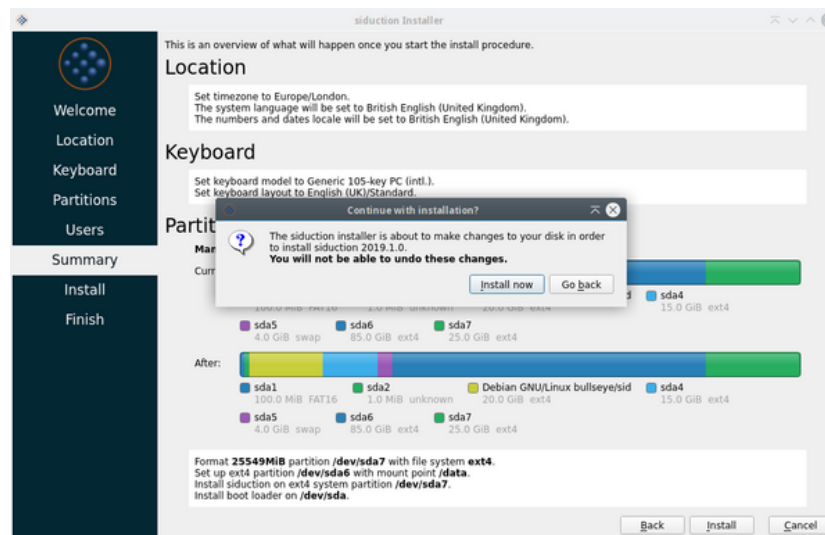
8. Now the installation starts. This takes some time depending on the hardware. The progress will be displayed respectively. Even if it takes a little longer, please do not abort the installation, but give the process time.
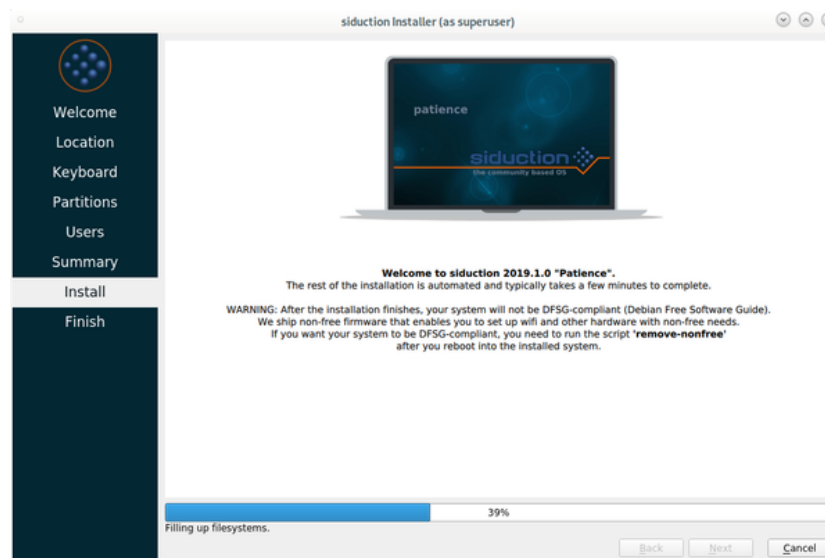
9. At the end, we get the possibility to reboot into the newly installed system.
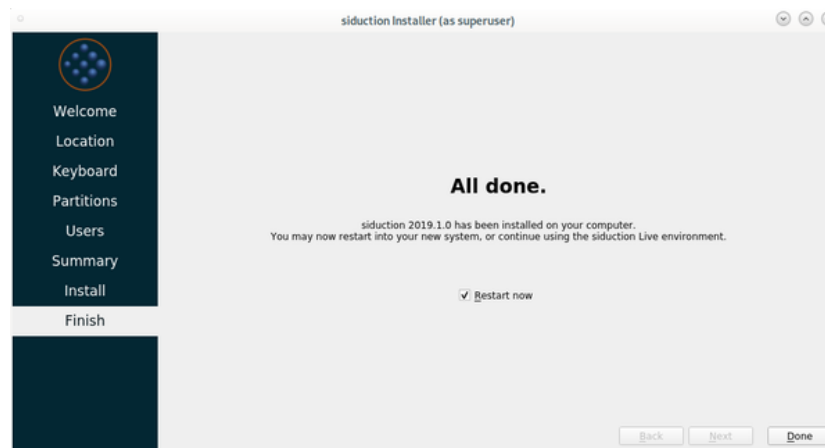
Remove the CD from the drive before rebooting!

4.1.7   Encrypt system

Since *siduction 2021.2 Farewell*, the installer provides the possibility to install the complete system onto an encrypted partition or hard disk. Only the first stage of the boot manager *Grub* is located (without encryption) on the *"BIOS-boot"* partition. Grub asks for the encrypted installations password at the beginning of every boot process before the boot menu appears.
The partitioning described in step 5 above is now slightly different. When selecting one of the first three options, the function *"Encrypt system"* appears in addition, where we also enter the password at the same time.
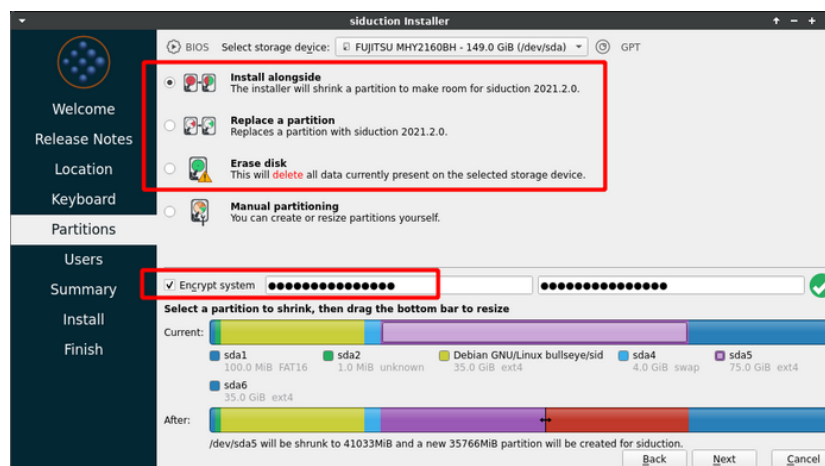
If we use *"Manual partitioning"*, we need an empty, unused area on the hard disk.
In case this is not available, we first delete one or more partitions that are no longer
needed. Then we create the new partition.
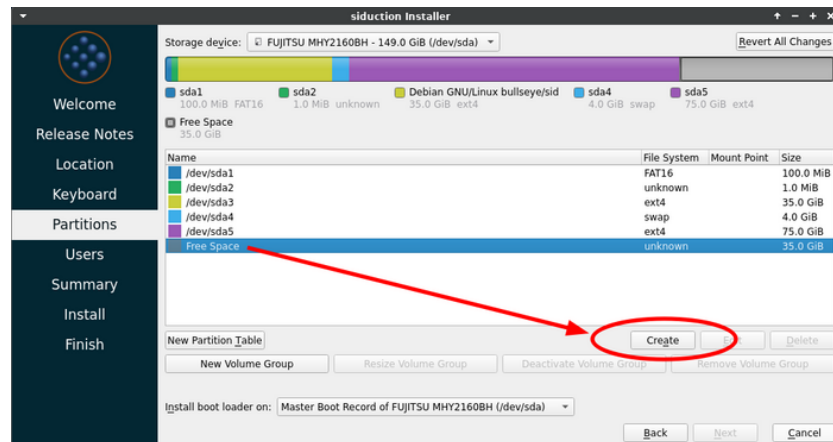
In the next step, the function *"Encrypt"* is selectable now.
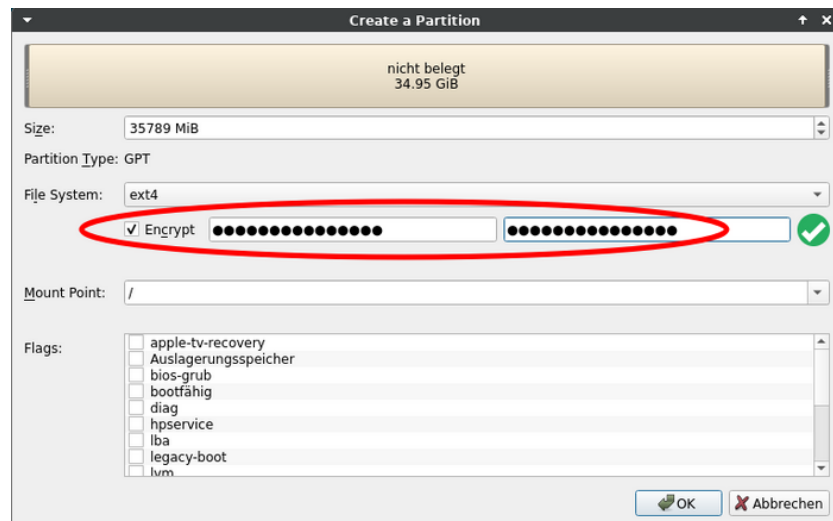
We enter our password and then select the root directory */* as mount point.
After finishing the partitioning, we continue the installation with the menu item
"User" as described above in step 6.

4.1.8   Add user

To add new users with automatic takeover of group permissions, run the following command as **root**:

```
~# adduser <username>
```

Pressing the **Enter** key will bring up more options that allow additional settings. Finally, a prompt appears, asking to enter the password twice.

siduction specific desktop icons (for the manual and IRC) must be added yourself.

To remove a user, enter:

```
~# deluser <username>
```

More information:

```
man adduser
man deluser
```

Last edited: 2022-12-14

4.2   Boot from ISO file

4.2.1   Overview

This cheat code boots from an ISO file located on the hard drive with an **ext4** file system. **For normal use, we recommend siduction's default file system, ext4**, which is well maintained.

Booting from a "fromiso" hard disk installation takes only a fraction of the time it takes to boot from a CD. In addition, the CD/DVD drive is available at the same time. Alternatively you can use VBox, KVM, or QEMU.

Prerequisites

- a working Grub installation (on floppy, a hard disk installation, or the live CD)
- a siduction image file, e.g. `siduction.iso` (name shortened) and a Linux filesystem like **ext4**

4.2.2   fromiso with grub2

siduction provides a grub2 file named `60_fll-fromiso` to generate a fromiso entry in the grub2 menu. The configuration file for fromiso can be found in the package `grub2-fll-fromiso`, with the path `/etc/default/grub2-fll-fromiso`.

First, open a terminal, become **root** and install `grub2-fll-fromiso`:

```
su
apt-get update
apt-get install grub2-fll-fromiso
```

Then, open the configuration file in an editor of your choice (`kwrite`, `mcedit`, `vim`, …):

```
mcedit /etc/default/grub2-fll-fromiso
```

In the lines that should be active, remove the comment sign (#) and replace the default statements inside the double quotes (**"**) with your own parameters.

Example: compare this modified `grub2-fll-fromiso` with the default settings:

```
# defaults for grub2-fll-fromiso update-grub helper
# sourced by grub2's update-grub
# installed at /etc/default/grub2-fll-fromiso
# by the maintainer scripts


#
# This is a POSIX shell fragment
#


# specify where to look for the ISO
# default: /srv/ISO
## Attention: This is the path to the directory where the
## ISO(s) are located, the path should not include the
## actual siduction.iso.
FLL_GRUB2_ISO_LOCATION="/media/disk1part4"


# array for defining ISO prefices --> siduction-*.iso
# default: "siduction- fullstory-"
FLL_GRUB2_ISO_PREFIX="siduction-"


# set default language
# default: en_US
FLL_GRUB2_LANG="de_DE"


# override the default timezone.
# default: UTC
FLL_GRUB2_TZ="Europe/Berlin"


# kernel framebuffer resolution, see
# http://manual.siduction.org/de/cheatcodes-vga-de.htm#vga
# default: 791
#FLL_GRUB2_VGA="791"


# additional cheatcodes
# default: noeject
```

```
FLL_GRUB2_CHEATCODE="noeject nointro"
```

Save the changes, close the editor and execute the following command as **root** in a terminal:

```
update-grub
```

This will update the grub2 configuration file `grub.cfg` to recognize the ISOs placed in the specified directory. These will be available for selection at the next reboot.

4.2.3 toram

Another useful alternative when booting from live media is `toram`. This is recommended if the computer has enough RAM available (4 GByte or more). `toram` copies the complete content of the live medium into the RAM. The advantage is that the system reacts very fast and you can remove the medium after boot. This is useful if the start was done from a USB stick and you want to use this USB port otherwise.

Last edited: 2022/03/31

4.3   Partitioning of installation media

The partitioning of the drives depends on many factors:

- Selection of the siduction variant

- Size of the available drives and RAM

- Single-boot or dual-boot with an already installed system (Windows, Linux, MAC)

- Sharing of data for the installed systems

For Linux beginners, we recommend that you create only two partitions `/root` (incl. `/home`) and `swap`, as this makes a first installation much easier. After the installation, additional data partitions can be created, or a separate `/home` if desired.

However, we rather advise against creating a `/home` partition. The `/home` directory should be the place where the individual configurations are stored, and only these. For all other private data, a separate data partition should be created. The advantages for data stability, data backup and also in case of data recovery are almost immeasurable.

Purchasing an external USB hard drive for regular data backup is also worth considering.

4.3.1   Minimum requirements

The minimum requirements for the reasonable use of a siduction installation are:

installed system	hard disk space
siduction NoX	5GB
siduction Xorg	10GB
siduction LXQt	15GB
siduction LXDE	15GB
siduction Xfce	15GB

installed system	hard disk space
siduction Cinnamon	15GB
siduction KDE Plasma	15GB

Otherwise, we recommend at least 20 GB of disk space when installing to the **Btrfs** file system and using `snapper`. 50 GB or more is useful if you want to use siduction on **Btrfs** for a longer period of time and many snapshots are kept.

4.3.2   Examples with different disk sizes

There are several good ways to divide your plates. These examples should give a first insight. They refer to partition tables of the type *"GPT"*. The first partition on the first disk is mandatory for the boot process.

Laptop with 8 GB RAM, Linux only
250 GB hard disk

Partition	Size	File system	Use
1	300 MB	FAT32	EFI-System (ESP)
2	40 GB	ext4	/
3	200 GB	ext4	data
4	10 GB	Linux Swap	Linux Swap

Desktop, Linux only
500 GB hard disk:

Partition	Size	File system	Use
1	300 MB	FAT32	EFI system (ESP)
2	40 GB	ext4	/
3	456 GB	ext4	data
4	4 GB	Linux Swap	Linux swap

Desktop, Linux only

500 GB hard disk with Btrfs snapshot:

Partition	Size	File system	Use
1	300 MB	FAT32	EFI system (ESP)
2	496 GB	btrfs	/
3	4 GB	Linux Swap	Linux swap

Desktop PC, Linux allein

500 GB hard disk with Btrfs snapshot and systemd-boot:

Partition	Size	File system	Use
1	300 MB	FAT32	EFI-System (ESP)
2	495 GB	Btrfs	/
3	1 GB	ext4	XBOOLDR
4	4 GB	Linux Swap	Linux Swap

If a dual boot with MS Windows™ is created, MS Windows must always be installed as the first system onto the hard disk. The first four partitions of our examples should be located directly after each other at the beginning of the hard disk. After that, the partitions for Linux and shared data follow.

See also Microsoft: UEFI/GPT partitioning, Windows 11.

Desktop, dual-boot (MS Windows and Linux)

1 TB hard disk:

Partition	Size	File system	Use
1	300 MB	FAT32	EFI-system (ESP)
2	16 MB	without	Windows MSR
3	50 GB	NTFS	Windows system
4	1 GB	NTFS	Windows RE
5	415 GB	exFAT	data for Windows and Linux

Partition	Size	File system	Use
6	30 GB	ext4	/ (Linux root)
7	500 GB	ext4	data for Linux
8	4 GB	Linux Swap	Linux Swap

Laptop with 32 GB RAM, dual boot (MS Windows and Linux)
1 TB hard disk:

Partition	Size	File system	Use
1	300 MB	FAT32	EFI system (ESP)
2	16 MB	without	Windows MSR
3	50 GB	NTFS	Windows system
4	1 GB	NTFS	Windows RE
5	499 GB	exFAT	data for Windows and Linux
6	30 GB	ext4	/ (Linux root)
7	380 GB	ext4	data for Linux
8	40 GB	Linux swap	Linux swap

4.3.3 File systems of the partitions

The type *"GPT"* should be selected as the partition table. In this way the advantages over *"MBR"* can be used. Only with old hardware *"MBR"* is still meaningful. The explanations for this can be found on our manual page Partitioning with gdisk.

Linux Swap

A `swap` partition corresponds in functionality to the swap file in Windows, but is far more effective than it. Its size depends on the installed system and the user's requirements. Some examples:

- For notebooks that are to be hibernated, we recommend a swap partition that is at least one GByte or 25% larger than the RAM.

- Current desktop PCs that are *not* to be hibernated and have enough RAM (16 GByte or more, depending on usage) do not need a swap partition.

- For desktop PCs with very limited RAM, the rule of thumb is still that the swap partition should be twice the size of the RAM used.

ext4

The *ext4* file system is the default file system on siduction. This applies to all partitions when only Linux operating systems are used.

Btrfs

Btrfs can be used instead of *ext4*. Together with the program *Snapper* it offers the possibility to create snapshots of the file system which are selectable in the boot manager Grub afterwards. You need a sufficiently large hard disk. See also System administration Btrfs.

NTFS

For a Windows installation, the partitions intended for this purpose must be formatted with *NTFS*. Siduction has read and write access to the data. It is the standard file system for Windows.

exFAT

A file system developed by Microsoft and used in many types of storage devices such as SD cards and USB flash drives. The patents for it were released in 2019 and as a result Linux supports *exFAT* from kernel 5.4. It is also very suitable for partitions that are to be accessed by different operating systems.

HFS+

For a dual-boot installation with Macintosh, a separate data partition with the *HFS* or *HFS+* file system is useful. Linux and MAC can access it read and write.

4.3.4   Partition editors

> **Caution**
> When using any partitioning software, there is a risk of data loss. Always back up important data to another disk in advance.

- **GParted**: an easy to use partition editor with a graphical interface
  Gparted is available on all siduction installations and installation media
  equipped with a graphical user interface. It supports a number of different par-
  tition table types. The manual page Partitioning the hard disk with GParted
  provides more information about the program.

- **KDE Partition Manager**: a Qt based, easy to use partition editor with a graph-
  ical user interface
  The *KDE Partition Manager* is the standard partition editor for the KDE Destk-
  top and as comprehensive as *Gparted*.

- **gdisk / cgdisk**: a console program for partition tables of the type *GPT - UEFI*
  gdisk is the classic text mode program, while *cgdisk* has a more user friendly
  ncurses interface. The manual page Partitioning with gdisk provides more
  information about the program.

- **fdisk / cfdisk**: a console program for partition tables of the type *msdos - MBR*
  Note: *fdisk* should only be used for old hardware that does not support *GPT
  - UEFI*.
  fdisk is the classic text mode program, while *cfdisk* has a more user-friendly
  ncurses interface. The manual page Partitioning with cfdisk provides more
  information about the program.

Mounted partitions (also swap) must be detached before editing.
You can do this by entering to following command as **root**:

```
# umount /dev/sda1
```

To mount a swap partition, use this command:

```
# swapoff -a
```

4.3.5   Further information

Here the comprehensive english documentation of GParted

Microsoft: UEFI/GPT partitioning, Windows 11

For more partitioning options see:

- Logical Volume Manager LVM partitioning

- partitioning with GPT to support UEFI Partitioning with gdisk

Last edited: 2024-08-29

4.4   UUID - naming of block devices

UUID (Universally Unique Identifier) and partition label

The permanent naming of block devices was made possible with the introduction of udev. The advantage is independence from the used controllers as well as from the type and number of connected devices. The `fstab` file created during the installation of siduction contains corresponding entries for all block devices connected at that time.

4.4.1   Types of block device naming

Currently, Linux uses five types of identifiers for block devices. All identifiers can be found below the `/dev/disk/` directory and are created automatically by the system. For *labels* this only applies if they have been assigned to the block devices beforehand.

1. **UUID**
   This is a unique identifier on file system level and stored in the file system's metadata. To read it, the file system type must be known and readable. It is unique because a new UUID is already created when a partition is formatted. A UUID is a 128-bit number. Anyone can create and use a UUID. The probability that a UUID is duplicated is not zero, but it is so small that the case can be neglected. All Linux file systems including swap support UUID. Although **FAT** and **NTFS** file systems do not support UUID, they are listed in `/dev/disk/by-uuid`.

2. **PARTUUID**.
   This is an identifier on partition table level that has been introduced with GPT. The PARTUUID is preserved when the partition is reformatted and is therefore not unique. For example, mounting through an `fstab` entry based on PARTUUID will fail if the partition was given a different filesystem without modifying `fstab`.

3. **Device ID (ID)**
   The ID is created from the device's metadata (manufacturer, connection type,

construction type, storage volume, etc.) and does neither take into account the partitioning nor the file systems on the partitions. It is unsuitable as a permanent identifier in `fstab`.

4. **PATH**

   This is composed of the controller name, the device type, and the partition number. As with ID, it is unsuitable as a permanent identifier in `fstab`.

5. **LABEL**

   Labels are easily recognizable identifiers assigned by the user. They are not unique, so care must be taken to avoid overlapping names.

By default, siduction uses UUID in /etc/fstab for the reasons named above.

4.4.2   Use label

The label of a block device has the advantage for us humans to be easily understandable and recognizable. Practically every type of file system can have a label. Partitions with a label can be found in the directory `/dev/disk/by-label`:

```
$ ls -l /dev/disk/by-label
total 0
lrwxrwxrwx 1 root root 10 Oct 16 10:27 data -> ../../sdb2
lrwxrwx 1 root root 10 Oct 16 10:27 home -> ../../sda6
lrwxrwx 1 root root 10 Oct 16 10:27 root -> ../../sda1
lrwxrwx 1 root root 10 Oct 16 10:27 swap -> ../../sda5
lrwxrwx 1 root root 10 Oct 16 10:27 windows -> ../../sdb1
```

The label can be created or changed with one the following commands, according to the respective file system:

- **swap**
  `swaplabel -L <label> /dev/sdXx`

- **ext2/ext3/ext4**
  `e2label /dev/sdXx <label>` or
  `tune2fs -L <label> /dev/sdXx`

- **JFS**

  `jfs_tune -L <label> /dev/sdXx`

- **XFS**

  `xfs_admin -L <label> /dev/sdXx`

- **ReiserFS**

  `reiserfstune -l <label> /dev/sdXx`

- **FAT**

  `fatlabel /dev/sdXx <label>`

- **NTFS**

  `ntfslabel /dev/sdXx <label>`

- **exFAT** `exfatlabel /dev/sdXx <label>`

An NTFS and FAT partition's label should consist only of uppercase letters, digits, and special characters that Windows™ allows for file names.

The syntax in `fstab` for the `<file system>` is `LABEL=\<label\>`.

It is essential to note:

The labels must have a singular name in order to work when mounted. This also applies to external devices (hard disks, sticks, etc.) that are mounted via USB or Firewire.

4.5   The fstab

The file `/etc/fstab` is read during system startup to mount the desired partitions. Here is an example of an fstab:

```
UUID=2e3a21ef-b98b-4d53-af62-cbf9666c1256 swap swap defaults,⤶
    noatime 0 2
UUID=1c257cff-1c96-4c4f-811f-46a87bcf6abb / ext4 defaults,⤶
    noatime 0 1
UUID=35336532-0cc8-4613-9b1a-f31b12ea58c3 /home ext4 defaults⤶
    ,noatime 0 2
tmpfs /tmp tmpfs defaults,noatime,mode=1777 0 0
```

```
UUID=e2164479-3f71-4216-a4d4-af3321750322 /mnt/TEST_root ext4↙
    noauto,noatime 0 0
LABEL=TEST_HOME /mnt/TEST_home ext4 noauto,users,noatime 0 0
UUID=B248-1CCA /mnt/TEST_boot vfat noauto,users,rw,noatime 0 ↙
    0
UUID=a7aeabe9-f09d-43b5-bb12-878b4c3d98c5 /mnt/TEST_res ext4 ↙
    noauto,users,rw,noatime 0 0
```

Partitions listed in `fstab` can be mounted with their <file system> identifier or with the <mount point>.

```
$ mount UUID=a7aeabe9-f09d-43b5-bb12-878b4c3d98c5
    or
$ mount /mnt/TEST_res
    or
$ mount LABEL=TEST_HOME
```

4.5.1   Adjusting the fstab

If you want the ability to use newly created partitions (let's take sda5 and sdb7 as examples) that do not appear in `fstab` or cannot be mounted with the previously mentioned commands, type the following command into the console as **user**:

```
$ ls -l /dev/disk/by-uuid
```

It will print something similar to this:

```
lrwxrwx 1 root root 10 May 29 17:51 1c257cff-1c96-4c4f-811f↙
    -46a87bcf6abb -> ../../sda2
lrwxrwxrwx 1 root root 10 May 29 17:51 2e3a21ef-b98b-4d53-↙
    af62-cbf9666c1256 -> ../../sda1
lrwxrwxrwx 1 root root 10 May 29 17:51 2ef32215-d545-4e12-↙
    bc00-d0099a218970 -> ../../sda5
lrwxrwxrwx 1 root root 10 May 29 17:51 35336532-0cc8-4613-9↙
    b1a-f31b12ea58c3 -> ../../sda4
```

```
lrwxrwxrwx 1 root root 10 May 29 17:51 4c4b9246-2904-40d1-↙
    addc-724fc90a2b6a -> ../../sdb3
lrwxrwxrwx 1 root root 10 May 29 17:51 a7aeabe9-f09d-43b5-↙
    bb12-878b4c3d98c5 -> ../../sdb7
lrwxrwxrwx 1 root root 10 May 29 17:51 B248-1CCA -> ../../↙
    sdb1
lrwxrwxrwx 1 root root 10 May 29 17:51 d5b01bbc-700c-43ce-↙
    a382-1ba95a59de78 -> ../../sdb6
lrwxrwxrwx 1 root root 10 May 29 17:51 e2164479-3f71-4216-↙
    a4d4-af3321750322 -> ../../sdb5
lrwxrwx 1 root root 10 May 29 17:51 f5ed412d-7b7b-41c1-80ce↙
    -53337c82405b -> ../../sdb2
```

In this example,
"2ef32215-d545-4e12-bc00-d0099a218970" is the missing entry for sda5 and
"a7aeabe9-f09d-43b5-bb12-878b4c3d98c5" is the missing entry for sdb7.

The next step is to add the UUID partitions to `/etc/fstab`. To achieve this, use a text editor (like `mcedit`, `kate`, `kwrite`, or `gedit`) with **root** privileges. In this example, the entry would look like this:

```
UUID=2ef32215-d545-4e12-bc00-d0099a218970 /media/disk1part5 ↙
    ext4 auto,users,exec 0 2
UUID=a7aeabe9-f09d-43b5-bb12-878b4c3d98c5 /media/disk2part7 ↙
    ext4 auto,users,exec 0 2
```

We then inform the kernel about the changes to `/etc/fstab` with the command **`systemctl daemon-reload`**.

4.5.2 Creation of new mount points

Note: A mount point that is specified in fstab must be associated with an existing directory. During the live session, siduction creates these directories in `/media` with the naming scheme **diskXpartX**.

Now, if the partition table was changed after the installation and `fstab` was adjusted (for example, two new partitions were created), no mount point exists yet. It must be created manually.

Example

First, become **root** and determine the existing mount points:

```
cd /media
ls
```

The output shows for example:

```
disk1part1 disk1part3 disk2part1
```

The mount points of the new partitions are now created in the `/media` directory:

```
mkdir disk1part5
mkdir disk2part7
```

Thus, the new partitions can be used or tested immediately:

```
mount /media/disk1part5
mount /media/disk2part7
```

After a reboot, the new file systems are mounted automatically if *auto* or *defaults* is entered in the `fstab` under *"<options>"*. See also:

```
man mount
```

Of course, you don't have to follow the naming scheme *"diskXpartX"*. Mount points and their associated identifiers in `fstab` can be assigned meaningful names, for example, *"data"* or *"music"*.

Last edited: 2023/11/07

4.6   Partitioning with GParted

Creating or editing partitions is not an everyday task. Therefore, it is a good idea to read the following guide at least once to get familiar with the concept of a partition manager.

4.6.1   Important notes

- Always create a data backup first!

- Regarding the naming of storage devices, consult the chapter about UUID, partition naming and fstab because siduction uses naming by UUID by default.

- Resizing **NTFS partitions** requires an immediate reboot after execution. No further changes to partitions may be made before that since this inevitably will lead to errors. Please read on here.

- A partition needs a file system. Linux can work on and with different file systems.
For normal use, we recommend the **ext4** file system.
NTFS should be used if the partition is also to be used by a Windows installation. siduction can read and write data to such partitions through the automatically installed `ntfs-3g`.

- The complete GParted documentation can be found in many languages on the GParted homepage.

4.6.2   Using GParted

The program launcher for GParted can be found in

- **KDE, LXQt, Xfce**
in the application menu - *"System" - "GParted"*

- **Gnome**
in *"Applications" - "Gparted"*

After clicking on the launcher, a dialog will open and ask for the root password.

When GParted starts, the program window opens and the available drives are read.
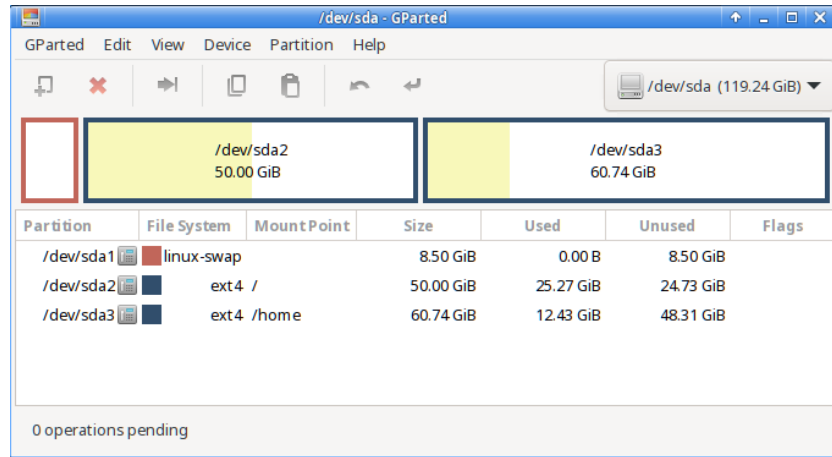
The first menu item *"GParted"* opens a drop-down list which allows you to read the drives again, to select a drive or to quit the program.
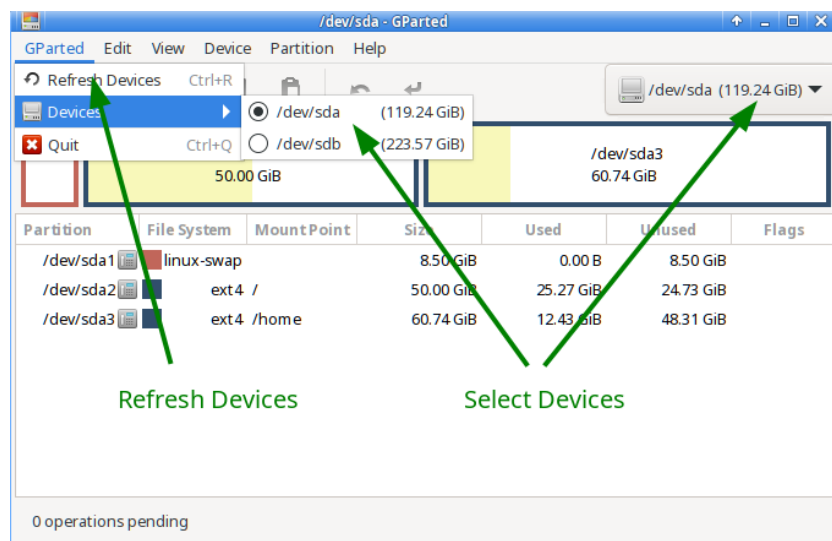
- **Edit**

    Edit is the 2nd menu item from the left. It shows three grayed out options that are very important and explained below:

    - *"Undo last operations"*,

- *"Clear all operations"*, and
- *"Apply all operations"*.

- **View**

  The next menu item offers the display options `Device Information` and `Pending Operations`.

  - Device Information
    The left frame contains details of the drives such as model, size, etc., which are important if there are multiple disks in the system. It can be used to check whether the correct disk has been selected for formatting.

  - Pending Operations
    The pending operations are displayed in a frame that opens at the bottom. This information is very useful to have an overview of which operations are to be performed. The frame also opens automatically when an operation is requested for a drive.

  The two areas are highlighted in green.
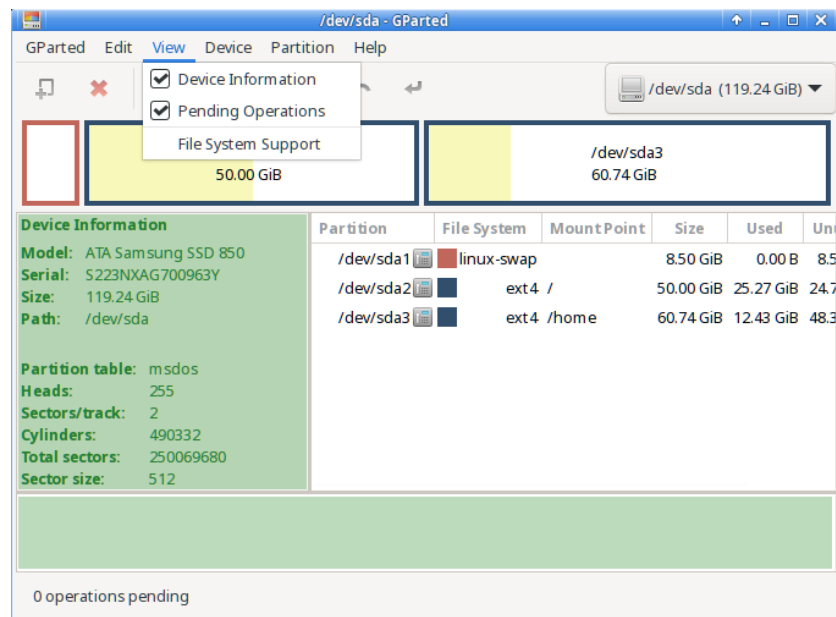
- **Device**

The menu item `Create Partition Table...` actually contains two options:

1. Create a new (empty) partition table of the same type, and thus remove all old partitions and data in the fastest way.

2. Change the partition table type. It makes sense to change from **msdos-MBR** to **gpt-UEFI** or vice versa. Here, too, all data will be lost.
   In 2009, UEFI with GPT was introduced, has been gradually spreading ever since, and will replace the MBR. While modern UEFI mainboards support MBR, the benefits of GPT are lost. More information about UEFI and GPT can be found on the manual page Partitioning with gdisk.
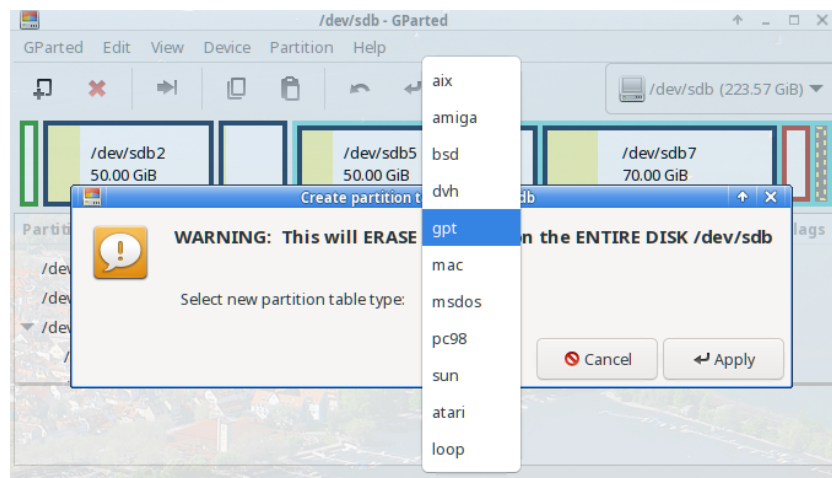
The option `Attempt Data Rescue...` offers the chance to get the data despite a defective partition table.
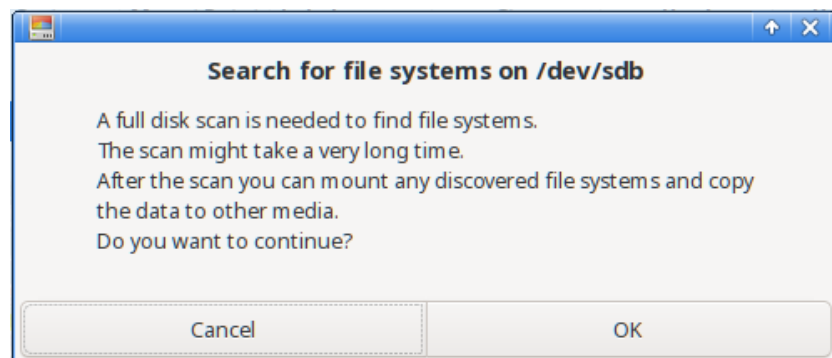
- **Partition**

The menu item `Partition` is of utmost importance. For the partition selected below, the menu shows all available operations depending on whether the partition is mounted or unmounted. Note that some of the sub-items can also perform critical or dangerous actions.

- **Create a new partition**

  In the toolbar, the `New` button allows you to create a new partition if an unallocated area was previously selected. The appearing window lets you specify the size and the file system type for a primary, extended, or logical partition.
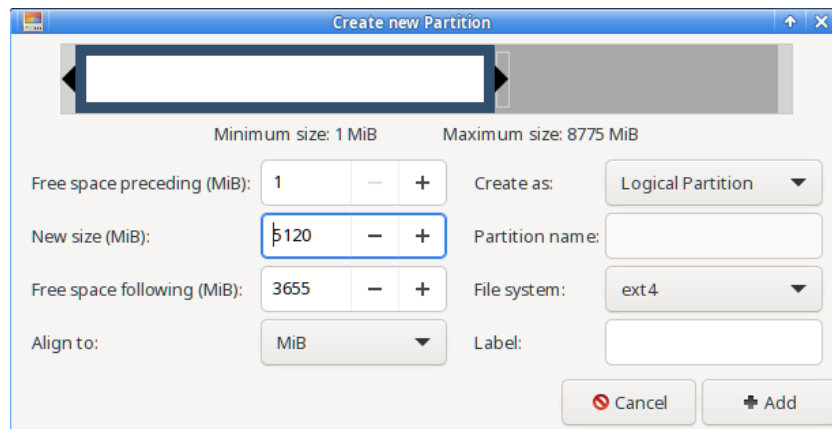
- **Resize/Move**

  The partition can be resized, enlarged, and moved with the mouse. Alternatively, enter the new values into the provided fields.
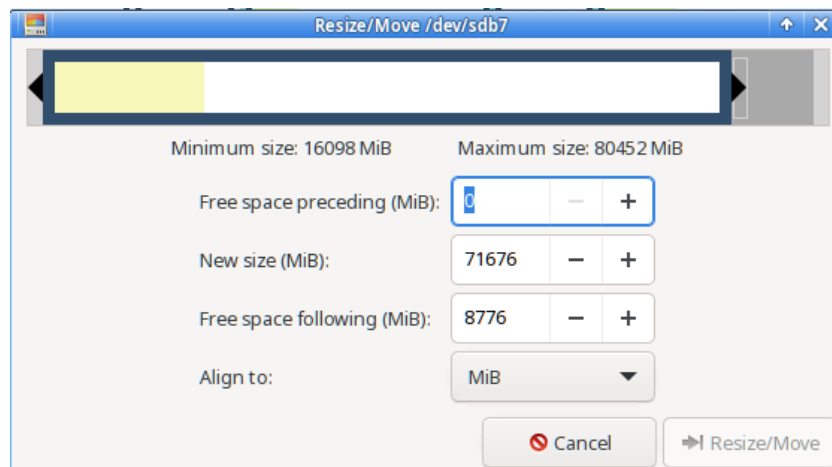
- **If a mistake has been made**

  In the `Edit` menu, you can find the options
  `Undo Last Operation` and
  `Delete All Operations`. The area is highlighted in green.
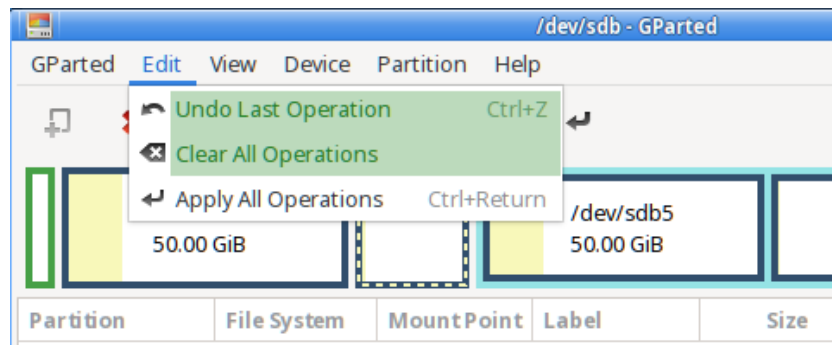
- **Apply**

No changes have been made to the drives yet. If you are sure that all the intended changes are correct, select `Apply All Operations` in the `Edit` menu. The following dialog will appear, which should be confirmed.
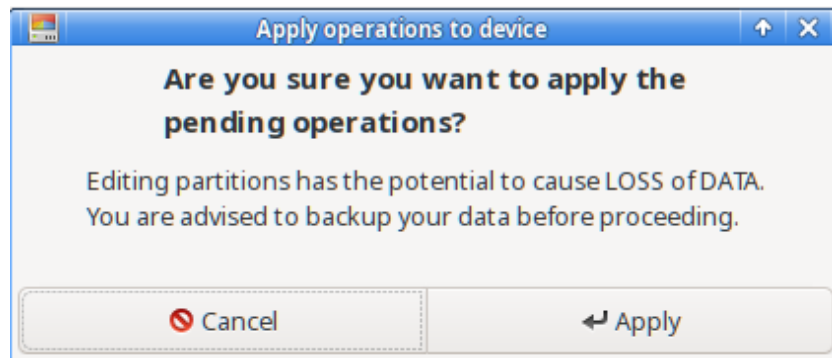
The duration of the operation depends on the size of the selected partition.

4.6.3   Adjust fstab

> After the changes have been written to the drives, the /etc/fstab file must be checked and adjusted if necessary.

See the manual page Adjusting fstab.
In a **root** terminal, enter the commands `cat /etc/fstab` as well as `blkid` and compare the UUIDs.

```
root@pc1:/# cat /etc/fstab
```

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for ↲
    a device; this may
# be used with UUID= as a more robust way to name devices ↲
    that works even if
# disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump><pass>
UUID=2e3a21ef-b98b-4d53-af62-cbf9666c1256 swap swap defaults,↲
    noatime 0 2
UUID=1c257cff-1c96-4c4f-811f-46a87bcf6abb / ext4 defaults,↲
    noatime 0 1
UUID=35336532-0cc8-4613-9b1a-f31b12ea58c3 /home ext4 defaults↲
    ,noatime 0 2
tmpfs /tmp tmpfs defaults,noatime,mode=1777 0 0
UUID=f5ed412d-7b7b-41c1-80ce-53337c82405b /mnt/photo ext4 ↲
    defaults,noatime 0 0
UUID=4c4b9246-2904-40d1-addc-724fc90a2b6a /mnt/Backup ext4 ↲
    noauto,users,noatime 0 0
UUID=a7aeabe9-f09d-43b5-bb12-878b4c3d98c5 /mnt/TEST_res ext4 ↲
    noauto,users,rw,noatime 0 0
```

```
root@pc1:/# blkid
/dev/sda1: UUID="2e3a21ef-b98b-4d53-af62-cbf9666c1256" TYPE="↲
    swap" PARTUUID="000403b7-01"
/dev/sda2: UUID="1c257cff-1c96-4c4f-811f-46a87bcf6abb" ↲
    BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="000403b7-02"
/dev/sda3: UUID="35336532-0cc8-4613-9b1a-f31b12ea58c3" ↲
    BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="000403b7-03"
/dev/sdb1: UUID="f5ed412d-7b7b-41c1-80ce-53337c82405b" ↲
    BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="2853e345-01"
/dev/sdb2: UUID="4c4b9246-2904-40d1-addc-724fc90a2b6a" ↲
    BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="2853e345-02"
```

```
/dev/sdb5: UUID="e2164479-3f71-4216-a4d4-af3321750322" ⤸
    BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="2853e345-05"
/dev/sdb6: UUID="2ef32215-d545-4e12-bc00-d0099a218970" ⤸
    BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="2853e345-06"
```

We can see that the last entry in the `fstab` (mounted to `/mnt/TEST_res`) is no longer contained in the `blkid` list. Instead, we have two new partitions. In this example, the PC would perform a reboot but would not be able to mount `/mnt⤸ /TEST_res` and the two new partitions automatically. The boot process would be delayed considerably.

> If the UUID's for the partitions of **/** (root), **/home**, and **swap** do not match the entries in **/etc/fstab**, it is mandatory to adjust the entries. Otherwise, the system will not start after a reboot.

4.6.4   Changing NTFS partition sizes with GParted

Resizing NTFS partitions requires an immediate reboot after execution. No further changes to partitions may be made before then. This will inevitably lead to errors.

- Once Windows has started and the Windows logo has disappeared, a **check-disk** window appears which says that **C:** is being checked for errors.
- Please let this AUTOCHECK finish its job: Windows must check the file system after a resize.
- After the check, the computer is automatically rebooted the second time. This ensures that the system can run without problems.
- After the restart, Windows will work properly. However, you have to let the system finish booting and wait for the login window!

Last edited: 2022/03/31

4.7   Partitioning with gdisk

Why use gdisk (GPT fdisk)?

gdisk is derived from **G**lobally Unique Identifier **P**artition **T**able (GPT) and is an application for partitioning disks of any size. gdisk is absolutely required for **disks larger than 2TB**.
It ensures that partitions are set up for SSDs (or for storage that does not have 512 Byte sectors).

A key advantage of GPT is that you no longer have to rely on the MBR's inherent primary, extended, or logical partitions. GPT can support an almost unlimited number of partitions and is limited only by the amount of space reserved for partition entries on the GPT volume. It should be noted that the gdisk application defaults to 128 partitions.

If GPT is used on small USB/SSD drives (for example on a USB drive with 8GB), this could have a counterproductive effect in case data is to be exchanged between different computers or operating systems.
For this purpose, and if older hardware is used, you should use *fdisk*, which creates partition tables based on the MBR. See the manual page Partitioning with Cfdisk.

Important notes

- The terms UEFI and EFI are interchangeable and refer to the same concept - **U**nified **E**xtensible **F**irmware **I**nterface.
  See Wikipedia UEFI.
  GPT is a part of the UEFI standard.

- Using GPT disks

  – GPT disks can be used on Linux systems with 32 bit and 64 bit.

  – Some operating systems do not support GPT disks.
    This includes all MS operating systems before Windows Vista SP1.
    Please consult the documentation of the respective system.

- Booting from GPT media

- Dual and triple boot from GPT media with Linux, BSD, and Apple is supported with 64-bit **EFI** mode.

- Dual boot of GPT volumes with Linux and MS Windows is possible since Windows Vista SP1. Prerequisite for Windows is the 64-bit version.

- Graphical partition editors for GPT

  Besides the command line program `gdisk`, graphical applications like `gparted` and KDE's `partitionmanager` support GPT disks. Nevertheless, we recommend gdisk to prevent unwanted anomalies. `gparted` as well as `partitionmanager` (and others) are, however, great tools, especially to visualize partitioning.

Essential reading matter:

- man gdisk
- GPT fdisk Tutorial by Roderick W. Smith
- Wikipedia UEFI operating system support
- Wikipedia GUID partition table

4.7.1 Partitioning a hard disk

> **Back up data beforehand!**
> When using any partitioning software, there is a risk of data loss. Data you want to preserve should always be backed up beforehand on another data medium.

In the following example, we will format a 150 GB hard disk so that two Linux systems can be installed as dual boot afterwards. In order to benefit from UEFI's advantages, we need an *EFI system* partition in the GPT.

We show the necessary steps with the partitioning program `cgdisk`, which supports GPT with UEFI.

4.7.2 Use cgdisk

cgdisk can only be used with non-mounted disks. For example, you can use a siduction live medium to edit the only available hard disk, or you can use cgdisk from the running system to partition an additional hard disk or USB stick.

The boot command in a root terminal is: `cgdisk /dev/sdX`.
`cgdisk` is the curses-based program variant of `gdisk`. It provides a user-friendly interface within the terminal.
Navigation is done using the arrow keys:

- **up** and **down** for the partitions
- **right** and **left** to select an action
- **Enter** to confirm the selection or input
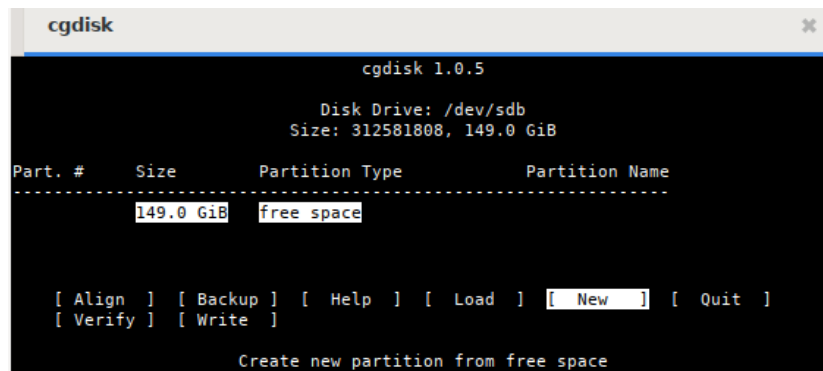
`cgdisk` starts with a warning message if no GPT is found.
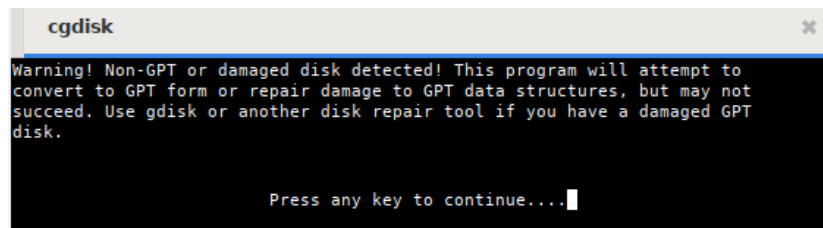
We need a total of five partitions for the two operating systems: two ROOT partitions, one shared DATA partition, and one SWAP partition for swap space. In addition, the *EFI system* partition already mentioned above (maximum 100MB) are required.

We recommend leaving the `/home` directory on the ROOT partition. The `/home`⤸ directory should be the place where the individual configurations are stored, and only these. A separate data partition should be created for all other private data. The advantages for data stability, data backup, and also in case of data recovery are almost immeasurable.

The start screen:
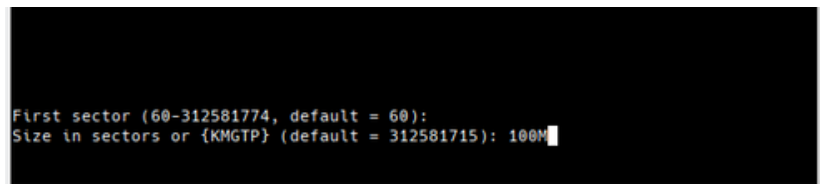
Create partition

We select *"New"* and confirm with **Enter**. Hitting **Enter** a second time, we accept the default first sector for the new partition. Then we enter the desired size of `100`↙ `M` for the *EFI-System* partition and confirm.
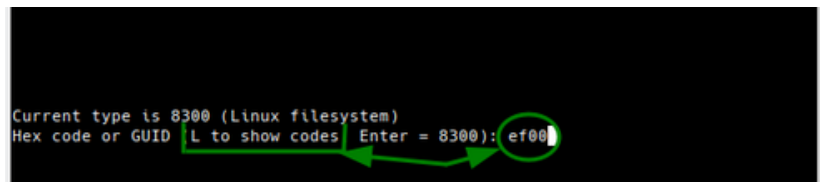
Now we are expected to enter the type code for the partition.

After entering **L**, a long list of codes and their usage appears. The integrated search function simplifies the selection. For us, the following codes are necessary:

"ef00" for EFI system

"8200" for Swap

"8304" for Linux root

"8300" for Linux data

So we enter `ef00` and confirm. Afterwards, we may optionally assign a name (label), which has been done in the example, and confirm the entry again. We proceed after the same pattern for the partitions Linux-root, and Swap. The next picture shows the result of our efforts. As we can see, there is still plenty of space for a second system and especially for a shared data partition.

After the two partitions have been created, we can see the partitioning of the entire disk in the next image.

The partitions that the two systems will use later during operation are color-coded. At the beginning and the end, there are still small, free areas. They are created by aligning the partition to the block boundaries of the disk and can also appear between the partitions. With *"Align"* the value for the number of sectors can be changed. It is usually 2048 sectors for SSD and M2 disks and 512 sectors for old disks. `gdisk` reads the metadata of the hard disks and sets the value for the sectors afterwards. Therefore usually no change is necessary.

Additional, detailed information about the partitions can be seen by entering the command *"Info"*.

With *"Verify"* the partitioning is checked and possible errors are shown.

Here, everything is ok.

If errors are reported, we mark the partition and use the command *"Info"*. Then we decide if the partition has to be deleted and recreated and if e.g. the size has to be changed as well. If a repair is not possible by these means, the Advanced commands of gdisk are available for experienced users.

Delete partition

To delete a partition, we select it and use the command *"Delete"*.

If necessary, we do the same with other partitions and then we can create the partitions again with changed values.

Write GPT

If the partitioning of the hard disk corresponds to our ideas, we check once more whether everyting is in order with the command *"Verify"*. If no errors are displayed, we choose *Write* and

```
[ Align ]  [ Backup ]  [ Help ]  [ Load ]  [ New  ]  [ Quit ]
[ Verify ]  [ Write ]
          Write partition table to disk (this might destroy data)
```
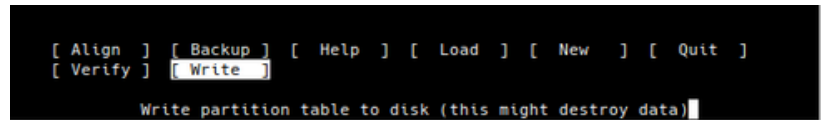
may answer the security query with "yes".

```
Are you sure you want to write the partition table to disk? (yes or no):
          Warning!! This may destroy data on your disk!
```
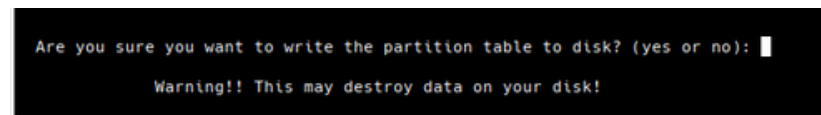
The warning should be taken seriously because after pressing the **Enter** key, all data that was previously on the disk disappears into nirvana.

Since `cgdisk` only creates partitions but no file systems, each of the new partitions must be formatted. With The *"Quit"* command terminates `cgdisk`.

4.7.3   Formatting the partitions

We stay in the **root** terminal and display the paths with the numbers for each partition:

```
fdisk -l | grep /dev/sdb
```

The command generates the following output:

```
Disk /dev/sdb: 149.5 GiB, 160041885696 bytes, 312581808 ⤴
   sectors
/dev/sdb1      2048     206847     204800 100M EFI System
/dev/sdb2    206848   52635647   52428800  25G Linux root
/dev/sdb3  52635648   61024255    8388608   4G Linux swap
/dev/sdb4  61024256  260270079  199245824  95G Linux filesyst
/dev/sdb5 260270080  312581807   52311728  25G Linux root
```

With this information, we format our previously created partitions.

Please read **man mke2fs**, **man mkfs.fat**, and **man mkswap**.

The EFI system partition will be given a **FAT32** file system.

```
mkfs.vfat /dev/sdb1
```

The EFI partition must be the first partition of the hard disk and must be mounted under `/boot/efi/` to comply with the siduction standard.
If the boot manager GRUB finds such a prepared *EFI-System* partition during the installation, it will use it.

We format the Linux partitions sdb2, sdb4, and sdb5 with **ext4**.

```
mkfs.ext4 /dev/sdb2
```

To set up the swap partition, format it with

```
mkswap /dev/sdb3
```

Now the system needs to know about this partition:

```
swapon /dev/sdb3
```

Check if the swap space is available:

```
swapon -s
Filename    Type       Size      Used  Priority
/dev/sdb3   partition  4194304   0     -2
```

If swap was detected correctly:

```
swapoff /dev/sdb3
```

We then inform the kernel about the changes with the command **systemctl ⤸ daemon-reload**.

4.7.4   Booting with GPT-UEFI or GPT-BIOS

If a bootable volume is to be created with GPT, there are two ways to create the boot sector of a GPT volume.

These possibilities are:

- The computer (the mainboard) has a UEFI and UEFI shall be used to boot the GPT medium. An *"EFI System"* partition formatted with FAT32 (gdisk type "EF00") is required as the first partition on the medium.

or

- The computer (mainboard) has **no** UEFI but a BIOS. The BIOS should boot the GPT medium. (All mainboards before 2009 do not have UEFI.)

Booting with UEFI and the boot loader GRUB

If UEFI is to be used for booting, an *"EFI System"* partition (type "EF00") formatted with FAT32 must be created as the first partition and mounted under `/boot/efi`↙
. This partition contains the boot loader(s).
The boot loader of siduction is stored in the directory `/boot/efi/EFI/`↙
`siduction/`.

Booting with UEFI and the boot loader systemd-boot

For systemd-boot, an additional *XBOOTLDR* partition of at least 1 GB (gdisk type "EA00") should be created somewhere on the same medium.
The *"EFI System"* partition must be mounted for systemd-boot under `/efi`
and the *XBOOTLDR* partition under `/boot`. See also the systemd-boot manual page.

Booting with BIOS

If the system does not have a UEFI, the first partition must be a *"BIOS boot"* partition (type "ef02"). Grub is written directly to this partition. It replaces the sector of an MBR-partitioned disk that is located between the partitioning table and the first partition.
The partition should have the size of 200MB. The reason for this size instead of

the conventional 32MB is to have a sufficiently large partition available in case of a switch to UEFI.

4.7.5 Advanced commands of gdisk

`gdisk` has advanced options and security mechanisms not available in `cgdisk`.

If problems are detected (e.g. overlapping partitions or non-matching main and backup partition tables), it is possible to fix them with various options in the **recovery & transformation** menu. We start `gdisk` with

```
gdisk /dev/sdb
```

At the command prompt **Command (? for help):**, we enter the command **r** to get into the submenu of *"recovery & transformation"* and then **?**.

```
recovery/transformation command (? for help): ?
b use backup GPT header (rebuilding main)
c load backup partition table from disk (rebuilding main)
d use main GPT header (rebuilding backup)
e load main partition table from disk (rebuilding backup)
f load MBR and build fresh GPT from it
g convert GPT into MBR and exit
h make hybrid MBR
i show detailed information on a partition
l load partition data from a backup file
m return to main menu
o print protective MBR data
p print the partition table
q quit without saving changes
t transform BSD disklabel partition
v verify disk
w write table to disk and exit
x extra functionality (experts only)
? print this menu
```

A third menu, *"experts"* , is reached with **x** from either the *"main menu"* or the *"recovery & transformation menu"*.

```
recovery/transformation command (? for help): x

Expert command (? for help): ?
a set attributes
c change partition GUID
d display the sector alignment value
e relocate backup data structures to the end of the disk
g change disk GUID
i show detailed information on a partition
l set the sector alignment value
m return to main menu
n create a new protective MBR
o print protective MBR data
p print the partition table
q quit without saving changes
r recovery and transformation options (experts only)
s resize partition table
v verify disk
w write table to disk and exit
z zap (destroy) GPT data structures and exit
? print this menu
```

This menu allows low-level editing such as changing the partition GUID or the disk GUIDs (**c** or **g**). The **z** option instantly destroys the GPT data structures. This can be useful if the GPT volume is to be used with a different partitioning scheme. If these structures are not erased, some partition editors may have problems because of the presence of two partitioning schemes.

Despite all this: the options of the menus *"recovery & transformation"* and *"experts"* should only be used if you are very familiar with GPT. As a "non-expert", you should only use these menus if a disk is damaged. Before any drastic action, the option **b** in the main menu should be used to create a backup copy in a file and save it on

a separate medium. This will allow the original configuration to be restored if the action does not go as desired.

Last edited: 2024/08/30

4.8   Partitioning with fdisk

fdisk and **cfdisk** create MBR partition tables based on the BIOS. The introduction of GPT partition tables based on UEFI began in 2000.

The newer **G**lobally Unique Identifier **P**artition **T**able (GPT) standard, which is part of the UEFI standard, has replaced MBR on current hardware and allows disks/-partitions larger than 2 TBytes and a theoretically unlimited number of primary partitions. More information about this can be found in Wikipedia GUID partition table.

We recommend to use *fdisk* and *cfdisk* only for partitioning on older hardware. For creating GPT partition tables, please refer to the manual page Partitioning with gdisk.

4.8.1   Naming storage devices

Please NOTE:
siduction uses UUID in `fstab` for storage device naming. Please refer to the chapter Naming by UUID.

Disks

Information about the devices can be easily obtained from an information window (pop-up) by hovering the mouse over the icon of a device on the desktop. This works both from the live ISO and with siduction installed.

We recommend creating a table (manual or generated) that contains the details of all devices. This can be very helpful if problems arise. In a terminal, we become **root** with **su** and type **fdisk -l**. For example, with two disks, we get output similar to that shown below.

```
user1@pc1:/$ su
password:
root@pc1:/# fdisk -l


Disk /dev/sda: 149.5 GiB, 160041885696 bytes, 312581808 ↙
    sectors
```

```
Disk model: FUJITSU MHY2160B
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x6513a8ff

Device    Boot    Start        End    Sectors Size Id Type
/dev/sda1          2048   41945087   41943040  20G 83 Linux
/dev/sda2      41945088   83888127   41943040  20G 83 Linux
/dev/sda3      83888128   88291327    4403200 2,1G 82 Linux swap
/dev/sda4      88291328 312581807  224290480 107G  5 Extended
/dev/sda5      88293376 249774079  161480704  77G 83 Linux
/dev/sda6     249776128 281233407   31457280  15G 83 Linux
/dev/sda7     281235456 312581807   31346352  15G 83 Linux


Disk /dev/sdb: 119,25 GiB, 128035676160 bytes, 250069680 ↙
    sectors
Disk model: Samsung SSD 850
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x000403b7

Device    Boot   Start        End    Sectors  Size Id Type
/dev/sdb1         2048   17831935   17829888  8.5G 82 Linux swap
/dev/sdb2     17831936 122687487  104855552   50G 83 Linux
/dev/sdb3    122687488 250068991  127381504 60,8G 83 Linux
```

By entering the command

fdisk -l > /home/<MY USER NAME>/documents/fdisk-l_output we get a text file with the same content.

Partitions

The partitions on an MBR hard disk are defined by a number between 1 and 15. A maximum of 14 mountable partitions is possible.

The following partition types exist:
primary, extended, and logical.

The logical partitions are located within the extended partition. A maximum of four primary or three primary and one extended partition can be created. The extended partition, in turn, can contain up to eleven logical partitions.
Primary or extended partitions are given a designator between 1 and 4 (for example sda1 to sda4). Logical partitions are always bundled and part of an extended partition. A maximum of eleven logical partitions can be defined with *libata*, and their names start with number 5 and end with number 15 at most.

Examples

```
4 partitions (all primary):


|sda1|sda2|sda3|sda4|



6 partitions (3 primary, 1 extended, and 3 logical):


|sda1|sda2|sda3|-
                 |        contains only
              |sda4|  -> references to
                 |        logical partitions
                 |
              |sda5|sda6|sda7|
```

/dev/sda5 can only be a logical partition (in this case the first logical one on this device). It is located on the first hard disk of the computer (depending on the BIOS configuration).

/dev/sdb3 can only be a primary or extended partition. The letter "b" indicates that this partition is on a different device than the partition of the first example, which contains the letter "a".

4.8.2   Use cfdisk

Backup data beforehand!

> There is a risk of data loss when using any partition editor. Always back up data you want to keep on another disk first.

cfdisk is started in a console as **root** (after **su** the root password is required):

```
user1@pc1:/$ su
password:
root@pc1:/#
cfdisk /dev/sda
```

cfdisk should only be used on a hard disk with all partitions unmounted. All data will be lost when the changed partition table is written.

The user interface

On the first screen, `cfdisk` shows the current partition table with the names and some information about each partition. At the bottom of the window, there are some command buttons. To switch between partitions, use the arrow keys **up** and **down**. To select commands, use the arrow keys **right** and **left**. The **Enter** key is used to execute the command.
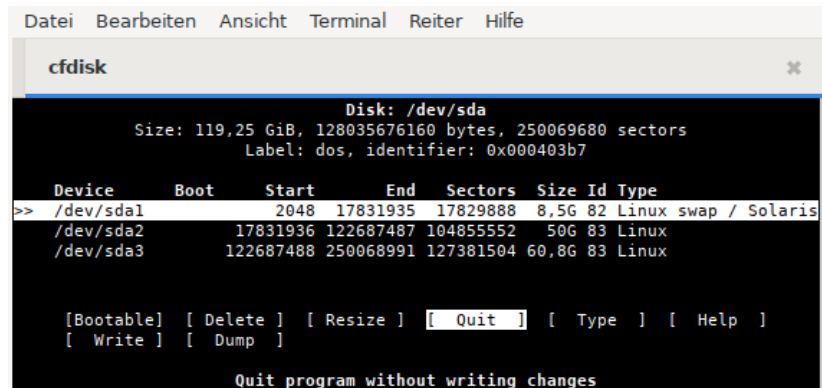
We have three partitions on the example disk.
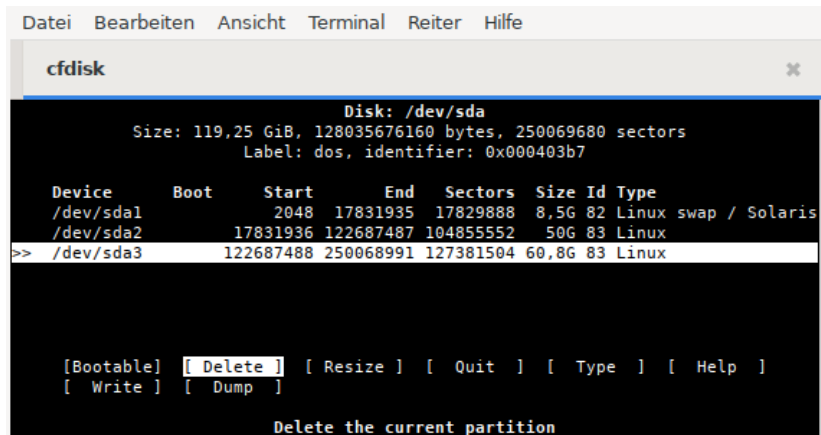
Device	Part. Size	Part. Type	Mountpoint
/dev/sda1	8.5G	82 Swap	-
/dev/sda2	50.0G	83 Linux	/
/dev/sda3	60.8G	83 Linux	/data

From the data partition, we want to move the directories `Pictures` and `Music`↗ to their own partitions and create more space for them. At the same time, they should be accessible for a Windows system residing on another harddisk. The root partition is oversized with 50 GB and will be reduced.

Delete a partition

To create space, we delete the data partition and then shrink the root partition.

To delete the partition **/dev/sda3**, highlight it with the up-down keys and select the command *"Delete"* with the left-right arrow keys. Finally, confirm the action by hitting `Enter`.

Resize a partition

Highlight the partition **/dev/sda2**, select the command *"Resize"*, and confirm.

Then the new size of *"20G"* is to be entered.

Creating a new partition

The hard disk's freed space is highlighted. The command selection automatically jumps to *"New"*, which has to be confirmed.
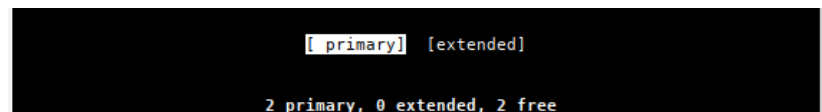
Then enter the new size of *"15G"* for the data partition.

Now we have to choose between a **primary** or an **extended** partition. We select a primary partition.

After that, we mark the free disk space again, confirm it, and confirm the preset total size as well. In the following selection, **extended** has to be chosen. This creates the extended partition (here called *"container"*) in which the two additional partitions are to be created.

Finally, the partitions for `Music` and `Images` are to be created in the desired size according to the procedure shown above. Since only logical partitions are possible, the selection between primary and extended partition is omitted.

This is how the result looks like.

Partition type

To change the type of a partition, select the desired partition and choose the com-
mand *"Type"*.

A selection list appears in which the partition type can be selected with the arrow
keys **up** and **down**. In our example, we select "*7 HPFS/NTFS/exFAT*" for the parti-
tions **/dev/sda5** and **/dev/sda6**. This way, the above mentioned Windows system
can access the partition.
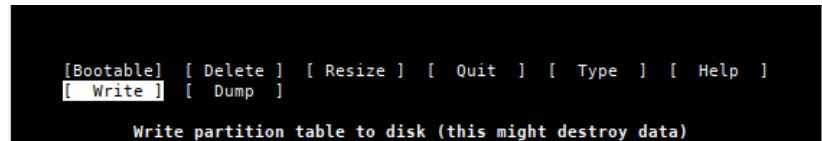
Make a partition bootable

For Linux there is no need to make a partition bootable, but some other operating
systems need it. This is done by highlighting the appropriate partition and select-
ing the *"Bootable"* command. (**Note:** when installing to an external hard drive, a
partition must be made bootable.)

Write partition table

When everything has been partitioned, the result can be saved with the command *"Write"*. The partition table is now written to the disk.

```
[Bootable]  [ Delete ]  [ Resize ]  [  Quit  ]  [  Type  ]  [  Help  ]
[  Write ]  [  Dump  ]
          Write partition table to disk (this might destroy data)
```

Since this will delete all data on the corresponding disk/partition, you should be really sure before typing **yes** and confirming again with the `Enter` key.

Quit cfdisk

By entering the command *"Quit"*, we can quit the program. After leaving `cfdisk` and before the installation, you should reboot in any case to read in the partition table again.

4.8.3 Formatting partitions

There are several file systems for Linux that can be used. There are **Ext2**, **Ext4**, **ReiserFs**, and for more experienced users **XFS**, **JFS**, and **ZFS**.
Ext2 may be of interest when accessing from Windows, as there are Windows drivers for this file system. Ext2 file system for MS Windows (drivers and documentation).

For normal use, we recommend the **ext4** file system. It is siduction's default file system.

After `cfdisk` has finished, the **root** console is still needed as formatting requires root privileges.
The command is `mkfs.ext4 /dev/sdaX`. For *"X"*, enter the number of the selected partition.

```
mkfs.ext4 /dev/sda2
mke2fs 1.45.6 (20-Mar-2020)
/dev/sdb2 contains an ext4 file system
   last mounted on Tue May 26 14:26:34 2020
```

```
Proceed anyway? (y,N)
```

The query is to be answered with **y** if you are sure that the correct partition should be formatted. Please check several times!

After the formatting is finished, you should get a message that ext4 was written successfully. If this is not the case, something went wrong during partitioning or **sdaX** is not a Linux partition. We check with:

```
fdisk -l /dev/sda
```

If something is wrong, you may have to partition again.

If the formatting was successful, this procedure can be repeated for the other partitions, adapting the command according to the partition type and the desired file system (e.g.: **mkfs.ext2** or **mkfs.vfat** or **mkfs.ntfs**, etc.). Please read the man page **man mkfs**.

Finally, format the swap partition, in this case sda1:

```
mkswap /dev/sda1
```

Next, the swap partition is activated:

```
swapon /dev/sda1
```

After that, you can check in the console if the swap partition is recognized:

```
swapon -s
```

With the swap partition mounted, the output of the previous command should look something like this:

```
Filename Type Size Used Priority
/dev/sda1 partition 8914940 0 -2
```

We then inform the kernel about the changes with the command **systemctl** ↙ **daemon-reload**.

Now the installation can begin.

Last edited: 2023/11/09

4.9 LVM partitioning - Logical Volume Manager

The following is a basic introduction. It is up to the esteemed reader to delve deeper into the subject. Please refer to the respective man page. Further sources of information can be found at the end of this text - the list does not claim to be complete.

Working with *Logical Volumes* is much easier than most users think. The best feature of LVM is that changes take effect without having to reboot the system. *Logical Volumes* can span multiple disks and are scalable. This distinguishes them from other methods of disk partitioning.

You should be familiar with three basic terms:

- **Physical Volume [PV]:** This includes the physical, real-world disks or partitions such as **/dev/sda** or **/dev/sdb1** which are used for mounting/mounting. LVM can be used to combine multiple physical volumes into volume groups.

- **Volume Group [VG]:** A volume group consists of *physical volumes* and is the location of *logical volumes*. A Volume Group can be seen as a "virtual disk" composed of *physical volumes*. Here are some examples for better understanding:

  - Multiple storage devices (e.g. hard disks, SSDs, M2 disks, external USB drives, etc.) can be combined into a volume group (a virtual drive).

  - Multiple partitions of a storage device can be combined into one volume group (a virtual drive).

  - The two aforementioned options can be combined. For example, you could combine two complete SSDs with two partitions of a third SSD into a volume group.

- **Logical Volume [LV]:** Logical volumes are created within a *volume group* and mounted to the system. You can also consider them "virtual" partitions. They are dynamically modifiable, can be resized, recreated, removed, and used. A logical volume can span multiple physical volumes within the volume group.

4.9.1 Six steps to logical volumes

> **Caution**
>
> We assume non-partitioned hard disks in our example. Note: If old partitions are deleted, all data will be irretrievably lost.

As partition editor, either cfdisk or gdisk must be used because currently neither GParted nor the KDE partition manager support the creation of *logical volumes*. See also the manual pages:

[Partitioning with cfdisk (msdos-MBR)](#)

[Partitioning with gdisk (GPT-UEFI)](#)

All of the following commands and actions require **root** privileges.

1. Creating of a partition table

```
cfdisk /dev/sda
n  -> creates a new partition on the drive
p  -> this partition becomes a primary partition
1  -> the partition gets the number 1
      as identification "size allocation" sets the
      first and last cylinder to default values.
      Press ENTER to span the whole drive
t  -> selects the partition type to create
8e -> the hex code for a Linux LVM
W  -> writes changes to the drive
```

The command `W` writes the partitioning table. If a mistake was made up to this point, the existing partitioning layout can be restored. For this purpose, enter the command `q` for `cfdisk` to exit without writing, and everything remains as it has been before.

If the volume group is to span more than one physical volume (disk), the above operation must be performed on each physical volume.

2. Creating a physical volume [PV]

```
pvcreate /dev/sda1
```

The command creates the PV on the first partition of the first hard disk. This process is to be repeated on each partition as needed.

3. Creating a volume group [VG]

   Now we add three PV (/dev/sda1 /dev/sdb1 /dev/sdc1) to a VG named *vulcan*:

   ```
   vgcreate vulcan /dev/sda1 /dev/sdb1 /dev/sdc1
   ```

   If this step has been performed correctly, the result can be seen in the output of the following command:

   ```
   vgscan
   ```

   `vgdisplay` displays the size with:

   ```
   vgdisplay vulcan
   ```

4. Creating a logical volume [LV]

   At this point you have to decide how big the LV should be at the beginning. One advantage of LVM is the ability to adjust the size without rebooting.

   In our example, we want a 300GB LV named *spock* inside the VG named *vulcan*:

   ```
   lvcreate -n spock --size 300g vulcan
   ```

5. Formatting the LV

   Please be patient, this process may take some time:

   ```
   mkfs.ext4 /dev/vulcan/spock
   ```

6. Mounting the LV

   Create the mount point with

   ```
   mkdir /media/spock/
   ```

Using *"/dev/vulcan/spock"* is preferable to using UUID numbers with an LVM because it makes it easier to clone the file system (no UUID collisions). An LVM allows to create file systems with identical UUID numbers (classic example: snapshots). To mount the LV during the boot process, `fstab` must be customized with a text editor:

```
mcedit /etc/fstab
```

Then insert the following line according to our example:

```
/dev/vulcan/spock /media/spock/ ext4 auto,users,rw,exec,↙
    dev,relatime 0 2
```

Optional:

The owner of the LV can be changed so that other users have read/write access: ~~~ chown root:users /media/spock chmod 775 /media/spock ~~~

We can now repeat steps 4 to 6 for the new LV *kirk* to be created.

A simple LVM should now be usable.

4.9.2   Resizing a volume

> **Caution**
> Always make a data backup first.

We recommend using a live ISO to resize a LV. Although increasing the LV size of the running system can be done without error, decreasing the size of a LV cannot. Anomalies can lead to data loss, especially if the `/` (root) or `/home` directories are affected.

Example of an enlargement

The LV *spock* is to be enlarged from 300GB to 500GB.
We first check whether there is enough free memory.
`vgdisplay` provides information.

```
vgdisplay vulcan
```

```
[...]
Free PE / Size      170890 / 667,50 GiB
[...]
```

There is enough free storage space for our project.

We can start working.

Unmounting the LV:

```
umount /media/spock/
```

> **Caution**
> Never unmount your root file system during operation.

Extend the LV and its file system:

```
lvextend -L+200g --resizefs /dev/vulcan/spock
```

The `lvextend` command needs to be given the size **difference** as an option, not the total size desired.

The file system is then first checked with the option *"–resizefs"* and then adapted to the new size of the LV.

Finally, we mount the LV again.

```
mount /media/spock
```

The enlargement of an LV, even for the `/` (root) file system, is also possible during operation. Only the two commands `unmount` and `mount` are omitted. However, no file system check is then carried out.

If you want to check the root file system, use the kernel command line parameter `fsck.mode=force` during the boot process.

Example of a downsizing

The LV *spock* is downsized from 500GB to 280GB:

```
umount /media/spock/
```

Reduce the size of the file system:

```
e2fsck -f /dev/vulcan/spock
resize2fs /dev/vulcan/spock 280g
```

After that, the LV is changed.

```
lvreduce -L-220g /dev/vulcan/spock
resize2fs /dev/vulcan/spock
mount /media/spock
```

Again, the `lvreduce` command must be given the size **difference** as an option. The `resize2sf` command resizes the file system exactly to the LV size.

4.9.3   Manage LVM with a GUI program

Gparted offers the possibility to manage already created *logical volumes*. The program needs to be executed as **root**.

4.9.4   More info

Logical Volume Manager - Wikipedia
Working with logical volumes #1
Working with logical volumes #2
Working with logical volumes #3

Last edited: 2023/12/12

4.10   Move the home directory

> Important information
>
> An existing **/home** should not be used or shared with another distribution as there may/will be conflicts with the configuration files.

Therefore, we generally advise against creating a **/home** partition.

The directory `/home` should be the place where the individual configurations are stored, and only these. For all other private data, a separate data partition should be created, and this should be mounted under `/data`, for example. The advantages for data stability, data backup, and also in case of data recovery are almost immeasurable.

If data is to be shared for parallel installations, this procedure is particularly advisable.

Preparations

The necessary steps will be explained on a realistic example.

The initial situation:

- The old, meanwhile too small hard disk has three partitions (**/boot/efi**, **/**, **swap**).
- There is no separate data partition yet.
- An additional built-in hard disk has four partitions with the **ext4** file system. We will use this partition's **sdb4** as the new data partition, which we mount to `/data`.

Our previous `/etc/fstab` has the content:

```
$ cat /etc/fstab
...
UUID=B248-1CCA                                  /boot/efi vfat    ↵
   umask=0077 0 2
UUID=1c257cff-1c96-4c4f-811f-46a87bcf6abb /          ext4   ↵
   defaults,noatime 0 1
UUID=2e3a21ef-b98b-4d53-af62-cbf9666c1256 swap       swap   ↵
   defaults,noatime 0 2
```

```
tmpfs                                         /tmp      tmpfs  ↙
    defaults,noatime,mode=1777 0 0
```

We need the UUID information of the additional hard disk. See also the manual page customize fstab.

The command **blkid** returns the following information:

```
$ /sbin/blkid
...
/dev/sdb4: UUID="e2164479-3f71-4216-a4d4-af3321750322" ↙
    BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="000403b7-04"
```

Backup of the old /home

Before making any changes to the existing file system, we use **root** privileges to backup everything inside /home into a tar archive.

```
# cd /home
# tar cvzpf somewhere/home.tar.gz ./
```

Mountpoint of the data partition

We create the directory Data in / and mount the partition **sdb4** there. As owner and group we set our own names. Some time later, we will copy the private data, but not the configurations, from the existing /home into it.

Create mountpoint and mount partition (as **root**):

```
# mkdir /data
# chown <user>:<group> /data
# mount -t ext4 /dev/sdb4 /data
```

4.10.1   Move private data

Analysis of /home

Let's first take a close look at our home directory.

(The output has been sorted for clarity.)

```
~$ ls -la
total 169
drwxr-xr-x 19 <user><group> 4096  4 Oct 2020 .
drwxr-xr-x 62 <user><group> 4096  4 Oct 2020 ..
-rw------- 1  <user><group> 330  15 Oct 2020 .bash_history
-rw-r--r-- 1  <user><group> 220   4 Oct 2020 .bash_logout
-rw-r--r-- 1  <user><group> 3528  4 Oct 2020 .bashrc
drwx------ 19 <user><group> 4096 15 Oct 2020 .cache
drwxr-xr-x 22 <user><group> 4096 15 Oct 2020 .config
-rw-r--r-- 1  <user><group>   24  4 Oct 2020 .dmrc
drwx------ 3  <user><group> 4096 15 Oct 2020 .gconf
-rw-r--r-- 1  <user><group>  152  4 Oct 2020 .gitignore
drwx------ 3  <user><group> 4096 15 Oct 2020 .gnupg
-rw------- 1  <user><group> 3112 15 Oct 2020 .ICEauthority
-rw-r--r-- 1  <user><group>  140  4 Oct 2020 .inputrc
drwx------ 3  <user><group> 4096  4 Oct 2020 .local
drwx------ 5  <user><group> 4096 15 Oct 2020 .mozilla
-rw-r--r-- 1  <user><group>  807  4 Oct 2020 .profile
drwx------ 2  <user><group> 4096  4 Oct 2020 .ssh
drwx------ 5  <user><group> 4096 15 Oct 2020 .thunderbird
-rw------- 1  <user><group>   48 15 Oct 2020 .Xauthority
-rw------- 1  <user><group> 1084 15 Oct 2020 .xsession-error
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Desktop
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Documents
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Downloads
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Music
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Pictures
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Public
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Templates
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Videos
```

The output shows the home directory shortly after installation with only minor changes.

We put our private documents into the, by default created, directories `Desktop` to

`Videos` at the end of the list. These and possibly additional, self-created directories with private data, will be moved into the new data partition later.

"Hidden" files and directories beginning with a dot (.) contain configuration and program-specific data that we do not move, with three exceptions. These exceptions are:

the cache `.cache`,

the internet browser `.mozilla`, and

the mail program `.thunderbird`.

All three reach a considerable volume over time, and they also contain a lot of private data. Therefore, we move them to the new data partition, too.

Copying the private data

For copying, we use the command `cp` with the archive option `-a`. Thus the rights, owners, and the timestamp are kept, and it is copied recursively.

```
~$ cp -a * /data/
~$ cp -a .cache /data/
~$ cp -a .mozilla /data/
~$ cp -a .thunderbird /data/
```

The first command copies all files and directories except for the hidden ones.
The following output shows the result:

```
~$ ls -la /data/
total 45
drwxr-xr-x 13 <user><group> 4096  4 May 2020 .
drwxr-xr-x 20 root  root    4096  4 Oct 2020 ..
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 images-en
drwx------ 19 <user><group> 4096 15 Oct 2020 .cache
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Desktop
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Documents
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Downloads
drwx------ 5  <user><group> 4096 15 Oct 2020 .mozilla
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 music
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Public
drwx------ 5  <user><group> 4096 15 Oct 2020 .thunderbird
```

```
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 Videos
drwxr-xr-x 2  <user><group> 4096  4 Oct 2020 templates
```

To check the copy action for errors, you can use the command
dirdiff /home/<user>/ /data/. Only the files and directories that we did not copy should be listed.

Now all private data from the old /home are additionally on the new partition.

Delete in /home.

For this action, all program windows should be closed, except for the terminal we use.

Depending on the desktop environment, various applications use the directories created by default during installation (e.g. Music) to store files there. In order to enable the access of the applications to the directories, these must be linked back, thus refer to the corresponding directories of the **/data** partition.

> Please check the commands carefully before executing them so you don't accidentally delete something wrong.

```
~$ rm -r Desktop/ && ln -s /Data/Desktop/ ./Desktop
~$ rm -r Documents/ && ln -s /Data/Documents/ ./Documents
~$ rm -r Downloads/ && ln -s /Data/Downloads/ ./Downloads
~$ rm -r Music/ && ln -s /Data/Music/ ./Music
~$ rm -r Pictures/ && ln -s /data/Pictures/ ./Pictures
~$ rm -r Public/ && ln -s /Data/Public/ ./Public
~$ rm -r Templates/ && ln -s /Data/Templates/ ./Templates
~$ rm -r Videos/ && ln -s /Data/Videos/ ./Videos
~$ rm -r .cache/ && ln -s /data/.cache/ ./.cache
~$ rm -r .mozilla/ && ln -s /data/.mozilla/ ./.mozilla
~$ rm -r .thunderbird/ && ln -s /data/.thunderbird/ ./.↙
   thunderbird
```

The data remaining in the /home directory will only occupy less than 10 MB of space.

4.10.2   Adjust fstab

In order for the new data partition to be mounted and available to the user at system startup, the `fstab` file must be modified. Additional information about the `fstab` can be found in our manual adaptation of the fstab.
We need the data partition's already read out UUID information. Before modifying the file, we create a backup copy of the `fstab` with date attachment:

```
 # cp /etc/fstab /etc/fstab_$(date +%F)
 # mcedit /etc/fstab
```

According to our example, we add the following line to fstab.

UUID=e2164479-3f71-4216-a4d4-af3321750322 /data ext4 defaults,↙
noatime 0 2

The `fstab` should now look like this:

```
UUID=B248-1CCA                              /boot/efi vfat    ↙
   umask=0077 0 2
UUID=1c257cff-1c96-4c4f-811f-46a87bcf6abb / 	    ext4    ↙
   defaults,noatime 0 1
UUID=e2164479-3f71-4216-a4d4-af3321750322 /data     ext4    ↙
   defaults,noatime 0 2
UUID=2e3a21ef-b98b-4d53-af62-cbf9666c1256 swap      swap    ↙
   defaults,noatime 0 2
tmpfs                                       /tmp      tmpfs  ↙
   defaults,noatime,mode=1777 0 0
```

Save the file with **F2** and quit the editor with **F10**.

If, nonetheless, anything goes wrong, we still have our data in the saved tar archive.

Last edited: 2022/04/01

5 Network

This section contains informations and notes on

- the Network Manager command line tool, its operation, which devices are present and configured, how to connect and disconnect, and how to switch from WLAN to LAN and back.

- the IWD, Intel's wireless daemon replaces the WPA supplicant. Its operation, and how its work with the Network Manager.

- Setting up a SAMBA Client to access Windows shares.

- Setting up and secure SSH

- setting up a LAMP test server for developers (local), its components and how to install them, where to find the log files, and how to fix any errors that may occur.

- LAMP - Apache, its directories in the file system, configuration, user and rights management, secure operation as a local server, and the use of HTTPS.

- LAMP - MariaDB, its directories in the file system, initial configuration, the command line interface `mariadb`, phpMyAdmin, and integration with systemd.

- LAMP - PHP, the directories in the file system, PHP support for Apache2, configuration, installation as well as handling of modules, and where to find the log files.

Last edited: 2022/03/23

5.1   Network Manager Command Line Tool

General hints

The network manager is now integrated in all graphical user interfaces of siduction and is mostly self-explanatory. It replaces the network commands `ifup`, `ifdown`, and `ifconfig` used in the terminal. The prejudice that the network manager is not suitable for the command line or even runs unstable belongs to the realm of fairy tales. If no graphical user interface is available, or the command line is preferred, **nmcli** is a powerful command line alternative for the daily use of the network manager.

In the following examples we assume two configured connections: a Wi-Fi connection (name: *"Unicorn_2"*, interface *"wtx7ckd90b81bbd"*, (formerly: wlan)) and a wired connection (name: *"WiredConnection_1"*, interface *"evp0s3f76"* (former: eth0)). Please adapt the connection names to your circumstances.

Installation of the Network Manager

If the network manager is not available on your system, you can install it. The following command will install all packages you need to configure every possible connection type (mobile broadband, Wi-Fi, and LAN connections), as well as the graphical KDE plasma widget for the NM. Please enter everything in one line and remember that you need **root** privileges.

```
 # apt install network-manager modemmanager mobile-broadband-↙
    provider-info network-manager-pptp
  plasma-nm network-manager-vpnc network-manager-openvpn
```

5.1.1   Use Network Manager

The entries can be made both in a virtual terminal (key combination **Ctr**+**Shift**↙ +**F2**) and in the console of a graphical user interface. In the examples shown, the information has been changed for privacy reasons.

Show configured connections

The command `nmcli c` can be used to display the configured connections that have been created on the system.

```
nmcli c
NAME                UUID                                    TYPE      DEVICE
WirelessAdapter_2   4c247331-05bd-4ae6-812b-6c70b35dc348    wifi      wtx7ckd90b81bbd
WiredConnection_1   847d4195-3355-33bc-bea8-7a016ab86824    ethernet  evp0s3f76
WiredConnection_2   efc70b04-01f1-31fc-b948-5fd9ceca651d    ethernet  --
MobileNetworkUMTS   fe0933bc-f5fa-4b94-8622-d03c4195721e    gsm       xyz72905dg34
```

In the above example, there are four connections: WLAN, 2x LAN, and a mobile broadband connection.

Show information about Wi-Fi networks.

To display all available Wi-Fi networks in a compact form, use the command `nmcli dev wifi list`.

```
nmcli dev wifi list
IN-USE  BSSID             SSID       MODE   CHAN  RATE        SIGNAL  BARS  SECURITY
*       14:CF:20:C6:1A:8F  WLAN-01    Infra  6     270 Mbit/s  92            WPA2
        54:67:64:3D:02:30  WLAN-02    Infra  1     405 Mbit/s  85            WPA2
        D0:AA:2A:17:EE:9B  WLAN-03    Infra  11    270 Mbit/s  52            WPA2
```

Display configured devices

If you want to know which devices (interfaces) are known to the network manager, enter `nmcli d`.

```
nmcli d
DEVICE           TYPE      STATE          CONNECTION
evp0s3f76        ethernet  connected      WiredConnection_1
wtx7ckd90b81bbd  wifi      connected      Unicorn_2
evp3u3           ethernet  unavailable    --
ttyACM0          gsm       not connected  --
```

Very detailed information (properties) about the own available connections can be displayed through `nmcli dev show`. Here only the excerpt for the Wi-Fi:

```
nmcli dev show
[...]
GENERAL.DEVICE:                       wtx7ckd90b81bbd
GENERAL.TYPE:                         wifi
GENERAL.HWADDR:                       7C:FA:83:C2:6B:BD
GENERAL.MTU:                          1500
GENERAL.STATE:                        100 (connected)
GENERAL.CONNECTION:                   Unicorn_2
GENERAL.CON-PATH:                     /org/freedesktop/NetworkManager/ActiveConnection/2
IP4.ADDRESS[1]:                       192.168.0.6/24
IP4.GATEWAY:                          192.168.0.1
IP4.ROUTE[1]:                         dst = 0.0.0.0/0, nh = 192.168.0.1, mt = 600
IP4.ROUTE[2]:                         dst = 192.168.0.0/24, nh = 0.0.0.0, mt = 600
IP4.DNS[1]:                           192.168.0.1
IP4.DOMAIN[1]:                        home
IP6.ADDRESS[1]:                       2a02:810d:cc0:c4c:7edd:90ff:feb2:1bbd/64
IP6.ADDRESS[2]:                       fe80::7edd:90ff:feb2:1bbd/64
IP6.GATEWAY:                          fe80::362c:c4ff:fe17:1bf1
IP6.ROUTE[1]:                         dst = 2a02:810d:cc0:c4c::/64, nh = ::, mt = 256
IP6.ROUTE[2]:                         dst = fe80::/64, nh = ::, mt = 256
IP6.ROUTE[3]:                         dst = ::/0, nh = fe80::dc53:e2ff:fe81:6d46, mt = 1024
IP6.ROUTE[4]:                         dst = ::/0, nh = fe80::362c:c4ff:fe17:1bf1, mt = 1024
IP6.ROUTE[5]:                         dst = ff00::/8, nh = ::, mt = 256, table=255
[...]
```

The Wi-Fi access data can be displayed with `nmcli dev wifi show`.

```
nmcli dev wifi show
SSID: Unicorn_2
Security: WPA
Password: That's not here now
```


The additionally generated QR code simplifies the login for smartphones and tablets.

Switch connections

To change a connection type, e.g. from LAN to Wi-Fi connection, you have to disconnect the existing active connection and activate the new one. Here, you definitely have to specify the interface because a **`nmcli con down id <name>`** will work, but the connection, if it is a system connection, will be re-established immediately.

To prevent the automatic connection, the command **`nmcli dev disconnect <`**↙
`interface_name>` is useful.
First we terminate the LAN connection and then query the status:

```
# nmcli dev disconnect evp0s3f76
Device "evp0s3f76" has been disconnected successfully.
# nmcli dev status
DEVICE TYPE STATE CONNECTION
evp0s3f76 ethernet not connected --
wtx7ckd90b81bbd wifi not connected --
evp3u3 ethernet not available --
ttyACM0 gsm not connected --
```

Now enable the Wi-Fi connection with **`nmcli con up id <connection_name>`**:

```
# nmcli con up id Unicorn_2
Connection was successfully activated
# nmcli dev status
DEVICE TYPE STATE CONNECTION
wtx7ckd90b81bbd wifi connected Unicorn_2
evp0s3f76 ethernet not connected --
evp3u3 ethernet not available --
ttyACM0 gsm not connected --
```

You can also put everything in one line, then the change will take effect immediately.

From LAN to Wi-Fi:

```
nmcli dev disconnect evp0s3f76 && sleep 2 && nmcli con up id ↙
    Unicorn_2
```

Reversed from Wi-Fi to LAN:

```
nmcli dev disconnect wtx7ckd90b81bbd && sleep 2 && nmcli con ↙
    up id 'WiredConnection_1'.
```

5.1.2   Further information

•
```
man nmcli
```

- Ubuntu community NetworkManager

Last edited: 2022/04/03

5.2   IWD

Intel's iNet wireless daemon (iwd) sends wpa-supplicant into well-deserved retirement. Only a tenth the size and much faster, iwd is the successor. It works alone or together with NetworkManager, systemd-networkd, and conman.
In rare cases, connections are lost with iwd. It is then advisable to return to the WPA supplicant.

Further information can be found on the Arch Linux wiki or the debian wiki.

Since siduction 2021.3.0, iwd is used as the standard for establishing connections to WLAN. Our implementation runs with NetworkManager.

Since siduction 2021.1.0, iwd has already been delivered in the flavours Xorg and NoX. If you want, you can install iwd on the other flavours. See below: IWD instead of wpa_supplicant.

Before siduction 2021.1.0: Even with a slightly older snapshot, iwd can be installed (tested with siduction 2018.3.0 and linux-image-5.15.12-1-siduction-amd64). Please also follow the instructions at IWD instead of wpa_supplicant.

5.2.1   Graphical configuration programs

- **NetworkManager**: For the NetworkManager, there are different graphical interfaces, e.g. `plasma-nm` for plasma-desktop/kde or `network-manager-⤸ gnome` for gnome and others. Their usage should be self-explanatory!
- **conman** is a small and resource saving network manager developed by Intel. Read more about it on the Arch-Wiki
- **iwgtk** is not available in debian sources. It has to be built from source code and can be found on github.

5.2.2   Configuration in terminal

iwd and NetworkManager

1. The fastest and easiest way to use iwd with NetworkManager is to open a terminal and type this command:

```
~$ nmtui
```

This will start the NetworkManager's text based graphical interface in the terminal. The program should be self-explanatory!

2. Use the NetworkManager's command line tool `nmcli`. Detailed information about this can be found on our manual page NetworkManager in the terminal.

   The following is a brief description of the fastest way to set up a network with the help of NetworkManager on the command line. Provided you have all the information, this one-liner is enough:

```
~$ nmcli dev wifi con "<ssid>" password <password> name "↙
    <name>"
```

   (*"ssid"* denotes the name of the network.)

   For example:

```
nmcli dev wifi con "HomeOffice" password R3allY+v3ry+↙
    s3creT name "HomeOffice"
```

iwd standalone (without NetworkManager)

Intel's iwd comes with its own command line tool called *iwctl*. Please only use iwctl if NetworkManager and wpa_supplicant are not installed or both are masked in systemd.

First we should call the help of iwctl to see what is possible. For this, we enter the command **iwctl** into the terminal and then **help** into the input prompt.

```
[iwd]# help
                        iwctl version 1.12
--------------------------------------------------------------------
 Usage
--------------------------------------------------------------------
 iwctl [--options] [commands]

                        Available options
--------------------------------------------------------------------
 Options                                Description
--------------------------------------------------------------------
 --username                             Provide username
 --password                             Provide password
 --passphrase                           Provide passphrase
 --dont-ask                             Don't ask for missing
                                        credentials
 --help                                 Display help

                        Available commands
--------------------------------------------------------------------
 Commands                               Description
--------------------------------------------------------------------

Adapters:
  adapter list                          List adapters
  adapter <phy> show                    Show adapter info
  adapter <phy> set-property <name> <value>   Set property

Ad-Hoc:
  ad-hoc list                           List devices in Ad-hoc mode
  ad-hoc <wlan> start <"network name"> <passphrase> Start or join an existing
                                        Ad-Hoc network called
                                        "network name" with a
                                        passphrase
```

To find out which Wi-Fi interface we are using, we enter the following command:

```
[iwd]# device list
            Devices                                      *
----------------------------------------------------
Name    Address             Powered   Adapter   Mode
----------------------------------------------------
wlan0   00:01:02:03:04:05   on        phy0      station
```

In this case, it is *"wlan0"* and it is running (*"powered on"*) in *"station"* mode.

Now we scan for an active network:

```
[iwd]# station wlan0 scan
[iwd]# station wlan0 get-networks
```

After that, we can connect to our network:

```
[iwd]# station wlan0 connect <ssid>
```

(*"ssid"* means the name of the network.)

We are asked for the password and we should then be connected to our network. We can check this with **station list** or **station wlan0 get-networks**.

```
[iwd]# station list
    Devices in Station Mode
-----------------------------
Name        State        Scanning
-----------------------------
wlan0       connected
```

The whole process can be abbreviated by the following command if you have all the necessary information!

```
iwctl --passphrase <passphrase> station <device> connect <↙
    ssid>
```

For example:

```
~$ iwctl --passphrase W1rkl1chS3hrG3h31m station wlan0 ↙
    connect HomeOffice
```

5.3 IWD instead of wpa_supplicant

For those who want to use iwd as a replacement for wpa_supplicant with a slightly older snapshot than siduction 2021.3.0, please follow the instructions below.

5.3.1 Install IWD

> Note:
> It is possible that non-free firmware must be installed from a USB stick or via LAN! Under Debian, it is unfortunately not possible to install the Network-Manager (standalone) without wpa_supplicant.

If you want to do this, there are two options. The second one is more sensible and easier.

1. Install NetworkManager from the sources.
2. Do not start or mask the wpa_supplicant.service. Since siduction uses systemd, we will not go into how iwd is configured without systemd!

If you want to use iwd without NetworkManager, you don't have to worry about that, but you have to remove NetworkManager and wpa_supplicant from the disk together with their configuration:

```
~# apt purge network-manager wpasupplicant
```

Procedure with NetworkManager installed
and iwd < 1.21-2

- First install **iwd**,
- then stop the **NetworkManager.service**,
- then stop and mask the **wpa_supplicant.service**.
- Now create the file `/etc/NetworkManager/conf.d/nm.conf` and enter **iwd** there,
- then create the file `/etc/iwd/main.conf` and fill it with appropriate content,
- activate and start the **iwd.service**,
- and start the **NetworkManager.service**.

Now just run the following commands as **root** in the terminal to use iwd:

```
~# apt update
~# apt install iwd
~# systemctl stop NetworkManager.service
~# systemctl disable --now wpa_supplicant.service
~# echo -e '[device]\nwiFi.backend=iwd' > /etc/NetworkManager↵
   /conf.d/nm.conf
~# touch /etc/iwd/main.conf
~# echo -e '[General]\nEnableNetworkConfiguration=true \n\n[↵
   Network]\nNameResolvingService=systemd' > /etc/iwd/main.↵
   conf
~# systemctl enable -now iwd.service
~# systemctl start NetworkManager.service
```

See if it worked

We display the two configuration files.

- /etc/NetworkManager/conf.d/nm.conf

```
~$ cat /etc/NetworkManager/conf.d/nm.conf
[device]
wiFi.backend=iwd
```

- /etc/iwd/main.conf

```
~$ cat /etc/iwd/main.conf
[General]
EnableNetworkConfiguration=true


[Network]
NameResolvingService=systemd
```

**Procedure with NetworkManager installed
and iwd >= 1.21-2**

From version 1.21-2 on, iwd brings its own configuration file /etc/iwd/main.⤸
conf. The procedure is similar to the one just mentioned with the exception that
we do not create the configuration file anymore, but remove the comment sign in
front of *"EnableNetworkConfiguration=true"* in it.

Please execute the following commands as **root** in the terminal:

```
~# apt update
~# apt install iwd
~# systemctl stop NetworkManager.service
~# systemctl disable --now wpa_supplicant.service
~# echo -e '[device]\nwiFi.backend=iwd' > /etc/NetworkManager⤸
   /conf.d/nm.conf
~# sed -i 's/#EnableNetworkConfiguration=true/⤸
   EnableNetworkConfiguration=true/' /etc/iwd/main.conf
~# systemctl enable -now iwd.service
~# systemctl start NetworkManager.service
```

See if it worked

We display the two configuration files.

- /etc/NetworkManager/conf.d/nm.conf

```
~$ cat /etc/NetworkManager/conf.d/nm.conf
[device]
wiFi.backend=iwd
```

- /etc/iwd/main.conf

```
~$ cat /etc/iwd/main.conf

[...]
[General]
# iwd is capable of performing network configuration on its
# own, including DHCPv4 based address configuration.
# By default this behavior is disabled, and an external
# service such as NetworkManager, systemd-network or
# dhcpclient is required.  Uncomment the following line if
# you want iwd to manage network interface configuration.
#
EnableNetworkConfiguration=true
#
[...]
```

With the commands described above, you are now able to display Wi-Fi hardware in the terminal **nmtui**, **nmcli**, or **iwctl**, configure it, and connect to a network. Or you can use the NetworkManager in the graphical user interface. See: graphical-configuration-programs

5.3.2   Back to wpa_supplicant

(Provided NetworkManager and wpa_supplicant are installed.)

- Stop the **NetworkManager.service**.
- Stop the **iwd.service** and mask it.

- Rename the **/etc/NetworkManger/conf.d/nm.conf** file.
- Unmask and start the **wpa_supplicant.service**.
- Restart the **NetworkManager.service**.

```
~# systemctl stop NetworkManager.service
~# systemctl disable --now iwd.service
~# mv /etc/NetworkManager/conf.d/nm.conf /etc/NetworkManager/↵
   conf.d/nm.conf~
~# systemctl unmask wpa_supplicant.service
~# systemctl enable --now wpa_supplicant.service
~# systemctl start NetworkManager.service
```

Now wpa_supplicant is used to connect to the Wi-Fi hardware.

Last edited: 2023/11/09

5.4 SAMBA

5.4.1 Client configuration

How to access a Windows network share using siduction

- All commands are executed in a terminal or console as **root**.

- The URL is called in Dolphin as **normal user** .
  "server" = server name or IP of the Windows machine
  "share" = name of the share

  In the KDE file manager Dolphin, the URL is entered as follows: `smb://↙`
  `server` or with the full path: `smb://server/share`.

In a console, the shares on a server can be displayed with:

```
smbclient -L <server>
```

To see a share in a directory (with access for ALL users), a mount point must exist. If not, a directory must be created as a mount point (the name is arbitrary):

```
mkdir -p /mnt/server_share
```

A share is mounted with this command:

```
mount -t cifs -o username=administrator,uid=$UID,gid=$GID //↙
    server/share /mnt/server_share
```

If you get an error message here, it may be due to the SMB protocol version you are using. In Debian, SMB 1.0 is no longer used for security reasons. Unfortunately, there are still systems which provide only SMB 1.0. To get access to such a share, the mount option `vers=1.0` is needed. The complete command is:

```
mount -t cifs -o username=Administrator,vers=1.0,uid=$UID,gid↙
    =$GID //server/share /mnt/server_share
```

A connection is terminated with this command:

```
umount /mnt/server_share
```

To mount a Samba share automatically, the `/etc/fstab` file can be amended according to this pattern (all in one line):

```
//server/share /mnt/server_share cifs noauto,x-systemd.
automount,x-systemd.idle-timeout=300,user=username,
password=**********,uid=$UID,gid=$GID 0 0
```

However, it is not recommended to write the password in plain text to fstab. A better alternative is to create `.smbcredentials` with the following content:

```
username=<user>
password=<password>
```

The resulting entry for `/etc/fstab` is (all in one line):

```
//server/share /mnt/server_share cifs noauto,x-systemd.
automount,x-systemd.idle-timeout=300,credentials=</path/
to/.smbcredentials>,uid=$UID,gid=$GID 0 0
```

The variables *"UID"* and *"GID"* correspond to those of the user to whom the share should be given. But you can also write `uid=<username>` and `gid=<groupname>`.

5.4.2   siduction as samba server

Of course, siduction can also provide an SMB server. Describing the setup as a Samba server here in the manual would go beyond its scope. The internet provides many HowTo's on this topic.

Our recommendations:

debian - a minimal Samba setup
Raspberry Pi - samba server
ubuntu - install and configure samba
redhat - using samba as a server

There are many more sites on this topic on the web.

Last edited: 2022/04/20

5.5   SSH

> **Enable SSH**.
>
> For siduction, ssh is not enabled both on the live ISO and after installation!
> To enable and disable *ssh* please use the scripts `sshactivate` and `sshdeactivate`. They are located in `/usr/sbin`.
> Alternatively use the starters in the application menu > Internet/Network.

Definition of SSH from Wikipedia :

Secure Shell or SSH refers to both a network protocol and corresponding programs that can be used to establish an encrypted network connection with a remote device in a secure manner. Often this method is used to bring a remote command line to the local computer, i.e. the local console displays the output of the remote console and the local keyboard input is sent to the remote computer. This gives the effect of sitting in front of the remote console, which can conveniently be used for remote maintenance of, for example, a root server located in a remote data center. The newer protocol version SSH-2 offers further functions like data transfer via SFTP.

In general, SSH should only be configured and used for remote control if you are sure about the effects of the settings in the `/etc/ssh/sshd_config` file and, if applicable, the files in the `/etc/ssh/sshd_config.d/` directory.
If in doubt, study the documentation in the man pages **man sshd_config**, **man ⟋ ssh_config** or on the Internet, or seek the advice of competent persons.

5.5.1   Securing SSH

It is not secure to make root logins via SSH the default. Debian should be secure, not insecure. Likewise, attackers should not be able to perform a wordlist-based password attack (brute force attack) on the SSH login over a long period of time. Therefore, we make some changes in the `/etc/ssh/sshd_config` file.

The following settings can be adjusted to increase security:
(To activate the function of the entries the comment character '#' must be removed).

- **Port** :

  This entry must point to the port that is enabled on the router for forwarding. IANA has assigned TCP port 22 to the protocol and Debian sets it as the default. However, it is advisable to use a port outside the default scanning range. We therefore use port 5874, for example, to make attacks more difficult, since the SSH port is not known to the attacker.

  ```
  Port 5874
  ```

- **ListenAddress** :

  Since the port is forwarded by the router, the computer must use a static IP address unless a local DNS server is used. But if something as complicated as SSH is to be set up using a local DNS server and these instructions are needed, a serious error can occur easily. We'll use a static IP for the example:

  ```
  ListenAddress 192.168.2.134
  ```

  The SSH protocol 2 with its improved and extended features is already default in Debian.

- **LoginGraceTime** :

  By deault, the allowed time is 2 minutes. Since it usually doesn't take two minutes to enter a username and password, we set a slightly shorter time period:

  ```
  LoginGraceTime 30
  ```

  Now you have 30 seconds to log in, and hackers don't have two minutes each time they try to crack the password.

- **PermitRootLogin <prohibit-password>**:

  Why Debian gives permission to log in as root here is not understandable. We correct to no:

  ```
  PermitRootLogin no
  StrictModes yes
  ```

Alternatively, you can set this option to *forced-commands-only*. This will allow root to log in via asymmetric authentication, but only if the *command* option is set (which is useful for performing remote backups, even if root logins are not normally allowed). All other authentication methods for root remain disabled.

- **MaxAuthTries** :
  More than 3 or 4 attempts should not be allowed:

  ```
  MaxAuthTries 3
  ```

The following settings must be added if they are not present:

- **AllowUsers** :
  Usernames which are allowed to access via SSH, separated by spaces. Only registered users can use the access, and only with user rights. With `adduser` you should add a user that is specifically meant to use SSH:

  ```
  AllowUsers whoever1 whoever2
  ```

- **PermitEmptyPasswords** :
  The user should be given a nice and long password that can't be guessed in a million years. They should be the only one with SSH access. Once logged in, they can become **root** with **su**:

  ```
  PermitEmptyPasswords no
  ```

- **PasswordAuthentication** :
  Obviously, 'yes' must be set here (unless you use a KeyLogin).

  ```
  PasswordAuthentication yes
  ```

Finally:

```
systemctl restart ssh
```

Now you have a somewhat secure SSH configuration. Not completely secure, just better, especially if you have added a user specifically for using SSH.

5.5.2 SSH for X Window Programs

ssh -X allows you to connect to a remote computer and display its X graphics server on your own local computer. You enter the command as **user** (not **root**) (and note that X is a capital letter):

```
$ ssh -X username@xxx.xxx.xxx.xxx (or IP)
```

Enter the password for the remote computer's username and start a graphical application in the shell. Examples:

```
$ iceweasel OR libreoffice OR kspread
```

On very slow connections, it may be advantageous to use the compression option to increase the transfer rate. However, for fast connections, the opposite effect may occur:

```
$ ssh -C -X username@xxx.xxx.xxx.xxx (or IP)
```

Further information can be obtained with **man ssh**.

Note:

If ssh refuses a connection and you get an error message, search in `$HOME` for the hidden directory `.ssh`, delete the file `known_hosts` and try a new connection. This problem occurs mainly when you have assigned the IP address dynamically (DCHP).

5.5.3 Copy scp via ssh

scp is a command line utility (Terminal/CLI) to copy files between network computers. It uses ssh for authentication and secure file transfer, so scp requires a password or passphrase to log in.

If you have ssh rights on a network PC or network server, scp allows you to copy partitions, directories, or files to or from a network computer (or an area on it) that you have access rights to. This can be, for example, a PC or server on the local

network, a computer on a remote network, or a local USB drive. The copy operation can take place between remote computers/storage devices.

It is also possible to recursively copy entire partitions or directories with `scp -r`. Note that this command also follows symbolic links in the directory tree.

Examples

1. Copying a partition:

   ```
   scp -r <user>@xxx.xxx.x.xxx:/media/disk1part6/ /media/↵
       diskXpartX/
   ```

2. Copying a directory on a partition, in this case a directory named `photos` in `$HOME`:

   ```
   scp -r <user>@xxx.xxx.x.xxx:~/photos/ /media/diskXpartX/↵
       xx
   ```

3. Copying a file in a partition's directory, in this case a file in `$HOME`:

   ```
   scp <user>@xxx.xxx.x.xxx:~/filename.txt /media/diskXpartX↵
       /xx
   ```

4. Copying a file on a partition:

   ```
   scp <user>@xxx.xxx.x.xxx:/media/disk1part6/filename.txt /↵
       media/diskXpartX/xx
   ```

5. If you are in the drive or directory where another directory or file shall be copied to, use only a `.` (dot):

   ```
   scp -r <user>@xxx.xxx.x.xxx:/media/disk1part6/filename.↵
       txt .
   ```

Additional information:

```
man scp
```

5.5.4 SSH with Dolphin or Thunar

Dolphin and Thunar are capable of accessing data from a remote machine using the *"sftp"* protocol present in ssh.

This is how it is done:

1. **For a remote server**.
   One opens a new file manager window or a new tab.
   Enter into the address bar according to the pattern:
   - `sftp://username@ssh-server.com`
     Then a dialog window opens and asks for the SSH password. One enters the password and clicks OK,

   - or the input already contains the password:
     `sftp://username:password@remote_hostname_or_ip`
     You will not be asked for a password, you will be connected directly.
2. **For a LAN environment**.
   Following the same pattern as before:
   - `sftp://username@192.168.x.x`
     with dialog window for SSH password,

   - or immediately with password:
     `sftp://username:passwort@192.168.x.x`

Please enter the correct IP!
A SSH connection is now established. In this window, you can work with the files on the SSH server as if they were local files.

NOTE:
If a port other than 22 (default) is used, it must be specified when using sftp. The input then follows the syntax:
`sftp://user@ip:port`,
this is the default syntax for many protocols/programs like sftp and smb.

5.5.5   SSHFS - mount on a remote computer

SSHFS is a simple, fast, and secure method using FUSE to mount a remote filesystem. On the server side, all you need is a running ssh daemon.

On the client side, you probably need to install sshfs first:

```
apt update && apt install sshfs
```

`fuse3` and `groups` are already on the ISO and do not need to be installed separately.

Mounting a remote filesystem is very easy:

```
sshfs -o idmap=user username@remote_hostname:directory ↙
    local_mountpoint
```

If no specific directory is specified, the remote user's home directory will be mounted. Please note: the colon *":"* is mandatory even if no directory is specified!

Once mounted, the remote directory behaves like any other local file system. You can browse, read and modify files, and execute scripts just like on a local file system.

Mounting the remote host is accomplished with the following command:

```
fusermount -u local_mountpoint
```

If you use sshfs regularly, it is recommended to make an entry in `/etc/fstab` (all in one line):

```
sshfs#remote_hostname://remote_directory /local_mount_point
  fuse -o idmap=user ,allow_other,uid=1000,gid=1000,noauto,
  fsname=sshfs#remote_hostname://remote_directory 0 0
```

Next, remove the comment character before *"user_allow_other"* in the file `/etc↙`
`/fuse.conf`:

```
# Allow non-root users to specify the 'allow_other'
# or 'allow_root' mount options.
```

```
#
user_allow_other
```

This allows any user in the fuse group to mount or unmount the filesystem:

```
mount /path/to/mount/point # mount
umount /path/to/mount/point # unmount
```

Use this command to check if you are a member of the fuse group:

```
cat /etc/group | grep fuse
```

The answer should look something like this:

```
fuse:x:117: <username>
```

If the username is not listed, use the `adduser` command as **root**:

```
adduser <username> fuse
```

Note:
The user will not be a member of the group "fuse" until he logs in again.
Now the desired username should be listed and the following command should be executable:

```
mount local_mountpoint


    and


umount local_mountpoint
```

Last edited: 2023/10/19

5.6   LAMP web server

A local test server for developers

The acronym **LAMP** refers to a set of free software used to run dynamic web sites:

- **L**inux: operating system
- **A**pache: web server
- **M**ariaDb: database server (as of Debian 9 'Stretch', previously mySQL)
- **P**HP, Perl, and/or Python: scripting languages

Use cases as a server:

1. **a local test server for webdesigners without Internet connection (see this chapter)**
2. a private (data) server with Internet connection
3. a private web server with full Internet connection
4. a commercial web server

Our goal is to set up a LAMP test server for developers that is directly connected to the workstation PC via LAN. Furthermore, for security reasons, the server should not be connected to a local network or even to the Internet. The only exception is that the server will be temporarily connected to the Internet via a second network interface exclusively for system and software updates.

> Please note:
> The desktop PC used for daily work should not be used as a server. Instead, a separate PC should be used, which does not perform any other tasks.

At least 500MB RAM should be available in the server PC. Less RAM will cause problems because a server with MariaDb/MySQL needs a lot of RAM to run appropiately.

The packages to install are:

```
apache2
mariadb-server
mariadb-client
```

```
php
php7.4-mysql
phpmyadmin
```

As usual with siduction, we run the installations in the "multi-user.target" (init 3) in the terminal.

Preparations

If the command line browser *w3m* has not been installed yet, we will do it now:

```
# apt update
# apt install w3m
```

This allows us to test *Apache* and *PHP* immediately in the terminal and return to the graphical user interface only after all necessary installations have been completed.

Now we need to clean up apt.
The command **apt autoremove** should result in the following output. If not, we confirm the removal of unneeded packages with `j`.

```
#apt autremove
Package lists are read... Done
Dependency tree is built.
Status information is read.... Done
0 updated, 0 reinstalled, 0 to remove, and 0 not updated.
```

In case of a corrupted installation, this will simplify the repair.
See below Troubleshooting.

It is useful to note down some data already before the installation.

Necessary during the installation:

- a **password** for the database user **root** in *phpMyAdmin*

Later, necessary for the configuration:

- **Apache**
    - *Server Name*

- – *Server alias*
- – the server's *IP address*
- – the PC's *Name*
- – the PC's *IP address*
- **MariaDB:**
  - – the *name of the database* to be used for the development project
  - – the *name* (login name) of a new databank user for the development project
  - – the *password* for the new databank user
  - – the *name* (login name) of a new database administrator
  - – the *password* for the database administrator

5.6.1   Install Apache

In order to install the Apache web server, you only need the following two commands. The install command gets the additional packages *apache2-data* and *apache2-utils*. Then we query the status of Apache and test the start and stop instructions right away.

```
# apt update
# apt install apache2
[...]
The following NEW packages will be installed:
  apache2 apache2-data apache2-utils
[...]
Do you want to continue? [Y/n] y
[...]

# systemctl status apache2.service
apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/system/system/apache2.service;
             enabled; vendor preset: enabled)
     Active: active (running) since Sun 2020-12-06 [...]
[...]
```

As you can see, Apache has been activated immediately.

```
# systemctl stop apache2.service
# systemctl status apache2.service
apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/system/system/apache2.service;
             enabled; vendor preset: enabled)
     Active: inactive (dead) since Sun 2020-12-06 [...]
[...]

# systemctl start apache2.service
# systemctl status apache2.service
apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/system/system/apache2.service;
             enabled; vendor preset: enabled)
     Active: active (running) since Sun 2020-12-06 [...]
[...]
```

The Apache web server is loaded and can be handled without any problems. Now we check its function with:

```
w3m http://localhost/index.html
```

The Apache welcome page with **It works!** appears.
We exit w3m with **q** and confirm with **y**.

The directory **/etc/apache2/** is called **ServerRoot**. It contains the configuration. The directory **/var/www/html/** is called **DocumentRoot**. It contains the website's files.

For more information and security hints, please refer to the manual page LAMP-Apache.

5.6.2 Install MariaDb

The installation of MariaDb is similarly simple. Just install the metapackages *mariadb-server* and *mariadb-client*.

```
# apt install mariadb-server mariadb-client
[...]
The following NEW packages will be installed:
galera-4 libcgi-fast-perl libcgi-pm-perl libdbd-mariadb-perl
 libfcgi-perl libhtml-template-perl libmariadb3
 mariadb-client mariadb-client-10.5 mariadb-client-core-10.5
 mariadb-common mariadb-server mariadb-server-10.5
 mariadb-server-core-10.5 mysql-common socat
[...]
Do you want to continue? [Y/n] y
```

For more information on MariaDb and configuration, see our manual in LAMP-MariaDB

5.6.3   Install PHP

To install the PHP scripting language, simply enter the command:

```
# apt install php
[...]
The following NEW packages will be installed:
apache2-bin libapache2-mod-php7.4 libaprutil1-dbd-sqlite3
 libaprutil1-ldap php php-common php7.4 php7.4-cli
 php7.4-common php7.4-json php7.4-opcache php7.4-readline
[...]
Would you like to continue? [Y/n] y
```

As before, the metapackage additionally brings in a whole bunch of dependencies. To check if php is running correctly after installation, create the file `info.php` in `/⤸ var/www/html` using the phpinfo() function this way:

```
mcedit /var/www/html/info.php
```

Insert the following text:

```
<?php
```

```
phpinfo();
?>
```

Save with **F2** and terminate mcedit with **F10**.

After that the terminal browser w3m will be linked to it:

```
w3m http://localhost/info.php
or
w3m http://yourip:80/info.php
```

```
PHP logo

PHP version 7.4.11

System      Linux <hostname> 5.9.13-towo.1-siduction-amd64
Build Date  Oct 6 2020 10:34:39
server API  Apache 2.0 Handler
...
```

If we get an output that starts as shown above and contains all php configurations and basic settings, PHP is working and uses the *Apache 2.0 Handler* as *server API*.

We exit w3m with **q** and confirm with **y**.

Now we need the module *php7.4-mysql*, so MariaDB/mysql will be supported in PHP.

```
# apt install php7.4-mysql
```

If we now go back to the "http://localhost/info.php" page, we will find the entries for *mysqli* and *mysqlnd* in the modules section (they are sorted alphabetically).

For more information on configuring PHP and managing its modules, see the manual page LAMP-PHP

5.6.4 Install phpMyAdmin

To administer the MariaDb database, we need *phpmyadmin*:

```
# apt install phpmyadmin
[...]
The following NEW packages will be installed:
dbconfig-common dbconfig-mysql icc-profiles-free
libjs-openlayers libjs-sphinxdoc libjs-underscore
libonig5 libzip4 php-bacon-qr-code php-bz2
php-dasprid-enum php-gd php-google-recaptcha
php-mbstring php-mysql php-phpmyadmin-motranslator
php-phpmyadmin-shapefile php-phpmyadmin-sql-parser
php-phpseclib php-psr-cache php-psr-container
php-psr-log php-symfony-cache php-symfony-cache-contracts
php-symfony-expression-language php-symfony-var-exporter
php-symfony-service-contracts php-tcpdf php-twig
php-twig-extensions php-xml php-zip php7.4-bz2 php7.4-gd
php7.4-mbstring php7.4-xml php7.4-zip phpmyadmin
0 updated, 38 reinstalled, 0 to remove and 60 not updated.
There are still 15.7 MB of archives to be downloaded out
of 15.8 MB. After this operation, 70.9 MB of additional
disk space will be used.
Do you want to continue? [Y/n] y
```

During the installation, two dialogs will appear.

In the first one, at the beginning, we select *"apache2"* and confirm with *"ok"*.
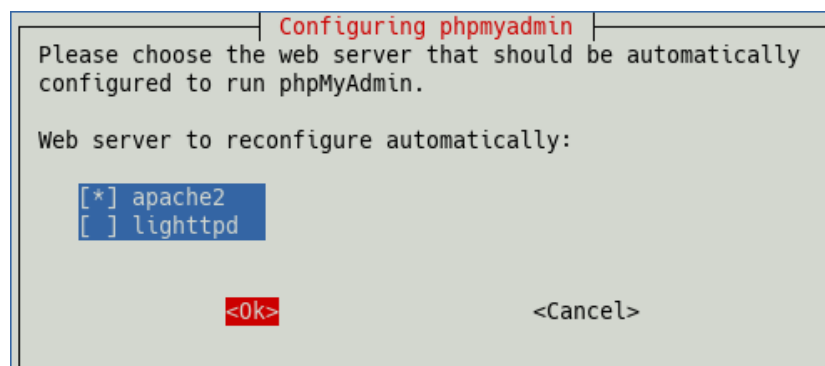
In the second one, at the end of the installation, we select *"yes"*.

```
┤ Configuring phpmyadmin ├

The phpmyadmin package must have a database installed and
configured before it can be used. This can be optionally handled
with dbconfig-common.

If you are an advanced database administrator and know that you
want to perform this configuration manually, or if your database
has already been installed and configured, you should refuse this
option. Details on what needs to be done should most likely be
provided in /usr/share/doc/phpmyadmin.

Otherwise, you should probably choose this option.

Configure database for phpmyadmin with dbconfig-common?

                    <Yes>                        <No>
```

In the following dialogs we need the password for the database user **phpmyadmin** (see the chapter *Preparations*).

5.6.5 Other software

If you are interested in developing websites, you can install a CMS for example, *WordPress*, *Drupal*, or *Joomla*, but you should consider our manual pages LAMP-Apache and LAMP-MariaDb for the configuration of the server and MariaDb beforehand.

5.6.6 Status data log files

Apache

The configuration status of the Apache web server can be output with `apache2ctl -S`.

The output shows the status without any changes to the configuration immediately after installation.

```
# apache2ctl -S
AH00558: apache2: Could not reliably determine the server's
fully qualified domain name, using 127.0.1.1. Set the '↙
   ServerName' directive
```

```
directive globally to suppress this message
VirtualHost configuration:
[::1]:80     127.0.0.1 (/etc/apache2/sites-enabled/000-↙
   default.conf:1)
127.0.0.1:80 127.0.0.1 (/etc/apache2/sites-enabled/000-↙
   default.conf:1)
ServerRoot: "/etc/apache2"
Main DocumentRoot: "/var/www/html"
Main ErrorLog: "/var/log/apache2/error.log"
Mutex default: dir="/var/run/apache2/" mechanism=default
Mutex mpm-accept: using_defaults
Mutex watchdog-callback: using_defaults
PidFile: "/var/run/apache2/apache2.pid"
Define: DUMP_VHOSTS
Define: DUMP_RUN_CFG
User: name="www-data" id=33
Group: name="www-data" id=33
```

The manual page LAMP-Apache contains a number of hints for customizing the configuration.

The directory `/var/log/apache2/` contains the log files. A look into them is helpful to identify error causes.

MariaDB

In the console, the command

```
# systemctl status mariadb.service
```

shows the current status of MariaDB and the last ten log entries.

The last twenty lines of the systemd journal are shown by

```
# journctl -n 20 -u mariadb.service
```

The command

```
# journctl -f -u mariadb.service
```

keeps the connection to the journal open and continuously shows the new entries. For more information, see the manual page LAMP-MariaDB.

PHP

The Apache server stores the error messages of PHP in its log files under `/var/⤸ log/apache2/`. Erroneous PHP functions generate a message on the called web page.
This behavior can be configured in the `php.ini` files of the respective interface. See the LAMP-PHP manual page.

5.6.7   Troubleshooting

The examples listed here show some troubleshooting possibilities.

File right in "DocumentRoot"

If calling the files `index.html` and `info.php` fails immediately after installation, please be sure to check the ownership and group membership of the web page directory first and change them if necessary:

```
# ls -la /var/www/html
drwxr-xr-x 2 www-data www-data 4096 14 Dec 18:56 .
drwxr-xr-x 3 root root 4096 14 Dec 18:30 ...
-rw-r--r-- 1 www-data www-data 10701 14 Dec 19:04 index.html
-rw-r--r-- 1 root root 20 14 Dec 19:32 info.php
```

In this case the Apache test page is displayed, the PHP status page is not. Then, the spirited use of

```
# chown -R www-data:www-data /var/www/html
```

will help. Now you should be able to call both pages.

HTML page load error

The web page **http://localhost/index.html** is not displayed and the browser reports a page load error.

We query the status of the Apache web server:

```
# systemctl status apache2.service
apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/system/system/apache2.service
             enabled; vendor preset: enabled)
     Active: failed (Result: exit-code) since
             Mon 2020-12-14 18:29:23 CET; 13min ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 4420 ExecStart=/usr/sbin/apachectl start
             (code=exited, status=1/FAILURE)

Dec 14 18:29:23 lap1 systemd[1]: Starting The Apache HTTP ⤶
    Server...
Dec 14 18:29:23 lap1 apachectl[4423]: AH00526: Syntax error ⤶
    on line 63 of /etc/apache2/conf-enabled/security.conf:
[...]
```

We see that the file `security.conf` has an error in line 63.
We edit the file and try again.

```
# systemctl start apache2.service
# systemctl status apache2.service
apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/system/system/apache2.service
             enabled; vendor preset: enabled)
     Active: active (running) since
             Mon 2020-12-14 18:34:59 CET; 3s ago
[...]
```

In general, a reload or restart of Apache is necessary after each configuration change.

Check Apache log files

A look into the log files under `/var/log/apache2/` helps to detect errors in the configuration of the network or the Apache server.

PHP, info.php only white page

This means that PHP is active but cannot display the page.
Please check:

- The content of the `info.php` file must be exactly the same as the example given in the PHP chapter.

- Check the file permissions as explained at the beginning of the Troubleshooting chapter and change them if necessary.

- If additional PHP modules have been installed or the configuration has been changed and the web server has not been restarted, this might help:

  ```
  # systemctl restart apache2.service
  ```

phpMyAdmin - Error

The call of http://localhost/phpmyadmin fails with the message "*phpMyAdmin - Error*" and the following information is displayed.

```
Error during session start; please check your PHP and/or
webserver log file and configure your PHP installation
properly. Also ensure that cookies are enabled in your
browser.

session_start(): open(SESSION_FILE, O_RDWR) failed: ⤢
   Permission denied (13)
session_start(): Failed to read session data: files (path: /⤢
   var/lib/php/sessions)
```

Check the permissions for the `/var/lib/php/sessions/` folder:

```
# ls -l /var/lib/php/
```

The output should contain this line:

```
drwx-wx-wt 2 root root 4096 14 Dec 17:32 sessions
```

Note the sticky bit (*"t"*) and the owner *"root.root"*. If there are any discrepancies, we will fix the error.

```
# chmod 1733 /var/lib/php/sessions
# chown root:root /var/lib/php/sessions
```

Now the login to *phpmyadmin* is possible.

5.6.7.1 If nothing helps Installing the LAMP stack takes less than fifteen minutes. However, troubleshooting can take hours.

Therefore, if the previously mentioned measures do not lead to a solution, it makes sense to remove the LAMP stack or parts of it and reinstall it. If, as mentioned in the chapter *Preparations*, apt has been cleaned up, the command **apt purge** helps to remove the previously installed packages with their configuration files without disturbing any other packages.

Here is an example with Apache:

```
# apt purge apache2
Package lists are read... Done
Dependency tree is built.
Status information is read.... Done
The following packages were installed automatically and are
no longer needed:
apache2-data apache2-utils
Use "apt autoremove" to remove them.
The following packages are REMOVED:
  apache2*
0 updated, 0 reinstalled, 1 to remove, and 0 not updated.
```

apache2 is removed and the packages *apache2-data* and *apache2-utils* have still remained.

Now please **don't use apt autoremove** because then the configuration files, where the error may be, will be left behind.

We use the command **apt purge**.

```
# apt purge apache2-data apache2-utils
```

If necessary, we do the same with the other parts of the program. Then we start a new attempt.

5.6.8   Security

The installation explained so far leads to a web server that is **"open like a barn door for everyone"**. Therefore, it should only be used standalone at a workstation and not connected to the private network and in no case to the Internet.

For securing the server, please read the manual pages
LAMP-Apache, LAMP-MariaDB, LAMP-PHP
regarding the configuration.

After that, exclusively for system and software updates, the server can be temporarily connected to the Internet via a second network interface.

Last edited: 2022/04/03

5.7 Set up Apache

This manual page is based on Apache 2.4.46.

According to our example from the installation guide, we want to set up a *LAMP test server for developers* that is directly connected to the workstation PC via LAN. Beyond that, for security reasons, there should be no connection for the server to a local network or even to the Internet.
The only exception is that the server will be connected to the Internet temporarily and exclusively for system and software updates via a second network interface.

5.7.1 Apache in the file system

Debian has fully integrated the Apache files into the file system according to their function:

- the executable program *apache2* into `/usr/sbin/`
- the installed modules for Apache into `/usr/lib/apache2/modules/`
- files that are also available to other programs into `/usr/share/apache2/`
- the configuration directories and files into `/etc/apache2/`
- the web page created by the user into `/var/www/html/`
- system files required at runtime into `/run/apache2/`, `/run/lock/apache2↙/`
- various log files into `/var/log/apache2/`

It is important to distinguish between the variables `ServerRoot` and `DocumentRoot`.

ServerRoot is the configuration directory, i.e. `/etc/apache2/`.
DocumentRoot contains the web page data, so `/var/www/html/`.

5.7.2 Connection to the server

The connection between the test server and the PC is placed in the IPv4 network segment **192.168.3.xxx**, while the PC's Internet connection is outside this network segment. The data used are:

server

IP: 192.168.3.1/24

name: server1.org

alias: www.server1.org

PC

IP: 192.168.3.10/24

name: pc1

We make a backup copy of the `*/etc/hosts*` file on the server and on the PC and add the necessary lines to both.

- server `*/etc/hosts*`:

```
cp /etc/hosts /etc/hosts_$(date +%f)
echo "192.168.3.1 server1.org www.server1.org" >> /etc/↵
    hosts
echo "192.168.3.10 pc1" >> /etc/hosts
```

- PC `*/etc/hosts*`:

```
cp /etc/hosts /etc/hosts_$(date +%f)
echo "192.168.3.1 server1.org www.server1.org" >> /etc/↵
    hosts
```

Next, in *NetworkManager*, we enter the data for the server in the fields outlined in red. The method is changed from *"Automatic (DHCP)"* to *"Manual"* and in the address fields we enter the values mentioned at the beginning.
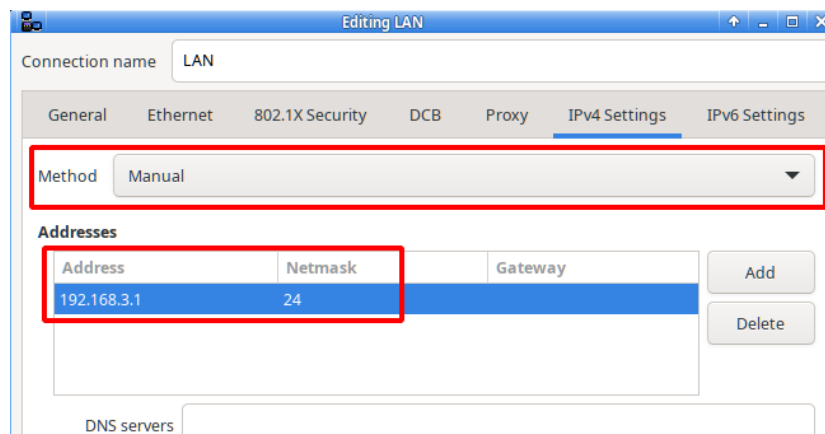
Additionally, in the tab *"General"*, the option *"Automatically connect with priority"* should be activated.

Accordingly, we set the appropriate settings on the PC for the used LAN interface.

On the PC, we test the connection in the console with

```
$ ping -c3 www.server1.org
```

and, if successful, we immediately check the function of Apache by entering http://www.server1.org/index.html in the address line of the web browser.

The Apache welcome page with *"It works!"* should appear.

5.7.3 Apache configuration

The configuration files and directories are located in the `ServerRoot` directory `/↲ etc/apache2/`.

The central configuration file is `apache2.conf`. It is usually not edited because many configurations are in separate files. Activation and deactivation is done via sym links. This has the advantage that a number of different configurations are available and only the required ones are included.

The configuration files are text files, which are created or edited with an editor and **root** permissions. The name of the file may be arbitrary, but the file extension must be *".conf"*. The valid directives that may be used in the configuration files are described in detail in the Apache documentation.

The files are located in the directories

`/etc/apache2/conf-available`,
`/etc/apache2/mods-available`, and
`/etc/apache2/sites-available`.

Your activation links can be found in

`/etc/apache2/conf-enable`,
`/etc/apache2/mods-enable`, and
`/etc/apache2/sites-enable`.

To enable or disable a *".conf"* file, we use `a2enconf` and `a2disconf`. This creates or removes the activation links. The command

```
a2enconf NAME_OF_FILE.conf
```

activates the configuration. Deactivation is done accordingly with:

```
a2disconf NAME_OF_FILE.conf
```

We proceed in the same way for modules and virtual hosts with the commands `a2enmod`, `a2ensite` and `a2dismod`, `a2dissite`.

The Apache web server reads the changed configuration with the command

```
systemctl reload apache2.service
```

Now we return to our *LAMP test server for developers* and adjust the configuration to the server data.

1. `/etc/apache2/apache2.conf` file

   It is one of the few exceptions for editing the `apache2.conf`. We add the following line at the beginning of the *"Global configuration"* section:

   ```
   ServerName 192.168.3.1
   ```

This tells the Apache web server the IP address where the development project should be reachable and suppresses redirections to IP 127.0.1.1 with error messages.

2. New `sites` file

   With the text editor of our choice, we create the file
   `/etc/apache2/sites-available/server1.conf`, e.g.

   ```
   mcedit /etc/apache2/sites-available/server1.conf
   ```

   Then we insert the following content, save the file, and exit the editor.

   ```
   <VirtualHost *:80>
   ServerName server1.org
   ServerAlias www.server1.org
   ServerAdmin webmaster@localhost
   DocumentRoot /var/www/html
   ErrorLog ${APACHE_LOG_DIR}/error_server1.log
   CustomLog ${APACHE_LOG_DIR}/access_server1.log combined
   </VirtualHost>
   ```

   Then we change the configuration to the new *VirtualHost* and announce the changes to the Apache web server:

```
# a2ensite server1.conf
   Enabling site server1.
[...]


# a2dissite 000-default.conf
   Site 000-default disabled.
[...]


systemctl reload apache2.service
```

5.7.4   Users and permissions

The Apache web server runs with the USER.GROUP *www-data.www-data*" and `DocumentRoot` belongs to *"root.root"* immediately after installation.

To give users write permissions to the files contained in `DocumentRoot`, a new group should be created specifically for this purpose. It does not make sense to use the existing group *www-data* because Apache runs with the rights of this group. We name the new group `work`.

With CMS

If a content management system (software for collaborative editing of website content) is added, we prepare `DocumentRoot` accordingly:

1. Create the group and assign it to the user.

   ```
   groupadd work
   adduser USERNAME work
   chgrp work /var/www/html
   ```

To activate the new permissions you have to log out and log in again, or use the `newgrp` command as user.

```
$ newgrp work
```

2. Set SGID bit for `DocumentRoot`
   so that all added directories and files inherit the group `work`.

   ```
   chmod g+s /var/www/html
   ```

3. Adjust owner and file permissions
   so that unauthorized people don't get access and the Apache web server runs properly.
   Let's look at the current permissions:

   ```
   # ls -la /var/www/html
   total 24
   drwxr-sr-x 2 root work 4096 Jan 9 19:32 .
   ```

```
            (DocumentRoot with SGID bit)
drwxr-xr-x 3 root root 4096 Jan 9, 19:04 ...
            (The parent directory /var/www)
-rw-r--r-- 1 root work 10701 9 Jan 19:04 index.html
-rw-r--r-- 1 root work 20 Jan 9, 19:32 info.php
```

For `DocumentRoot` we change the owner to *"www-data"*, give write permission to the group, and revoke read permission from everyone else as well (all recursively).

```
chown -R www-data /var/www/html
chmod -R g+w /var/www/html
chmod -R o-r /var/www/html
```

We check the result again.

```
# ls -la /var/www/html
total 24
dr-xrws--x 2 www-data work 4096 Jan 9 19:32 .
drwxr-xr-x 3 root root 4096 Jan 9 19:04 ...
-rw-rw---- 1 www-data work 10701 9 Jan 19:04 index.html
-rw-rw---- 1 www-data work 20 9 Jan 19:32 info.php
```

Now only members of the group *work* have write permission in `DocumentRoot`. Apache web server can read and write the files, all others are denied access.

4. Disadvantages of these settings

When creating new directories and files below `DocumentRoot`, the owner is the respective **user** and not *www-data*. This prevents the Apache web server from reading the files.
The solution is a *Systemd Path Unit*, which monitors changes below `DocumentRoot` and adjusts the owner and file permissions. (See the example in the Systemd-Path manual page.)

Without CMS

For static websites, a content management system is often not necessary and only constitutes another security risk and increased maintenance effort. In addition to the settings made before, the write permission to `DocumentRoot` can be revoked from the Apache web server to strengthen security because in case an attacker finds a hole in Apache, this will not give him write permission to `DocumentRoot`.

```
chmod -R u-w /var/www/html
```

5.7.5   Security - Apache Standard

Important safeguards are already included in the file `/etc/apache2/apache2.⤸ conf` by default.

The following three directives prevent access to the root file system and then release the two directories used by the Apache web server, "/usr/share" and "/var/www".

```
<Directory />
  Options FollowSymLinks
  AllowOverride None
  Require all denied
</directory>

<Directory /usr/share>
  AllowOverride None
  Require all granted
</directory>

<Directory /var/www/>
  Options Indexes FollowSymLinks
  AllowOverride None
  Require all granted
</directory>
```

The options *"FollowSymLinks"* and *"Indexes"* constitute a security risk and should be changed unless absolutely necessary. See below.

The following directive disables the display of the files `.htaccess` and *.⤸* `htpasswd`.

```
<FilesMatch "^\.ht">
   Require all denied
</FilesMatch>
```

5.7.6 Security - other configurations

- In the file **/etc/apache2/apache2.conf**:

  FollowSymLinks may cause content outside `DocumentRoot` to be listed.
  Indexes lists the contents of a directory if there is no `index.html` or `index`⤸
  `.php`, etc..

  It is recommended to remove *"FollowSymLinks"* and place all the project
  data below *"DocumentRoot"*. For the *"Indexes"* option, the entry has to be
  changed to

  ```
  Options -Indexes
  ```

  if the display of the directory contents is **not** desired.
  Alternatively, create an empty *index* file in the directory that is delivered to
  the client in place of the directory contents. For example, for the `upload`⤸
   directory:

  ```
  $ echo "<!DOCTYPE html>" > /var/www/html/upload/index.⤸
     html
        or
  $ echo "<?php" > /var/www/html/upload/index.php
  ```

- In the host configuration
  `/etc/apache2/sites-available/server1.conf`

  we can use the `<Directory>` block to block all IP addresses except those
  listed in it.

```
<Directory "/var/www/html">
    Order deny,allow
    Deny from all
    Allow from 192.168.3.10
    Allow from 192.168.3.1
</directory>
```

- **Merging** the configuration:

  The directives of the configuration are spread over a number of files within `ServerRoot` and the `.htaccess` files in `DocumentRoot`. It is therefore particularly important to know where to place the directive to achieve the desired effect.
  We strongly recommend to consult the web page
  apache.org - How the sections are merged
  intensively.

- The **owner** of `DocumentRoot`

  is *"root.root"* after installation and should be changed. See the chapter users and permissions.

5.7.7   Use HTTPS

Without HTTPS no website project can be launched today.
How to obtain a certificate is described, for example, in detail and in an easy-to-understand manner on the website wpbeginner.

First we create the necessary folders inside `DocumentRoot`:

```
cd /etc/apache2/
/etc/apache2/# mkdir ssl ssl/certs ssl/private
```

In these we put the certificate file `server1.org.crt` and the private key `server1↙.org.key`.

Then we secure the directories against unauthorized access.

```
/etc/apache2/# chown -R root.root ssl
/etc/apache2/# chmod -R o-rwx ssl
/etc/apache2/# chmod -R g-rwx ssl
/etc/apache2/# chmod u-w ssl/certs/server1.org.crt
/etc/apache2/# chmod u-w ssl/private/server1.org.key
```

Finally, we use the `ls` command to check:

```
/etc/apache2/# ls -la ssl
   total 20
   drwx------ 5 root root 4096 Jan 25 18:17 .
   drwxr-xr-x 9 root root 4096 Jan 25 18:43 ...
   drwx------ 2 root root 4096 Jan 25 18:16 certs
   drwx------ 2 root root 4096 Jan 25 18:16 private

/etc/apache2/# ls -l ssl/certs
   -r-------- 1 root root 1216 25 Jan 15:27 server1.org.crt
```

5.7.8 Security Tips

- The Apache documentation contains a recommended page with various security tips.
  apache.org - Security Tips

- In addition, there are numerous tips on the Internet for the secure operation of the Apache web server.

- Regular checking of the log files in `/var/log/apache2/` helps to detect errors or security holes.

- If the server is connected to the local network or to the Internet in a different way than intended in this manual page, a firewall is essential.

5.7.9   Integration in Apache2

The ssl module is activated in Apache by default. It is enough to edit the file `/↵ etc/apache2/sites-available/server1.conf`.

- Insert a new VirtualHost directive at the beginning. This redirects incoming client requests from port 80 to port 443 (ssl) using *"Redirect"*.

- Rewrite the previous VirtualHost directive to port 443.

- Add the SSL directives after the standard host directives.

- In case our web project should contain dynamically generated web pages, the last two *"FileMatch"* and *"Directory"* directives need to be inserted with the *"SSLOptions"* directive.

The extended `server1.conf` then has the following content:

```
<VirtualHost *:80>
    ServerName server1.org
    ServerAlias www.server1.org
    Redirect / https://server1.org/
</VirtualHost>

<VirtualHost *:443>
    ServerName server1.org
    ServerAlias www.server1.org
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error_server1.log
    CustomLog ${APACHE_LOG_DIR}/access_server1.log combined

    SSLEngine on
    SSLProtocol all -SSLv2 -SSLv3
    SSLCertificateFile /etc/apache2/ssl/certs/server1.org.crt
    SSLCertificateKeyFile /etc/apache2/ssl/private/server1.↵
        org.key

    <Directory "/var/www/html">
```

```
      Order deny,allow
      Deny from all
      Allow from 192.168.3.10
      Allow from 192.168.3.1
    </directory>

    <FilesMatch "\.(cgi|shtml|phtml|php)$">
      SSLOptions +StdEnvVars
    </FilesMatch>

    <Directory /usr/lib/cgi-bin>
      SSLOptions +StdEnvVars
    </Directory>
</VirtualHost>
```

In case our finished project is to be located at a hoster without access to `ServerRoot` (this is the rule), we can add a rewrite statement to the `.htaccess` file in `DocumentRoot` or create the file with the rewrite statement.

```
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteCond %{HTTPS} !=on
RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]
</IfModule>
```

5.7.10 Sources Apache

apache.org - Documentation
apache.org - Configuration files
apache.org - SSL Howto
Let's Encrypt - A nonprofit Certificate Authority

Last edited: 2022/04/03

5.8   Set up MariaDB

5.8.1   MariaDB in the file system

Debian has fully integrated the files of MariaDB into the file system according to their function:

- the executable program *mariadb* and the link *mysql* into `/usr/bin/`
    - (The latter points to `/usr/bin/mariadb`.)
- the installed plugin for MariaDB into `/usr/lib/mysql/plugin/`
- shared program parts and localizations into `/usr/share/mysql/`
- the configuration directories and files into `/etc/mysql/`
- the databases and log files into `/var/lib/mysql/`
- system files necessary at runtime into `/run/mysqld/`

The files inside the directories mentioned before should not be edited manually. The only exception is the configuration of MariaDB under `/etc/mysql/`, if you know exactly how to proceed. Otherwise use the MariaDB-CLI or a frontend like phpMyAdmin.

5.8.2   Initial configuration

After installation, as described in LAMP test server for developers, MariaDB is *'open like a barn door to anyone'*, because, by default, the two users **root** and **anonymous** are created (without password) as well as a test database.

Therefore we call the program `mysql_secure_installation` in the root terminal. Here we make quite a few settings to secure the database. The necessary entries are highlighted like this: `\<- - [ ]`.

```
# mysql_secure_installation

In order to log into MariaDB to secure it, we'll need the
current password for the root user. If you've just
installed MariaDB, and you haven't set the root password
yet, the password will be blank, so you should just press
enter here.
```

```
Enter current password for root:        «--[Enter]
OK, successfully used password, moving on...


Setting the root password or using the unix_socket ensures
that nobody can log into the MariaDB root user without
the proper authorisation.


You already have your root account protected,
so you can safely answer 'n'.


Switch to unix_socket authentication [Y/n]:   «--[n]
 ... skipping.


You already have your root account protected,
so you can safely answer 'n'.


Change the root password? [Y/n]:   «--[y]
New password:              «--[mein_mariadb_root_passwort]
Re-enter new password:    «--[mein_mariadb_root_passwort]
Password updated successfully!
Reloading privilege tables..
... Success!


By default, a MariaDB installation has an anonymous user,
allowing anyone to log into MariaDB without having to
have a user account created for them. This is intended
only for testing, and to make the installation go a bit
smoother. You should remove them before moving into a
production environment.


Remove anonymous users? [Y/n]:   «--[y]
... Success!


Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
```

```
the root password from the network.

Disallow root login remotely? [Y/n]   «--[y]
... Success!

By default, MariaDB comes with a database named 'test'
that anyone can access. This is also intended only for
testing, and should be removed before moving into a
production environment.

Remove test database and access to it? [Y/n]   «--[y]
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all
changes made so far will take effect immediately.

Reload privilege tables now? [Y/n]   «--[y]
... Success!

Cleaning up...

All done! If you've completed all of the above steps,
your MariaDB installation should now be secure.

Thanks for using MariaDB!
```

As a result, the user **root** has received a (hopefully secure) password and can no longer log in remotely. The user **anonymous** and the database *"Test"* have been removed.

5.8.3   MariaDB CLI

We reach the commandline interface in the terminal by typing **`mariadb -u \<`**↵
`user\> -p`. After entering the password, we see the greeting and the new prompt
`MariaDB [(none)]>`.

```
# mariadb -u root -p
Enter password:
Welcome to the MariaDB monitor.  [...]


MariaDB [(none)]>
```

For security reasons we only log in as **root** at the beginning to create the project
database, a user for everyday work on it, and a user to replace root.
Later in the phpMyAdmin section, we revoke the root user's all-encompassing priv-
ileges so that a potential attacker will be unsuccessful at this point.

Create a database

We are still logged into the terminal and create a new database for our project:

```
MariaDB [(none)]> CREATE DATABASE sidu;
Query OK, 1 row affected (0.002 sec)
```

That's all. If we want to delete this database, the required command is **`DROP`** ↵
`DATABASE sidu;`.

Create a user

First we create our project user with the name **tomtom** and assign him exclusively
all rights to the project database *"sidu"*:

```
MariaDB [(none)]> CREATE USER tomtom@localhost IDENTIFIED BY ↵
    '<enter a password for tomtom here>';
Query OK, 0 rows affected (0.002 sec)


MariaDB [(none)]> GRANT ALL ON sidu.* TO tomtom@localhost;
Query OK, 0 rows affected (0.001 sec)
```

Now repeat the same procedure for the user **chef**, who should take over the task of root.

```
MariaDB [(none)]> CREATE USER chef@localhost IDENTIFIED BY '<↙
   enter a password for chef here>';
Query OK, 0 rows affected (0.002 sec)


MariaDB [(none)]> GRANT ALL ON *.* TO chef@localhost WITH ↙
   GRANT OPTION;
Query OK, 0 rows affected (0.001 sec)


MariaDB [(none)]> FLUSH PRIVILEGES;
```

The new users differ in their rights.

tomtom has all rights only for the database *"sidu"* (sidu.*).
chef has all rights to all databases (*.*) and users (WITH GRANT OPTION).

So the user **chef** can take over the function of the user **root**, and **tomtom** is the user for work on our project database.
The logout is done by: **\q**.

```
MariaDB [(none)]> \q
Bey
#
```

Queries

We look at the result in a terminal, this time as user **chef**.
First the users and then the existing databases:

```
MariaDB [(none)]> SELECT User,Host FROM mysql.user;
+-------------+-----------+
| User        | Host      |
+-------------+-----------+
| chef        | localhost |
| mariadb.sys | localhost |
| mysql       | localhost |
```

```
| phpmyadmin  | localhost |
| root        | localhost |
| tomtom      | localhost |
+-------------+-----------+
6 rows in set (0.002 sec)


MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| phpmyadmin         |
| sidu               |
+--------------------+
5 rows in set (0.001 sec)
```

If we log out of MariaDB and log back in as user **tomtom**, the two queries look like this:

```
MariaDB [(none)]> SELECT User,Host FROM mysql.user;
ERROR 1142 (42000): SELECT command denied to user 'tomtom'@'↙
   localhost' for table 'user'.


MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| sidu               |
+--------------------+
2 rows in set (0.001 sec)
```

It is easy to see that the user **tomtom** does not get access to system relevant data.

5.8.4 phpMyAdmin

As seen before, MariaDB can be administered completely via the command line. If you know the syntax, which requires profound knowledge, you will quickly get the desired result this way.

We use the programm `phpMyAdmin`, which is more suitable for less experienced users, and enter:

http://localhost/phpmyadmin/

into the browser address bar. If we have already gone through the configuration according to the manual page LAMP - Apache, the call is:

https://server1.org/phpmyadmin/

To remove the rights of the database admin **root**, as mentioned above, we use our new database admin **chef** with his password in the login window right away.
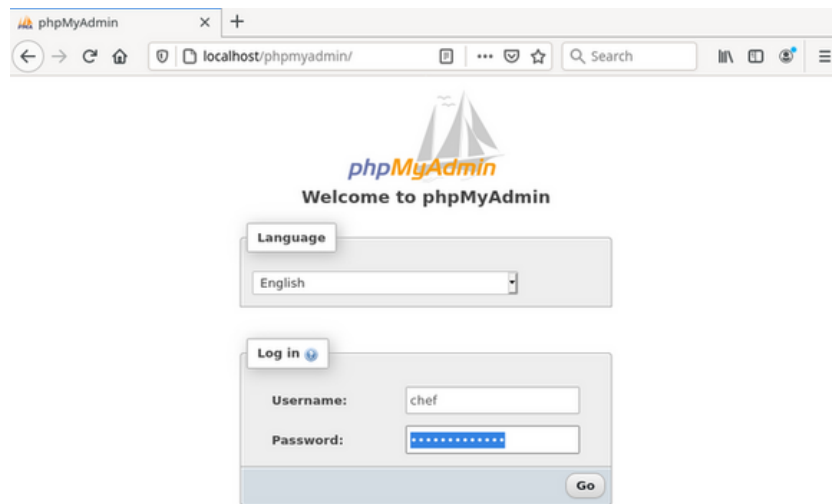
In the start window we see all databases in the left column. Then we select the tab *"User accounts"* in the center area.
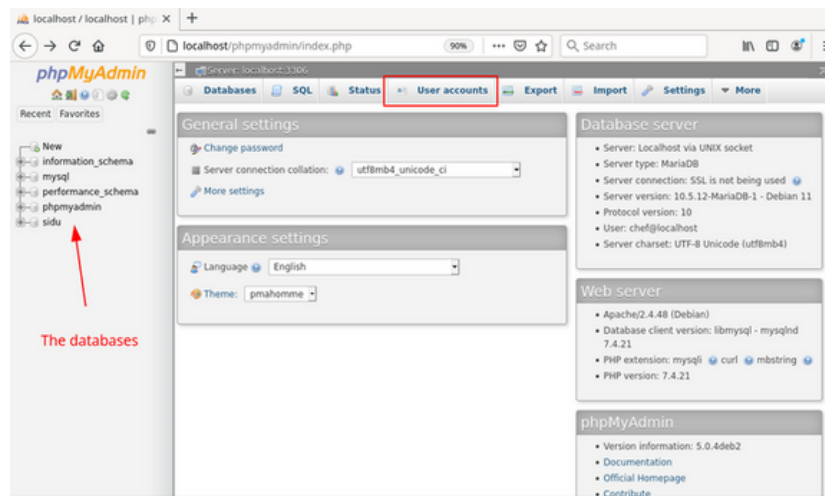
The user accounts overview shows all users and their rights in short form. Here we select the switch *"Edit privileges"* for the **root** user.
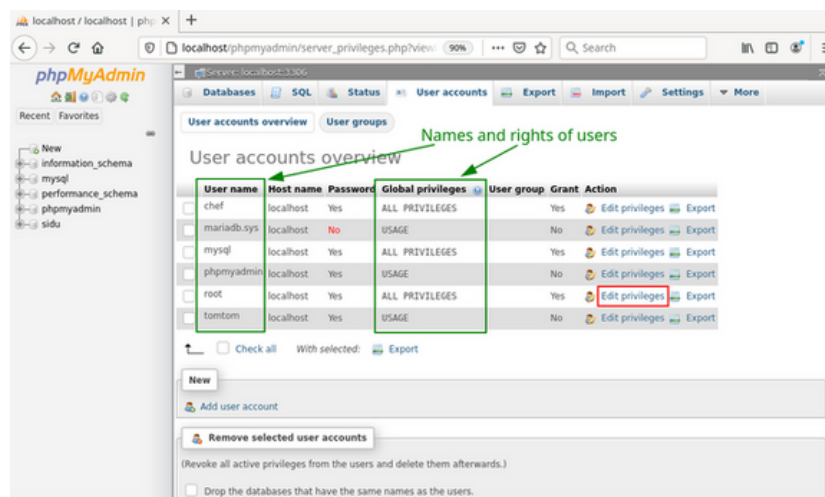
Now we see the detailed permissions for the **root** user. Here we first remove all his rights (1a), then, in the area *"Administration"*, grant the right *"Super"* (1b), and execute the action by clicking the *"OK"* button at the very bottom right of this page (not visible in the screenshot).
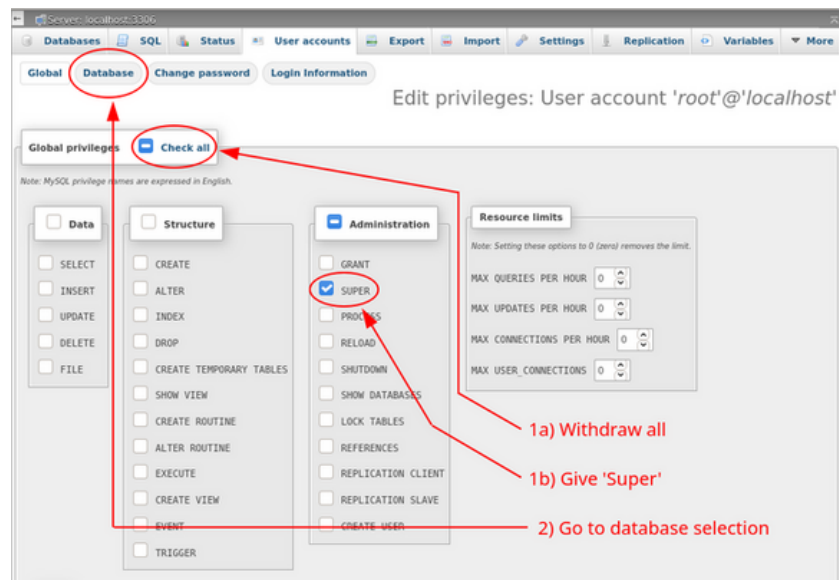
Afterwards we go to the next page via the *"Database"* button (2).
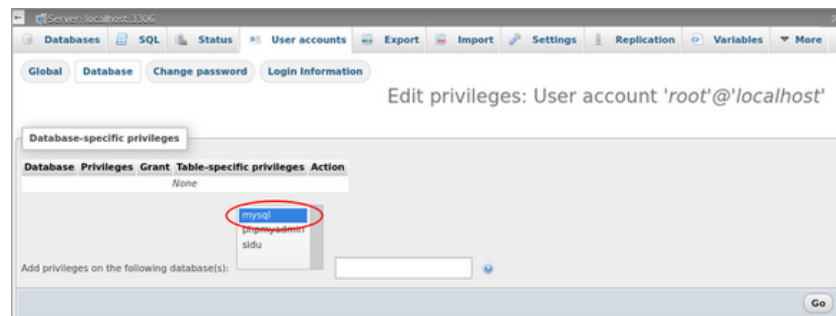
After selecting the database *"mysql"* and *"OK"*, a window opens with the detailed rights to the database *"mysql"* for the user **root**.
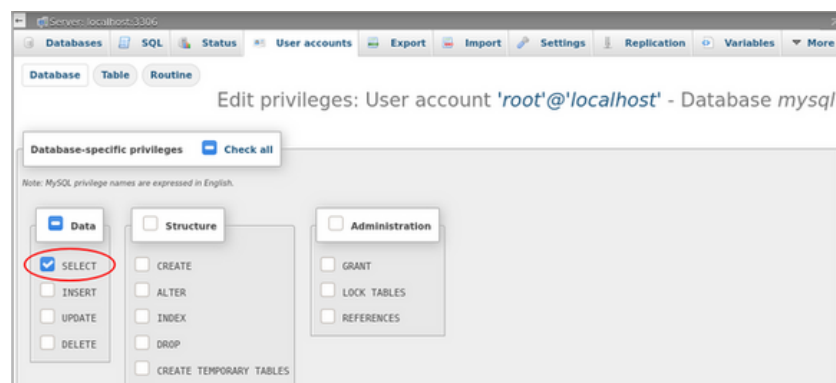
Only choose the method *"SELECT"*. A click on *"OK"* executes the sql command.

So we are done and leave *phpMyAdmin* via the door icon placed in the left column.
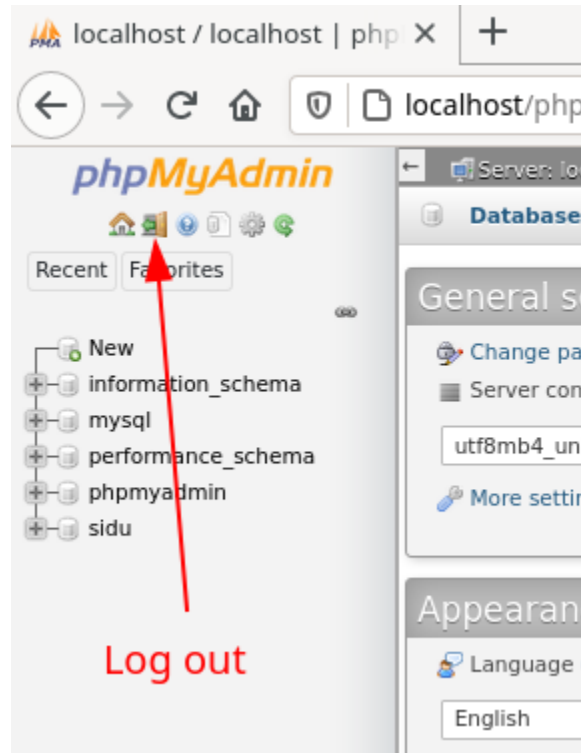
phpMyAdmin offers extensive possibilities for the administration of databases, their tables, and their contents. Note the *"Export"* tab in the main window, behind which you will find the option to backup data.

5.8.5 Integration in Systemd

The control of MariaDB has been integrated into Systemd in Debian, and thus also in siduction. MariaDB starts automatically when the server is booted. The control calls are:

```
# systemctl [start | stop | restart] mariadb.service
```

Startup and error messages of the server are logged in the systemd journal. Detailed information is available on the external web page MariaDB Systemd.

When searching the Internet for MariaDB's control panel, make sure that the search results refer to Systemd.

5.8.6   MariaDB Log

The Systemd Journal contains messages about the startup process of the `mariadb.service`. It is the first place to go to when errors occur.
In the console, the command `journalctl` displays the messages about MariaDB with:

```
journalctl -n 25 -u mariadb.service
```

(here the last 25 lines)

Or continuously with:

```
journalctl -f -u mariadb.service
```

In addition, you can switch on the logging of sql actions in the MariDB CLI like this:

```
MariaDB [(none)]> SET GLOBAL general_log=1;
```

This creates a log file with the pattern `<host>.log` in the directory `/var/lib/mysql/`.
Caution: This is an absolute performance killer and only meant to monitor actions in the short term.

5.8.7   Sources MariaDB

MariaDB Documentation
MariaDB Systemd

and the manpage

```
man mariadb
```

phpMyAdmin documentation

Last edited: 2022/04/04

5.9   Set up PHP

PHP is ready to use in siduction after installation with the default configuration.

5.9.1   PHP in the file system

Debian has fully integrated the files of PHP into the file system according to their function:

- the executable program `php7.x` and the link `php` into `/usr/bin/`
  - (The latter points to `/usr/bin/php7.x` via `/etc/alternatives/php`.)
- the installed modules into `/usr/lib/php/`
- shared program parts and modules into `/usr/share/php/` and `/usr/⤸ share/php<module>`
- the configuration directories and files into `/etc/php/`
- the current state of modules and sessions at runtime into `/var/lib/php/`

5.9.2   PHP support for Apache2

By default, the Apache web server loads support for PHP. We check this with the following command (replace the *"x"* with the minor attribute of the currently used PHP version, i.e. something like 7.4):

```
# ls /etc/apache2/mods-enabled/* | grep php
/etc/apache2/mods-enabled/php7.x.conf
/etc/apache2/mods-enabled/php7.x.load
```

We see that Apache has loaded the PHP module for version 7.x. To cause the PHP interpreter to process files with the extension "*.php*", the `DirectoryIndex` directive in the Apache configuration file `dir.conf` must contain the value `index.php`. We check this as well:

```
# cat /etc/apache2/mods-available/dir.conf
<IfModule mod_dir.c>
    DirectoryIndex index.html index.cgi index.pl index.php ⤸
        index.xhtml index.htm
</IfModule>
```

Nothing stands in the way of using PHP, because the value *"index.php"* is included.

5.9.3   PHP configuration

The directory `/etc/php/7.x/` contains the configuration sorted by the available interfaces.
The output shows the state after the initial installation.

```
# ls -l /etc/php/7.x/
total 20
drwxr-xr-x 3 root root 4096 18 Dec 16:54 apache2
drwxr-xr-x 3 root root 4096 18 Dec 16:54 cli
drwxr-xr-x 2 root root 4096 18 Dec 16:54 mods-available
```

With the modules *"php7.x-cgi"* and *"php7.x-fpm"* installed below, two new directories have been added.

```
# ls -l /etc/php/7.x/
total 20
drwxr-xr-x 3 root root 4096 18 Dec 16:54 apache2
drwxr-xr-x 3 root root 4096 1 Feb 21:23 cgi
drwxr-xr-x 3 root root 4096 18 Dec 16:54 cli
drwxr-xr-x 4 root root 4096 1 Feb 21:23 fpm
drwxr-xr-x 2 root root 4096 1 Feb 13:22 mods-available
```

Each of the `apache2`, `cgi`, `cli`, and `fpm` directories contains a `conf.d` folder and a `php.ini` file.
The respective *"php.ini"* contains the configuration for the corresponding interface and can be changed or supplemented if necessary. The *"conf.d"* folder contains the links to the activated modules.

5.9.4   PHP modules

Queries

A large number of modules are available for PHP. You can find out which ones are already installed with

---

```
# dpkg-query -f='${Status}\ ${Package}\n' -W php7.4* | grep ↙
    '^install'
install ok installed php7.4
install ok installed php7.4-bz2
install ok installed php7.4-cli
install ok installed php7.4-common
install ok installed php7.4-curl
install ok installed php7.4-gd
install ok installed php7.4-imagick
install ok installed php7.4-json
install ok installed php7.4-mbstring
install ok installed php7.4-mysql
install ok installed php7.4-opcache
install ok installed php7.4-readline
install ok installed php7.4-xml
install ok installed php7.4-zip
```

To show available but not installed modules, we change the end of the command
a bit:

```
# dpkg-query -f='${Status}\ ${Package}\n' -W php7.4* | grep '↙
    not-install'
unknown ok not-installed php7.4-calendar
unknown ok not-installed php7.4-cgi
unknown ok not-installed php7.4-ctype
unknown ok not-installed php7.4-dom
unknown ok not-installed php7.4-exif
unknown ok not-installed php7.4-ffi
unknown ok not-installed php7.4-fileinfo
unknown ok not-installed php7.4-fpm
unknown ok not-installed php7.4-ftp
unknown ok not-installed php7.4-gettext
unknown ok not-installed php7.4-iconv
unknown ok not-installed php7.4-pdo
unknown ok not-installed php7.4-pdo-mysql
```

```
unknown ok not-installed php7.4-phar
unknown ok not-installed php7.4-posix
unknown ok not-installed php7.4-shmop
unknown ok not-installed php7.4-simplexml
unknown ok not-installed php7.4-sockets
unknown ok not-installed php7.4-sysvmsg
unknown ok not-installed php7.4-sysvsem
unknown ok not-installed php7.4-sysvshm
unknown ok not-installed php7.4-tokenizer
unknown ok not-installed php7.4-xsl
```

Now we know the exact names of the modules.

Info

More detailed descriptions of the modules are provided by the command

```
# apt show <module_name>
```

Installation

To install modules we use e.g.:

```
# apt install php7.x-cgi php7.x-fpm
```

Both modules support CGI scripts and Fast/CGI requests.
Then we restart Apache:

```
# systemctl restart apache2.service
```

Handling

The state of PHP modules can be changed during runtime. This also allows controlling modules in scripts to load them before use and unload them afterwards.

- phpenmod - activates modules in PHP
- phpdismod - disables modules in PHP
- phpquery - shows the status of PHP modules

Unnecessary modules (imagick in the example) are deactivated in the console by the command

```
# phpdismod imagick
```

To load the imagick module for all interfaces, use the command

```
# phpenmod imagick
```

If we use the option `-s apache2`, e.g.:

```
# phpenmod -s apache2 imagick
```

the module will be loaded for Apache2 only.

The status query with `phpquery` always requires the module version and interface to be specified. Here are some examples:

```
# phpquery -v 7.4 -s apache2 -m zip
zip (Enabled for apache2 by maintainer script)

# phpquery -v 7.4 -s cli -m zip
zip (Enabled for cli by maintainer script)

# phpquery -v 7.4 -s fpm -m zip
zip (Enabled for fpm by maintainer script)

# phpquery -v 7.4 -s apache2 -m imagick
imagick (Enabled for apache2 by local administrator)
```

For the imagick module, the string *"Enabled for apache2 by local administrator"* tells us that it was not loaded automatically at startup like the zip module, but that the administrator has enabled it manually. The reason is the previously used `phpdismod` and `phpenmod` commands for this module.

5.9.5   Apache Log

The Apache server stores the error messages of PHP in its log files under `/var↙` `/log/apache2/`. At the same time, if PHP functions fail, a message appears on the called web page.

Alternatively, we can display the log functions.

```
# php --info | grep log
[...]
error_log          => no value
log_errors         => On
log_errors_max_len => 1024
mail.log           => no value
opcache.error_log  => no value
[...]
```

In the files `/etc/php/7.x/<Interface>/php.ini`, we have the possibility to re-place the unset values with our own, actually existing log files.

5.9.6   Sources PHP

PHP - manual

PHP - current messages

tecadmin - module handling

Last edited: 2022/04/04

6   Hardware

6.1   Graphics drivers

for nVidia, Intel, ATI/AMD

We only cover the most common graphics cards here in the manual. Exotic or relatively old graphics hardware, as well as server graphics are not discussed.

6.1.1   Open source Xorg driver

It is relatively easy to find out which graphics hardware is installed:

```
inxi -G
lspci | egrep -i "vga|3d|display"
```

This information is also very important if you have problems with the graphics and are looking for help in the forum or IRC.

The graphics system under Linux consists of 4 basic parts:

- kernel driver

  - radeon/amdgpu (ATI/AMD graphics)
  - i915 (Intel graphics)
  - nouveau (nVidia graphics)

- Direct Rendering Manager

  - libdrm-foo

- DDX driver

  - xserver-xorg-video-radeon/amdgpu
  - xserver-xorg-video-intel
  - xserver-xorg-video-nouveau

Xorg can also use modesetting-ddx, which is now part of the Xserver itself. This is automatically used for Intel graphics and is also used if no special xserver-xorgvideo-foo package is installed.

- dri/mesa

    – libgl1-mesa-glx
    – libgl1-mesa-dri
    – libgl1-mesa-drivers *This part of Xorg is the free OpenGL interface for Xorg.*

Open source Xorg drivers for nVidia (modesetting/nouveau), ATI/AMD (modesetting/radeon/amdgpu), Intel (modesetting/intel), and others are pre-installed with siduction.

Note: xorg.conf is usually no longer needed for open source drivers. Exceptions are e.g. multi-screen operation.

6.1.2   Proprietary drivers

Proprietary drivers are actually only available for nVidia graphics cards. AMD also has a proprietary driver called amdgpu-pro, but this only officially supports Ubuntu in certain versions and is not packaged in Debian. Also, this driver is designed for professional cards rather than desktop cards.

Here you can get more information about the drivers of
Intel
ATI/AMD
nouveau
X.Org.

6.1.3   Video driver 2D

Pretty much any video card that uses a KMS driver kernel-side is suitable for 2D operation under all surfaces. In general (with a few exceptions of exotic or old hardware), 3D acceleration is also available.

6.1.4   Video driver 3D

3D acceleration is available under Linux for Intel, AMD, and nVidia graphics cards. How well the free drivers have 3D implemented depends somewhat on the graph-

ics card itself. In general, it should be noted that almost all graphics cards require non-free firmware to run smoothly. This firmware is only available in the non-free repository in Debian because it is not DFSG compliant. If the correct firmware is installed, 3D support is available with Intel or AMD graphics cards without any further action. With nVidia graphics the story is a bit different. Older cards, which are classified as legacy cards by nVidia, work relatively well, although problems are always to be expected since the desktop used also plays a role. The free nouveau driver is developed without support from nVidia via reverse engineering.

Since the non-free firmware is usually required for correct operation (AMD, Intel from Skylake on, and Nvidia from Fermi on), an entry similar to

```
deb http://deb.debian.org/debian/ unstable main contrib non-↙
    free
```

should be set. To prevent subsequent problems with WiFi, network, Bluetooth, or similar, a

```
apt update && apt install firmware-linux-nonfree
```

makes sense. This will install more firmwares than you might need, but that should not be a disadvantage.

6.1.5   nVidia closed source driver

Selection, installation with dkms support and integration in Xorg.

nVidia divides its graphics card drivers into 7 generations:

1. Riva TNT, TNT2, GeForce, and some GeForce 2000 GPUs.
2. GeForce 2000 to GeForce 4000 series GPUs
3. GeForce 5000 series GPUs
4. GeForce 6000 and 7000 series GPUs
5. GeForce 8000 and 9000 series GPUs
6. GeForce 400 and 500 series GPUs (Fermi GF1xx)

7. Geforce 600, 700, 800 (Kepler GK1xx GK2xx, Maxwell GM1xx GM2xx,);
    Geforce 10xx (Pascal GP1xx), Geforce 16xx/20xx (Turing TU1xx); Geforce
    30xx (Ampere GA1xx)

Cards of the generations 1 - 5 are no longer supported by nVidia, only old driver
versions are available, which neither work with current kernels nor with current versions of the Xorg server. For a complete and up-to-date list of supported graphics
chips, please consult the "Supported Products List" on the NVIDIA Linux graphics
driver download page.

Debian provides the following versions of the binary drivers:

```
- nvidia-legacy-304xx-driver (for 4.)
- nvidia-legacy-340xx-driver (for 5.)
- nvidia-legacy-390xx-driver (for 6.)
- nvidia-driver (for 7.)
```

Since these are proprietary drivers, contrib and non-free must be activated in
the sources (like for the firmware for free drivers). You have to make sure in advance that the kernel headers are installed to match the running kernel. This is
the case once linux-image-siduction-amd64 and linux-headers-siduction-amd64
are installed. In addition, the packages *gcc*, *make* and *dkms* are necessary. With
dkms additionally installed (nVidia) kernel modules are automatically updated during a kernel update. After you have found out which nVidia card or which nVidia
chip you have, you can install the driver as follows:

GeForce 8000 and 9000 series

```
apt update && apt install nvidia-legacy-340xx-driver
```

GeForce GF1xx Chipset, Fermi Cards

```
apt update && apt install nvidia-legacy-390xx-driver
```

Kepler, Maxwell, Pascal, and newer (GKxxx, GMxxx, GPxxx, TU1xx)

```
apt update && apt install nvidia-driver
```

If this runs without errors, enter

```
mkdir -p /etc/X11/xorg.conf.d; echo -e 'Section "Device"
\n\tIdentifier "My GPU"\n\tDriver "nvidia"\nEndSection'
 > /etc/X11/xorg.conf.d/20-nvidia.conf


(all in one line)
```

to tell Xorg to use this installed driver. After a reboot the system should hopefully boot up to the desktop. If problems occur, i.e. the desktop does not start, you should consult **/var/log/Xorg.0.log**.

Since the legacy drivers 304.xx and 340.xx are no longer supported by nVidia, it is likely that they will not work with a new kernel or new Xorg.

Notebooks with hybrid graphics Intel/nVidia, so-called Optimus hardware, are problematic. In the past, Bumblebee was recommended, but this solution is anything but optimal. nVidia itself recommends configuring these setups with PRIME. Our recommendation is to avoid such hardware if possible. We cannot provide setup tips for Optimus hardware here.

Last edited: 2022/02/16

7   System Administration

This section contains informations and notes on

- Terminal - command line, a basic introduction, working as root, configuring colors in a terminal, getting help in a terminal, and using scripts.

- Doas - Alternative to Sudo, our recommendation for all those who miss *sudo*.

- System administration in general. Short and sweet, we provide a stiff through system administration, boot options, managing systemd services, terminating processes, managing passwords, fonts in siduction, the printing system CUPS, and sound in siduction.

- Btrfs filesystem on siduction, the subvolumes after installation, creating and managing new subvolumes, snapshot in Btrfs.

- Btrfs Snapshots with snapper Creating and managing Btrfs snapshots with Snapper. Configuring Snapper and working with systemd. System rollback and recovery of files.

- APT package management, package sources, managing packages, updating the system, searching program packages, and why to use apt exclusively.

- Local APT mirror, apt-cacher, the proxy server for Debian packages, and how to install server as well as client configuration.

- Nala for package management, a front-end that optimizes and accelerates APT for the user.

- Installing new kernels, upgrading the kernel without a system upgrade, and removing 3rd party modules as well as old kernels.

- systemd - the system and services manager, the concept of systemd, unit types, systemd in the file system, and handling services.

  - The systemd unit file, directories and hirarchies of unit files, the incorporation in systemd, the structure of unit files with a description of numerous options, the function of unit files on the example of CUPS, and the tools that systemd provides.

- **–** systemd-service unit, creating a service unit, and the description of all essential options.

- **–** systemd-mount unit, contents of the mount unit, contents of the auto-mount unit, naming conventions, areas of use, and some examples.

- **–** systemd-target - target unit, from runlevel to systemd-target, special features to consider.

- **–** systemd-path unit, the required files, the options of path-unit, creating and including path-unit, and the example "Monitoring DocumentRoot of Apache web server".

- **–** systemd-timer unit, the required files, the options of the timer unit, creating as well as including timer units, and timer units as cron replacement.

- **–** systemd-boot, the boot manager of systemd. Special features, comparison with Grub, installation.

- systemd-journal, using journald locally and over a network, configuring journald, querying the systemd journal with journalctl, filtering and controlling the output, examples to master journalctl.

Last edited: 2024/09/21

7.1 Terminal - command line

A terminal, also called a console, is a program that allows you to interact directly with the GNU/Linux operating system through directly executed commands. The terminal, also often called the *"shell"* or *"command line"*, is an extremely powerful tool and it is well worth the effort to learn the basics of its use.

In siduction, you can invoke the terminal/console by clicking on the PC monitor icon to the right of the menu, or by going to *"Menu" > "System" > "Terminal"*, or, even easier, by typing *"kons"* or *"term"* into the menu search bar.

After calling the terminal, you will see the prompt:

```
username@hostname:~$
```

username in the above example corresponds to the username of the logged in user. The tilde ~ shows that you are in your home directory /home/username, and the dollar sign (the prompt) $ means you are logged into the terminal with limited user privileges. At the end, the cursor is blinking. All this is the command line. This is where you enter commands that you want the terminal to execute.

Many commands can only be executed with **root** rights, i.e. administrator rights. root rights can be obtained by typing **su** and pressing **Enter**. After that you have to enter the root password. The password is not displayed on the screen during the input. (See below work-as-root.)

If the input is correct, the command line now shows:

```
root@hostname:/home/username#
```

Note that the dollar sign $ has been replaced by a hash #. In a terminal, the hash # always means that you are logged in with **root** privileges.
When command line commands are specified in the manual, the information before the prompt ($ or #) is omitted. A command like:

```
# chmod g+w <file>
```

means: you open a terminal, log in as **root** (**su**) and execute the command in a root prompt #. The hash is not included.

Another note:

For users who are new to the terminal, it is often confusing if no message appears after executing a command, but only the empty prompt again. This is intentional and means that the command was executed without errors. (In the example above, the group members were given write permissions to the <file>.)

7.1.1   Work as root

> **Caution**
> While logged into the terminal with root privileges, you are allowed to do anything, e.g. delete files, without which the operating system will stop working, and so on. When working with root privileges, you must be aware of *what* you are doing because it is easily possible to cause irreparable damage to the operating system.

It must be taken into account that all actions, if provided for in the program, are also executed with **root** privileges. The simple copy command **cp <source> <⤶ destination>** in a user directory leads to files with the owner **root** in the destination directory. This is probably not intended and also not useful.

Therefore: **Work as root only where it is really necessary!**.

About su

A number of commands must be started with **root** privileges. These rights can be obtained by entering **su**. After entering the correct password the root prompt appears.

```
$ su
Password:
#
```

Now it is possible to execute all commands in the terminal and start all programs that require root privileges. You can exit this state by typing

```
# exit
$
```

and the prompt for the user appears again.

About su-to-root

In contrast to the general command su, `su-to-root` allows the execution of pro-
grams with graphical user interface with **root** privileges. `su-to-root` transfers X
properties to the target user using *su*. The command is:

```
su-to-root -X -c <program>
```

If error messages related to **dbus** occur, expand the input:

```
su-to-root -X -c 'dbus-launch <program>'
```

Another terminal opens, into which the root password is to be entered. If successful,
the desired program will launch with **root** privileges.
Examples of using graphical applications via su-to-root are: editing a configuration
file with a text editor, using the partition manager *gparted*, or using file managers
like *dolphin* or *thunar*.

Use in desktop environments:

- Plasma (KDE and LXQt)
  The command is not necessary in Plasma and is not supported. For programs
  that need **root** privileges, a password prompt occurs and for the editor the
  prompt occurs when you want to save the changed file. Therefore only use
  `su` in the terminal, if necessary.
- Gnome and Cinnamon
  The behavior is similar to Plasma, except that the command (su-to-root) is
  supported, but not necessary.
- Xfce and Xorg
  Here the command unfolds its full power, and you are able to start the de-
  sired graphical program with **root** privileges. However, you are also in the

obligation to consider, when and with which program root rights are really necessary.

> Under no circumstances should productive programs that are normally started with user rights be booted as root with this option: Internet browsers, e-mail programs, office programs, etc.

sudo is not configured

sudo is only available in live mode, because no root password is set there.
After an installation sudo is not enabled. The reason is: If an attacker grabs the user password, he does not yet gain super-user privileges and cannot make any harmful changes to the system.
Another problem with sudo is that a root application running with the user configuration can change permissions and thus make them unusable for the user. The use of **su** or **su-to-root** is recommended!
If you want to use sudo despite all warnings, you have to add the corresponding $user to the sudo group!
This can be done with the command `adduser <USER> <GROUP>` as root.
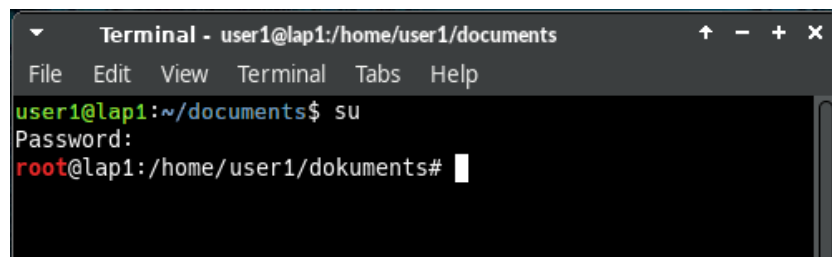
As a slim alternative to sudo we recommend `doas`. The manual page Doas - Alternative to Sudo explains the advantages of doas and the doas configuration.

7.1.2 Colored terminal

Colored prompts on the terminal can save you from unpleasant or catastrophic errors if you are **root #** and perform a task you meant to do as **user $**.
That's why in siduction, by default, the **user $**'s prompt is green, blue, and white, and **root #**'s prompt has the word "root" displayed in red.

The focus when working with the terminal should be on the input and output of the commands and not on colored prompts. In siduction we nevertheless decided to use the colors to give users a warning when they are system administrators with **root** privileges.

Change prompt color

Before changing the configuration file, we first create a backup copy in the terminal with a date stamp.

```
$ cp ~/.bashrc ~/.bashrc_$(date +%F)
```

Then we open the file `~/.bashrc` with a text editor of our choice (e.g.: kate, gedit, mcedit, vim, …) and look for the following line, which is located approximately in the middle of the file:

```
PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h↙
    \[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
```

The font and color code are immediately followed by the promt parts that are to receive this representation. The following figure shows the relation between the prompt parts and their abbreviations.
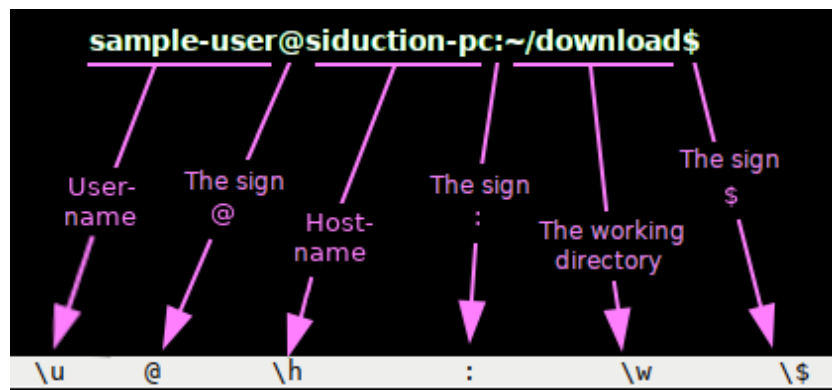
The following table explains the values of the syntax "[\033[**01;32**m]", where the bold part determines the font attributes and the color.

font code	font attribute	color code	color
00m	Default for font and color		
00;XX	default font	XX;m	default color
01;XX	bold	XX;30	black
02;XX	dark	XX;31	red
03;XX	italic	XX;32	green
04;XX	underlined	XX;33	yellow
05;XX / 06;XX	flashing	XX;34	blue
07;XX	block, inverted	XX;35	magenta
08;XX	background color (invisible)	XX;36	cyan
09;XX	strikethrough	XX;37	white

The "PS1" line quoted above is therefore displayed as follows:

font code	prompt parts and their display
[01;32m]\u@\h	user, @ and host get the attributes "bold" and "green"
[00m]:	colon gets the default attributes of the terminal
[01;34m]\w	the working directory gets the attributes "bold" and "blue"
[00m]$	the prompt gets the default attributes of the terminal

If the color is to be removed from the prompt, we prefix the PS1 line with a hash # and a **space**. This comments out the line. Now it is sufficient to add the line

```
PS1='${debian_chroot:+($debian_chroot)}\[\033[00m\]\u@\h:\w\$↙
    '
```

immediately as the next line.

If the color is to be changed in the prompt, the color coding must be adjusted for each part of the prompt.

```
PS1='${debian_chroot:+($debian_chroot)}\[\033[03;32m\]\u@\h↲
    \[\033[01;34m\]:\w\[\033[00m\]\$ '
```

This code example creates a prompt in which **username @ hostname** is green and italicized; the **:** and the **working directory** are blue and bold; the **$** character and the command prompt are given the contrasting color to the background of the terminal.

The new colors and formats appear after opening a new terminal.

Color settings of the terminal

In the terminal menu, under *"Edit" - "Settings…" -* tab *"Colors"*, there are a ton of setting options. We recommend a rather plain setting.

7.1.3   When the terminal hangs

Sometimes a terminal can no longer respond as desired. This is usually because a program has terminated with an error and left the terminal in an abnormal state. Then

```
reset
```

must be entered and the **Enter** key must be pressed.

If the output of a terminal appears distorted, this can usually be fixed by pressing **Ctrl**+**l**, this will rebuild the terminal window. Such distortions usually occur when working with programs that use an ncurses interface, for example *cgdisk*.

A terminal may appear frozen, but this is usually not the case; input continues to be processed even if it does not appear to be so. This can be caused by accidentally pressing **Ctrl**+**s**. In this case, **Ctrl**+**q** can be tried to free the console again.

7.1.4   Help in the terminal

Most commands/programs have command line help and also instructions. The are called *"man page"* or *"manual page"*. The syntax to call the man page is:

```
$ man <command>
```

or

```
$ man -k <keyword>
```

This calls the man page of a command. Navigation in the man pages is done by the arrow keys and they can be terminated with **q** for quit. Example:

```
$ man apt-get
```

To exit a man page, type **q**.

Another useful tool is the `apropos` command. apropos allows you to search the man pages for a command if, for example, you forget the syntax. Example:

```
$ apropos apt-
```

This lists all commands for the package manager apt. apropos is a powerful tool, for more in-depth information about apropos enter

```
$ man apropos
```

7.1.5   Linux console commands

A very good introduction to the BASH console can be found at linuxcommand.org. Of course you can also use your favorite search engine to find more.

Burning CD, DVD, and BD.

The command line programs are the basis for popular GUI programs like K3b, Brasero, or Xfburn.
Those who prefer the full range of options provided by the command line programs `cdrdao`, `wodim`, `growisofs`, etc. use the terminal. Our manual page Burn DVD without GUI contains many examples and tips to detect available hardware, compile data, and then burn it to CD, DVD, and BD.

7.1.6 Using scripts

A console script is a convenient way to bundle several commands into one file. Entering the filename of the script executes the commands that are in the script. siduction comes with some very useful scripts that provide simplifications to system administration.

A script is started in the console as follows, if you are in the same directory:

```
./name_of_script
```

Some scripts require **root** access, depending on the scope of the script.

Installation and execution

Use `wget` to load a script onto the machine. It is best to place it in the recommended directory, for example in `/usr/local/bin`. To copy and paste in the console, the mouse can also be used after gaining root privileges with `su`.

Example with wget and root privileges

```
$ su
password:
# cd /usr/local/bin
# wget -c ftp://<remote_server>/script-name.sh
```

After that, the file must be made executable:

```
# chmod +x script-name.sh
```

Since the directory `/usr/local/bin` is included in the search path of **root**, this simple command is sufficient to start the script:

```
# script-name.sh
```

The file can also be loaded onto the computer with a browser and moved to the appropriate location, but it must be made executable even then.

Example with wget as user

This is how to save a file in $HOME (the prompt is '$') as a user:

```
$ wget -c ftp://<remote_server>/user-script-name.sh
$ chmod +x user-script-name.sh
```

The script is started like this:

```
$ ./user-script-name.sh
```

Of course, this will only work as **user** if the script does not contain any commands that require **root** privileges.

Last edited: 2022/04/04

7.2 System administration in general

7.2.1 Boot options cheat codes

At the beginning of the boot process, the kernel command line can be edited by pressing the **e** key as soon as the Grub menu appears. In edit mode, use the arrow keys to navigate to the kernel line and insert the desired cheatcode(s) at the end. The space character serves as separator. To conitnue the boot process, enter **Ctrl**+**X**.

The following links lead to the manual page with the tables for the boot options.

1. siduction specific parameters (Live-CD only)
2. Bootoptions for graphics server X
3. General parameters of the linux kernel
4. Values for the general parameter **vga**

Detailed reference list for kernel bootcodes from kernel.org

7.2.2 systemd - managing services

systemd knows a total of 11 unit types. The units we deal with most often in everyday life are:

- systemd.service
- systemd.target
- systemd.device
- systemd.timer
- systemd.mount
- systemd.path

We briefly introduce some of the unit types here. Their names already give an indication of their intended functionality. More detailed explanations of the units can be found on our manual page System administration - systemd. The complete documentation can be found in the man pages `man systemd.unit`, `man systemd.↙ special`, and `man systemd.<unit_type>` respectively.

The systemd system can be controlled with follwing command, which requires **user** or **root** rights depending on the units:

```
systemctl [OPTIONS...] command [UNIT...]
```

`systemctl` knows autocompletion by **TAB** and the display of all variations by **TAB TAB**. Please read the man page `man systemctl`.

A list sorted by types with all active units or unit files can be output with the following commands:

```
$ systemctl list-units # for units
$ systemctl list-unit-files # for unit files
```

With the `-a` option all inactive units or unit files are also output.

7.2.3 systemd.service

To start or stop a .service unit, use the commands:

```
$ systemctl start <UNIT>.service
$ systemctl stop <UNIT>.service
$ systemctl restart <UNIT>.service
```

"*Restart*" is useful, for example, to notify the service of a changed configuration. If **root** privileges are required for the action, the root password is requested.
The command can also be used to terminate a service:

```
$ systemctl kill -s SIGSTOP --kill-who=control <UNIT>.service
```

With "*kill*", in contrast to "*stop*", the options `-s`, `--signal=`, and `--kill-who=` are available.

- "*-s*" sends one of the signals `SIGTERM`, `SIGINT`, or `SIGSTOP`. Default is "*SIGTERM*".
- "*–kill-who=*" allows selection of the processes within the hierarchy to which a signal should be sent. The options are `main`, `control`, or `all`. This sends the signal to the main process, the child processes, or both. Default is "*all*".

This behavior is similar to the old and still usable command pkill, which is explained below in the section Terminating a process.

7.2.4 systemd - UNIT inclusion

To have a (self-made) unit loaded automatically when the computer is booted, enter as **root**:

```
# systemctl enable <UNIT_file>
```

This creates a group of symlinks according to the requirements in the unit's configuration. Following this, the system manager configuration is automatically reloaded.

The command

```
# systemctl disable <UNIT_file>
```

removes the symlinks again.

Example

If a PC or laptop without Bluetooth hardware is in use, or you don't want to use Bluetooth, the command (as **root**)

```
# systemctl disable bluetooth.service
```

will remove the symlinks from all requirements and dependencies within systemd and the service will no longer be available and will not be started automatically.

7.2.5 systemd-target - formerly runlevel

Already since the 2013.2 "December" release, siduction has been using systemd as the default init system.

The old sysvinit commands are still supported. (for this a quote from man ↗ systemd: "… is provided for compatibility reasons and because it is easier to type.") More detailed information about systemd can be found on the manual page System administration - systemd.

The various runlevels that are booted or switched to are described by systemd as **target** units. They have the extension **.target**.

Target Unit	Description
emergency.target	starts into an emergency shell on the main console. It is the minimum version of a system boot to obtain an interactive shell. This unit can be used to guide the boot process step by step.
rescue.target	starts the base system (including system mounts) and an emergency shell. Compared to multi-user.target, this target could be considered as single-user.target.
multi-user.target	starts a multi-user system with a working network, without graphics server X. This unit is used when you want to stop X or to not boot into X. A system update (dist-upgrade) is performed on this unit .
graphical.target	starts multi-user mode with network capability and a running X Window System.
default.target	is the default unit that systemd starts at system startup. In siduction this is a symlink to graphical.target (except NoX).

A look into the documentation `man SYSTEMD.SPECIAL(7)` is mandatory to understand the relationships of the different *".target - units"*.

To switch to the system update runlevel, use the following command as **root** in the terminal:

```
# systemctl isolate multi-user.target
```

Important here is the *"isolate"* command, which ensures the termination of all processes and services that the selected unit does not request.

To shut down or restart the system, the command

```
# systemctl poweroff
   or
# systemctl reboot
```

can be used. *"poweroff"* or *"reboot"* (each without .target) are commands that starts several units in the correct order to terminate the system in an orderly fashion and to reboot if necessary.

7.2.6   Terminating a process

pgrep and pkill

Independently of systemd, `pgrep` and `pkill` are a very useful duo to terminate unwelcome processes. Run with **user** or **root** privileges in a console or TTY:

```
$ pgreg <tab> <tab>
```

The command lists all processes with their name, but without the process ID (PID). We use Firefox as an example in the following.
The `-l` option prints the PID and the full name:

```
$ pgrep -l firefox
4279 firefox-esr
```

To display subprocesses, if any, we also use the `-P` option and only the PID:

```
$ pgrep -l -P 4279
4387 WebExtensions
4455 file:// Content
231999 Web Content
```

then

```
$ pkill firefox-esr
```

terminates Firefox with the default signal *SIGTERM*.

With the option **–signal**, followed by the signal number or the signal name, *pkill* sends the desired signal to the process. A clear list of signals can be obtained with *kill -L*.

htop

Entered in the terminal, htop is a good alternative because a lot of useful information about the processes and the system load is presented. This includes a tree view, filter and search function, kill signal, and some more. The operation is self-explanatory.

Emergency exit

As a last resort before pulling the power plug, you can use the command **killall** ↙ **-9** in the terminal.

7.2.7   Forgotten root password

A forgotten root password cannot be recovered, but a new one can be set.

To do this, the live CD must first be booted.

The root partition must be mounted as **root** (e.g. as /dev/sdb2)

```
mount /dev/sdb2 /media/sdb2
```

Now enter the root partition with chroot (chroot = changed root) and define a new password:

```
chroot /media/sdb2 passwd
```

7.2.8   Setting new passwords

To change a user password, as **user** :

```
$ passwd
```

To change the root password, as **root** :

```
# passwd
```

To change a user password as administrator, as **root** :

```
# passwd <user>
```

7.2.9   Fonts in siduction

To improve the display of fonts, if necessary, it is important to check the correct settings and configurations of the hardware beforehand.

Check settings

- **Correct graphics drivers**
  Some newer ATI and Nvidia graphics cards do not harmonize very well with the free Xorg drivers. The only reasonable solution in these cases is to install proprietary, non open source drivers. For legal reasons, siduction cannot pre-install these. Instructions for installing these drivers can be found on the Graphics Drivers page of the manual.

- **Correct screen resolutions and refresh rates**.
  First, it's a good idea to look at the manufacturer's technical documentation, either in print or online. Each monitor has its own perfect combination of settings. These DCC values are usually passed correctly to the operating system. Only sometimes it is necessary to intervene manually to overwrite the basic settings.

  To check which settings the X server is currently using, we use xrandr in the terminal:

  ```
  $ xrandr
  Screen 0: minimum 320 x 200, current 1680 x 1050,
  maximum 16384 x 16384
  ```

```
HDMI-1 disconnected
  (normal left inverted right x axis y axis)
HDMI-2 connected 1680x1050+0+0 (normal left
  inverted right x axis y axis)  474mm x 296mm

  1680x1050     59.95*+
  1280x1024     75.02    60.02
  1440x900      59.90
  1024x768      75.03    60.00
  800x600       75.00    60.32
  640x480       75.00    59.94
  720x400       70.08
DP-1 disconnected
  (normal left inverted right x axis y axis)
```

The value marked with "*" indicates the setting used,
1680 x 1050 pixels with a physical size of 474 x 296 mm. In addition, we calculate the actual resolution in px/inch (dpi) to get an indication of the settings for the fonts. With the values given above we get 90 dpi.
1680 Px x 25,4 mm/inch / 474 mm = 90 Px/inch (dpi)

- **Check**
  We use a folding rule or tape measure to determine the actual size of the monitor. The result should differ by less than three millimeters from the values output by xrandr.

Basic font configuration

siduction uses free fonts that have proven to be balanced in Debian. In the graphical user interface TTF or outline fonts are used. If own fonts are chosen, new configuration adjustments may have to be made to get the desired font appearance.

The system-wide basic configuration is done in the terminal as **root**, using:

```
# dpkg-reconfigure fontconfig-config
```

For the dialogs called, these settings have proven to be useful:

1. For screen display, please select the preferred method for font tuning.

   'autohinter´

2. Please select to what extent font hinting is applied by default.

   medium

3. The inclusion of the subpixel layer improves the text display on flat panel displays (LCD).

   automatic

4. By default, applications that support fontconfig use only outline fonts. Use bitmap fonts by default?

   no

Subsequently

```
# dpkg-reconfigure fontconfig
```

is necessary to rewrite the configuration.

Sometimes rebuilding the font cache is a solution (the first command is for saving data with a date appendix, the second command is to be entered without a line break, i.e. on one line):

```
# mv /etc/fonts/ /etc/fonts_$(date +%F)/

# apt-get install --reinstall --yes -o DPkg::Options::=
--force-confmiss -o DPkg::Options::=--force-confnew
 fontconfig fontconfig-config
```

7.2.10   User configuration

Display type, size, 4K display

It should be noted that each font has an ideal size range, so identical size settings do not necessarily lead to the same good result for each font.
The settings can be made conveniently in the graphical interface. They take effect

on the desktop immediately, applications have to be restarted to some extent. The list shows where in the menu the settings can be found.

- KDE Plasma

    – *"System Preferences" > "Fonts" > "Fonts"*
    – *"System Preferences" > "Display Setup" > "Display Setup" > "Global Scaling"*

- Gnome (Tweak Tool)
  "Applications" > "Optimizations" > "Fonts"

- Xfce
  "Preferences" > "Appearance" > tab: *"Fonts"*

Explanation of terms

"Edge smoothing / Antialising":
This is the brightness gradation of the neighboring pixels at the edges to reduce the staircase effect on curves. However, it causes some blurring of the characters.

"Subpixel rendering / color order / RGB":
This is an extension of antialising for LCD screens by additionally controlling the color components of a pixel.

"Hinting":
This is the adaptation (change) of the characters to the pixel grid of the screen. It reduces the need for antialiasing, but the font shape no longer conforms exactly to the specifications, unless the font developers have already incorporated hinting variations. For **4K** screens, hinting is usually not necessary.

"DPI value / scaling factor":
This setting allows a different DPI value or size for the fonts only. Here the display on a **4K** screen can be improved quickly. The following table illustrates the relationship between screen diagonal and DPI value for **4K** screens.

4K resolution: 3840 x 2160 (16:9)

Diagonal	X-axis	Y-axis	DPI
24 inch	531 mm	299 mm	184
27 inch	598 mm	336 mm	163
28 inch	620 mm	349 mm	157
32 inch	708 mm	398 mm	138
37 inch	819 mm	461 mm	119
42 inch	930 mm	523 mm	105

Accordingly, a scaling factor of 2.0 is required for 4k screens with a diagonal of 24 inches, and a scaling factor of 1.2 is required for screens with a diagonal of 37 inches in order to obtain approximately equal displays corresponding to SXGA or WSXGA screens with 90 DPI.

7.2.11   CUPS - the printing system

KDE has a large section on CUPS in the KDE help. Nevertheless, here is a guide on what to do if you have problems with CUPS after a full-upgrade. One of the known solutions is:

```
# modprobe lp
# echo lp >> /etc/modules
# apt purge cups
# apt install cups
        OR
# apt install cups printer-driver-gutenprint hplip
```

CUPS will now be restarted:

```
# systemctl restart cups.service
```

Afterwards open a web browser and type this into the address line:

http://localhost:631

A small problem occurs when CUPS opens the corresponding dialog box for legitimation. Occasionally, the user's own user name is already entered there and

the password is expected. However, entering the user password does not work. Nothing works. The solution is to change the user name to **root** and enter the root password.

The OpenPrinting database contains extensive information about various printers and their drivers. Drivers, specifications, and configuration tools are available. Samsung used to supply its own Linux drivers for its printers. After the printer division had been sold to HP, the download page has no longer been available, and HP unfortunately did not include the Samsung drivers in *"hplib"*. Currently, the package printer-driver-splix works best for Samsung printers and Samsung multifunction devices. CUPS is currently in transition and is moving towards printing without drivers via PWG - IPP Everywhere, see also debian - an introduction to IPP-Everywhere.

7.2.12   Sound in siduction

In older siduction installations, sound is disabled by default.

Most sound problems can be solved by clicking on the sound icon in the control bar, opening the mixer, and unchecking "mute", or using the appropriate slider. If the speaker icon is not present, a right click on the control bar is sufficient, then select

in KDE: *"Control Panel Options" > "Add Mini Programs…"*
in Xfce: *"Bar" > "Add new items…"*

and select the desired module.

KDE Plasma

A right click on the speaker icon in the control bar opens the sound output settings window. The user interface is self-explanatory.

Gnome

Right-clicking on the speaker icon in the control bar opens a drop-down menu that contains a slider for the volume.
Further settings are possible as follows:

Right-click on the desktop > *"Settings"* > *"Audio"*

Xfce Pulse Audio

The settings are made via the speaker icon (pulse audio module) in the control bar. Again, the user guidance is self-explanatory. If the icon is missing, you can quickly get started with a terminal by entering the command

```
$ pavucontrol
```

and configuring the settings in the appearing window.

Alsamixer

If you prefer alsamixer, you can find it in the alsa-utils package:

```
# apt update
# apt install alsa-utils
# exit
```

The desired sound settings are made as **user** from a terminal:

```
$ alsamixer
```

Last edited: 2022/04/05

7.3   Doas - Alternative to Sudo

We, the siduction team, have decided to use a real root account and have not configured Sudo. For users who are used to Sudo and don't want to do without its functionality, the slim alternative Doas is a good choice. Doas is tailored to desktop systems, having only about 1/100 of code lines in comparison to Sudo. With *siduction 2021.3 wintersky*, Doas is automatically installed in version 6.8.1-3, but is not yet fully configured.

7.3.1   Configure Doas

The only thing missing to be able to use Doas is the configuration file `/etc/doas`↙ `.conf`. It contains line-by-line rules that assign actions to a user. A `#` introduces comments. Doas reads the lines one after the other, executing the action of the last applicable rule. To understand the rules in the configuration file, there are a few things to keep in mind.
- Only actions for which at least one rule applies are executed. - By the fact that Doas evaluates the rules line by line one after the other, hierarchies can be built up. - For rules that contain commands with arguments, the arguments must be specified exactly and completely. - Rules with commands that require variable arguments are not possible. - Doas checks the syntax of the configuration file before executing the requested action. In case of incorrect rules, Doas outputs `doas:`↙ `syntax error at line 4` and then exits. The write access to the configuration file is then only possible with the **root** account.

The configuration is particularly simple if only one user account exists on the siduction system. A single line is sufficient to execute commands with root privileges using the prefix "doas".
Log in to a terminal as **root** and execute the following command, replacing "tux" with the name of your user account.

```
tux@sidu:~$ su
Password:
root@sidu:/home/tux# echo "permit keepenv nopass tux" > /etc/↙
   doas.conf
root@sidu:/home/tux# exit
```

```
tux@sidu:~$
```

The configuration line consists of:

The action *permit|deny* with

the option *keepenv* (this allows to start graphical programs like *gparted*),

the option *nopass|persist* (no password request|the one-time password entry remains valid for a limited time), and

the user *tux* to which the action is to be applied.

If the username stands alone, **tux** may execute commands as any user present on the system. The default is **root**. If the execution of the action is to be allowed only with the rights of a user other than **root**, the name must be specified within the rule (e.g. *tux as anne*). Instead of the user, a group (e.g. *:vboxusers*) can gain permissions by prepending a `:`.

7.3.2   Doas and multiple users

Example

On the workstation PC, in addition to **tux**, three other users named **anne**, **bob**, and **lisa** are allowed to log in.

Anne only wants to allow Bob to run two of her scripts from her **/home** directory. Anne has restrictively set the permissions on her scripts to 700.

Lisa is especially trustworthy, so she should be in charge of system upgrades.

Now, as user **tux**, we use Doas in a terminal to edit the configuration file.

```
tux@sidu:~$ doas mcedit /etc/doas.conf
```

We convert the previously mentioned permissions into rules and add some comments to the file.

```
# doas config file /etc/doas.conf

# tux gets root privileges
permit keepenv nopass tux
```

```
# bob may execute anne's script
permit bob as anne cmd /home/anne/bin/script1 args -n
permit bob as anne cmd /home/anne/bin/script2 args

# lisa may execute system upgrade
permit persist lisa cmd init
deny lisa cmd init args 1
deny lisa cmd init args 5
permit persist lisa cmd apt args update
permit persist lisa cmd apt args full-upgrade
```

Explanations

bob may execute the scripts *script1* and *script2* inside Anne's **/home/anne/bin** directory (the former exclusively with the argument *-n*, the latter must not be given any argument). Specifying *args* in the rule line for the *script2* without a following argument forces the file to be called without an argument and thus without potentially malicious code. **bob** must supply the username when calling scripts, using the *-u* option.

```
bob@sidu:~$ doas -u anne /home/anne/bin/script1 -n
doas (bob@sidu) password:


bob@sidu:~$
```

The script was executed without comment after Bob entered his user password.

To allow **Lisa** to perform the system upgrade, she should switch to *multi-user.target* (init 3) and perform a *systemctl reboot* (init 6) after completion. The rule line `permit persist lisa as root cmd init` without specifying *args* causes all other calls of init are allowd, except those that are prohibited by the following rules below. Therefore, she cannot go directly from the *multi-user.target* to the *graphical.target* (init 5). Here we see the structure of a hierarchy.

Notes

If you keep typing *sudo*, the line `alias sudo="doas"` in your *.bashrc* will help. Doas plays its decisive advantage where only one user is granted root rights by

doas. The above example with Lisa shows how extensive the configuration for a restricted rights assignment can become. Furthermore, a rule for a program call with variable arguments (e.g. *apt install <package name>*) is not possible.

Sources

man doas

man doas.conf

github, doas

DE: LinuxNews, Linux Rechtemanagement, sudo durch doas ersetzen

DE: LinuxUser 08.2021, Kleiner Bruder

Page last updated 2022/03/06

7.4 Btrfs

Btrfs is a modern copy-on-write (COW) file system for Linux.

siduction supports installation into a partition formatted with *Btrfs*. The release of 2022.12.0 enables you to manage snapshots of Btrfs with Snapper and to boot via Grub. The installer creates subvolumes within the selected partition for the root directory `@`, the user directories `@home` and `@root`, the directory `@var@log`, and a subvolume `@snapshots` for system snapshots.

Btrfs works well with SSDs and conventional hard disks. Its own built-in RAID mechanism (RAID 0, 1, and 10 are supported) works reliably even with disks of different sizes. Metadata and file data are handled differently by Btrfs. Usually, metadata is stored twice even with only one drive. If multiple drives are present, the administrator can set different RAID levels for the metadata and file data within the same file system.

Btrfs manages the data within the drives in subvolumes, superficially similarly to conventional partitions. It can take snapshots of the subvolumes, which can be used for data reconstruction if needed. A mounted Btrfs file system behaves mostly like any other Linux file system. Occasionally, however, some differences come to light because Btrfs does most of its work in the background. For example, deleting a large file without immediately increasing the available free space causes confusion. Some time later, the missing space is there after all, or not if a previous snapshot references the file.

There is a lot of documentation about Btrfs on the Internet. We will therefore not repeat the extensive possibilities as well as the commands and their application here. Reading `man btrfs` and `man btrfs-<command>` is mandatory. In addition, we recommend the extensive kernel.org Wiki and the detailed documentation on readthedocs.io.

> **Please note**
> siduction does not support a separate boot partition when using the Btrfs file system.

The */boot* directory is an essential part of the operating system. With a separate partition, it would be excluded from system snapshots and thus a rollback would lead to errors.

Use Btrfs

For the advanced features of Btrfs (snapshots, compression, defragmentation, self-recovery for data and metadata, integrated volume management, …) e.g. compared to ext4, we need recognizably larger drives. Currently, this is usually not a problem, because even inexpensive PCs and laptop often have 500 GB drives.
As a minimum size of the Btrfs drive, into which the complete installation should take place, we recommend 100 GB. Depending on the volume of private data, this can be considerably more. If you want to use Btrfs only for the root partition, it should have a size of at least 50 GB. For users who do not want to allocate that much space, the usual approach is to use either Btrfs without snapshots or ext4. Btrfs understands arbitrary abbreviations on the command line for its commands and options, as long as they are unique. For example, **`btrfs su li /`** becomes **`btrfs subvolume list /`** internally.

7.4.1   Btrfs subvolume

During the first install to a single partition, the following subvolumes are created.

Subvolume	Mount point	Remarks
@	/	
@home	/home	
@root	/root	The **root** user
@var@log	/var/log	
@snapshots	/.snapshots	Snapshots of @ are stored here

For Btrfs, they are all located at the highest level (*top level 5*). It is also called *"flat layout"* because there are no nestings. The file system root itself is not mounted, but the *top level 5* subvolumes are. It is no longer necessary to mount the "root" device if only the contents of the subvolumes are of interest. During operation, we are already in the subvolume @.

The command **btrfs subvolume list /** prints all subvolumes of the file system root. The `-t` option creates a clearly arranged list.

```
# btrfs subvolume list -t /
ID   gen    top level  path
--   ---    ---------  ----
256  22981  5          @
257  22952  5          @root
258  22982  5          @daten
269  22972  5          @var@log
260  22967  5          @snapshots
```

The default subvolume.

In siduction the subvolume `@` should be set as default from the beginning, because in case of a rollback the command **snapper rollback <No>** is used. If no default subvolume was set before, Snapper will do it now with the rollback subvolume. This can lead to considerable confusion if the user in the console usually mounts the root partition using the device file and the command **mount -t btrfs /dev⤸ /sdxX /mnt/**. This is because after setting a default subvolume, the *top level 5* subvolumes are no longer accessible with this command.

The following commands first show the state without a default subvolume, next the subvolume `@` with ID 256 is set as default. The output of the last command shows the change.

```
# btrfs subvolume get-default /
ID 5 (FS_TREE)
# btrfs subvolume set-default 256 /
# btrfs subvolume get-default /
ID 256 gen 22981 top level 5 path @
```

Mount subvolume

As described before, the access to the *top level 5* subvolumes changes after setting a default subvolume.
Example without default subvolume:

```
# mount -t btrfs /dev/sdxX /mnt/
# ls /mnt/
@  @daten  @root  @snapshots  @var@log
```

Example after setting the subvolume @ as default:

```
# mount -t btrfs /dev/sdxX /mnt/
# ls /mnt/
bin    disks  initrd.img      lib64   proc  srv  var
boot   etc    initrd.img.old  libx32  root  sys  vmlinuz
daten  fll    lib             media   run   tmp  vmlinuz.old
dev    home   lib32           mnt     sbin  usr
```

To get to the *top level 5* subvolumes with the default subvolume set, the *subvolid* must be specified in the mount command.

Example with default subvolume and mount option `subvolid=5`:

```
# mount -t btrfs -o subvolid=5 /dev/sdxX /mnt/
# ls /mnt/
@  @daten  @root  @snapshots  @var@log
```

After installation, the `/etc/fstab` file already contains all the necessary entries to automatically mount the subvolumes.

To show how to manually mount a subvolume and to extend the `/etc/fstab` file, we use the `@data` subvolume created in the next chapter.

With the command

mount -t btrfs -o subvol=@data,defaults /dev/sdxX /data/

we mount the subvolume manually.

This simple variant is not suitable for permanent use. It also suppresses the advantageous capabilities of Btrfs. We look at an entry from the `/etc/fstab` file.

```
# grep home /etc/fstab
UUID=<here>  /home  btrfs  subvol=/@home,defaults,noatime,↙
    space_cache=v2,autodefrag,compress=zstd 0 0
```

The option *"space_cache=v2"* caches the addresses of the free blocks on the drive to speed up write operations.

The option *"autodefrag"* ensures defragmentation of the files during runtime.

We achieve data compression with the *"compress=zstd"* option.

Our self-created subvolume @data should be automatically and permanently available with these options. Therefore we add the required entry to /etc/fstab either with an editor or by means of two commands.

```
# echo "# Extended by root on $(date +%F)" >> /etc/fstab
# grep home /etc/fstab | sed 's!home!data!g' "$@" >> /etc/⤶
   fstab
```

Immediately after, the subvolume is available by the short command **mount /⤶ data** and it is mounted like all the others at every boot.

Create new subvolume

To create a new *top level 5* subvolume @data, we mount the siduction Btrfs partition under /mnt.

```
# mount -t btrfs -o subvolid=5 /dev/sdxX /mnt/
# ls /mnt/
@  @home  @root  @snapshots  @var@log
```

The *ls* command shows the existing *top level 5* subvolumes after installation.

Now we create the new subvolume as well as its mount point and reissue the contents of /mnt.

```
# btrfs subvolume create /mnt/@data
# mkdir /mnt/@data
# ls /mnt/
@  @data  @home  @root  @snapshots  @var@log
```

To allow normal users access to the directory, we change the group:

```
# chgrp users /mnt/@data
```

Subvolumes can also be nested and thus be created within existing subvolumes. For a better overview, we rather recommend the flat scheme.

7.4.2 Btrfs snapshot

A snapshot is a subvolume like any other, but with a given initial content. Viewed in the file manager, it appears to contain a complete copy of the original subvolume. Btrfs is a copy-on-write file system, so it is not necessary to actually copy all the data. The snapshot simply has a reference to the current filesystem root of its original subvolume. Only when something is changed does Btrfs create a copy of the data. File changes in a snapshot do not affect the files in the original subvolume.

A snapshot is not recursive. A subvolume or snapshot is effectively a barrier. Files in nested subvolumes do not appear in the snapshot. Instead, there is a blind subvolume, which could cause confusion in nested layouts. The non-recursive behavior explains why siduction created additional subvolumes during installation. Thus, private and variable data from `@home`, `@root`, and `@var@log` subvolumes do not end up in a snapshot of `@`.

It should be noted that snapshots of Btrfs file systems are in no way a substitute for thoughtful data protection. Even for RAID1 and RAID10 systems with Btrfs, the focus is on failover and not on backup.

Create snapshot

> **Caution**
> Use only if you do **not** want to use Snapper.

Since a snapshot is a subvolume within its source, it makes sense to create a corresponding subdirectory. For the example we take our self created subvolume `@data`, create the directory and immediately afterwards the first snapshot.

```
# mkdir /data/.snapshots
# btrfs subvolume snapshot -r /data/ /data/.snapshots/01
  Create a readonly snapshot of '/data' in '/data/.snapshots⤸
      /01'
```

The command is syntactically reminiscent of a simple copy operation, where `01` is the folder where the files of the snapshot are located.

Instead of `01` you can use `$(date +%F_%H-%M)` to get the date and time as folder name.

By default, snapshots are created with read and write access. With the `-r` option they are read-only. We strongly advise using the `-r` option because a snapshot represents the state of the subvolume at the time it is created. How to access the data of a snapshot is explained in the manual in the chapters starting with "Snapper Rollback".

7.5 Snapper

Snapper is a tool for managing file system snapshots on Linux for Btrfs file systems and thin-provisioned LVM volumes. Besides creating and deleting snapshots, it can also compare snapshots and undo differences between snapshots. It allows users to view older versions of files and undo changes. In addition, Snapper supports automatic snapshots according to schedules or to actions.

The default configuration of Snapper in siduction includes automatic pre- and post-snapshots of the `@` subvolume when changes are made to the system and preparation of scheduled snapshots for any other subvolumes.

The Snapper files are located in:

- `/usr/bin/` The `snapper` executable program.

- `/usr/lib/snapper/` Utilities for snapper.

- `/etc/default/snapper` An overview of the configured subvolumes.

- `/etc/snapper/configs/` The configuration files of the configured subvolumes.

- `/usr/share/snapper/config-templates/` The configuration templates.

- `/var/log/snapper.log` Snapper's log file.

Please read the man pages **man snapper** and **man snapper-configs**.

7.5.1 Snapper configuration

Snapper works together with systemd. Some settings regarding the handling of automatic snapshots are hidden in the associated systemd units. The chapter "Snapper and systemd" explains the functions and gives hints for their adjustment.

Siduction automatically creates the configurations for the @ and @home subvolumes during installation. For the other sub-volumes, we must create configurations ourselves as required according to the following pattern.

```
# snapper -c <config_name> create-config -t <config_template>↲
    <subvolume_mount_point>
```

But first let's take a look at the configuration for the subvolume @ with the name `root`, @home with the name `home`, and the default template `default` supplied by snapper.

```
Snapper configuration
----------------------+-------+-------+-------+
Subvolume             |   @   | @home |  --   |
----------------------+-------+-------+-------+
conf-name or templ-name| root | home  |default|
======================+=======+=======+=======+
Key                   | Value | Value | Value |
----------------------+-------+-------+-------+
ALLOW_GROUPS          | users | users |       |
ALLOW_USERS           |       |       |       |
BACKGROUND_COMPARISON | yes   | yes   | yes   |
EMPTY_PRE_POST_CLEANUP | yes  | yes   | yes   |
EMPTY_PRE_POST_MIN_AGE | 1800 | 1800  | 1800  |
FREE_LIMIT            | 0.2   | 0.2   | 0.2   |
FSTYPE                | btrfs | btrfs | btrfs |
NUMBER_CLEANUP        | yes   | yes   | yes   |
```

```
NUMBER_LIMIT             | 50    | 50    | 50    |
NUMBER_LIMIT_IMPORTANT   | 10    | 10    | 10    |
NUMBER_MIN_AGE           | 1800  | 1800  | 1800  |
QGROUP                   |       |       |       |
SPACE_LIMIT              | 0.5   | 0.5   | 0.5   |
SUBVOLUME                | /     | /home | /     |
SYNC_ACL                 | yes   | yes   | no    |
TIMELINE_CLEANUP         | yes   | yes   | yes   |
TIMELINE_CREATE          | no    | no    | yes   |
TIMELINE_LIMIT_DAILY     | 10    | 10    | 10    |
TIMELINE_LIMIT_HOURLY    | 10    | 10    | 10    |
TIMELINE_LIMIT_MONTHLY   | 10    | 10    | 10    |
TIMELINE_LIMIT_WEEKLY    | 0     | 0     | 0     |
TIMELINE_LIMIT_YEARLY    | 10    | 10    | 10    |
TIMELINE_MIN_AGE         | 1800  | 1800  | 1800  |
-------------------------+-------+-------+-------+
```

A *"pre '* and *'post "* snapshot is created from the `@` subvolume for each APT action. The key `NUMBER_LIMIT=50` ensures that the most recent twenty-five snapshot pairs are retained.

siduction has set the keys `ALLOW_GROUPS=users` and `TIMELINE_CREATE=no`. The former allows all members of the *users* group to perform Snapper actions, the latter prevents an excessive increase in the number of snapshots held.

We can also change individual *key=value* pairs on the command line. In the example, we reduce the number of numbered snapshots held in the `root` configuration.

```
# snapper -c root set-config NUMBER_LIMIT=20
```

Now the most recent ten instead of twenty-five pre- and post-snapshot pairs remain after APT actions. For standard use on a laptop or PC, this value should be sufficient.

If you take into account the Snapper *default* configuration with an active *TIMELINE_CREATE* key and an APT action every day, the snapshots add up to almost 100 within a month. In addition, the very first *timeline snapshot* has been floating

around in our file system for at least ten years, believe it or not. Who wants to reset their production system to this snapshot and keep all the data for that long?
Please note: Snapper and snapshots are not a means of data backup. They enable the system to be reset promptly in the event of errors occurring or actions initiated by us that have had unintended effects.

At this point, every siduction user should consider how many snapshots they want to keep and for how long and adjust the configuration accordingly.

This is also possible with your own configuration template. For example, for the *@data* subvolume created in the *Btrfs* chapter.
To do this, we copy the file `/usr/share/snapper/config-templates/default` into the same directory with the new name `user` and then change the key-value pairs as required. We use this template to create the configuration for our `@data` subvolume.

```
# snapper -c data_pr create-config -t user /data
```

This:

1. creates the `/etc/snapper/configs/data_pr` configuration file based on the `/usr/share/snapper/config-templates/user` template.

2. creates the `/data/.snapshots` subvolume where future snapshots of `@data` will be stored. The path of a snapshot is `/data/.snapshots/#/↙ snapshot`, where # is the snapshot number.

3. adds the name of the `data_pr` configuration to the key *"SNAP-PER_CONFIGS"* in the `/etc/default/snapper` file.

Now the configuration is active. If the key `TIMELINE_CREATE=yes` is set, systemd takes over the regular creation of *"timeline snapshots"* through its timers.

7.5.2   Snapper and systemd

Snapper installs three systemd unit pairs to create or delete snapshots depending on APT actions and time.

- When creating snapshots with the keys

  `DISABLE_APT_SNAPSHOT="no"` in the `/etc/default/snapper` file

  with the help of the Systemd unit

  `grub-btrfs.path` and `grub-btrfs.service`

  and

  `TIMELINE_CREATE="yes"` in the configuration files of the subvolumes

  with the help of the Systemd unit

  `snapper-timeline.timer` and `snapper-timeline.service`.

- When deleting snapshots with the keys

  `EMPTY_PRE_POST_CLEANUP=yes`,

  `NUMBER_CLEANUP=yes`,

  `TIMELINE_CLEANUP=yes` in the configuration files of the subvolumes

  with the help of the Systemd unit

  `snapper-cleanup.timer` and `snapper-cleanup.service`.

The fact that Snapper creates a pre- and post-snapshot for every APT action should definitely be kept in siduction. siduction is a rolling release based on Debian sid. It is quite possible to get single packages that do not work as intended when upgrading. A rollback with Snapper is then a good alternative for the user to continue to work reliably.

On the other hand the *TIMTLINE* function offers room for individual adjustments. The right addressees are the two timer units `snapper-timeline.timer` and `snapper-cleanup.timer`. The former is the timer for creating snapshots, the latter determines the time for removing old and empty snapshots.

The manual page *systemd-timer* explains how the timer unit works.

Now we turn to the contents of the systemd unit `snapper-timeline.timer` in the directory `/lib/systemd/system/`.

```
[Unit]
Description=Timeline of Snapper Snapshots
Documentation=man:snapper(8) man:snapper-configs(5)

[Timer]
```

```
OnCalendar=hourly


[Install]
WantedBy=timers.target
```

With the command **systemctl edit --full snapper-timeline.timer** we open a text editor and change the file as follows:

```
[Unit]
Description=Timeline of Snapper Snapshots
Documentation=man:snapper(8) man:snapper-configs(5)


[Timer]
#OnCalendar=hourly
OnBootSec=30
OnUnitActiveSec=2h


[Install]
WantedBy=timers.target
```

With this change, we get a snapshot thirty seconds after the boot and every two hours thereafter. From now on, Snapper creates a maximum of twelve snapshots every day instead of twenty-four.

We save the file and close the editor. systemd creates the changed file with the same name in the /etc/systemd/system/ directory and runs the **systemctl** ↙ **daemon-reload** command to load the changed configuration.

The second systemd timer unit snapper-cleanup.timer takes care of disposing of old, excess and empty snapshots. It has the following content:

```
[Unit]
Description=Daily Cleanup of Snapper Snapshots
Documentation=man:snapper(8) man:snapper-configs(5)


[Timer]
OnBootSec=10m
```

```
OnUnitActiveSec=1d


[Install]
WantedBy=timers.target
```

With the knowledge of the contents of the TIMELINE timer we can weigh now whether the configuration is meaningful. For someone who restarts his PC every day, the key `OnBootSec=10m` might be rather unfavorable if he finds that a serious error has crept in shortly before closing time on the previous day. In this case it probably makes more sense to set the key to `OnBootSec=4h`. The file is changed in the same way as in the example shown above.

7.5.3   Snapper - manual snapshots

Of course, with Snapper we can also create snapshots independently of the automatic actions. For this, the executing user must be listed in the subvolume's Snapper configuration with group or user rights.

The syntax of the command corresponds to the following pattern which also shows frequently used options.

```
# snapper -c <config_name> create -t <type> -d <description> ↵
    -c <cleanup-algorithm> -u <userdata>
```

- snapper **-c <config_name>** create
  This snapper command creates a snapshot of the subvolume of the named configuration. If the option is missing, Snapper applies the command to the Btrfs default subvolume (the root directory *I*) with the `root` configuration. This rule applies to all Snapper commands.

- **-t <type>**
  specifies the type of snapshot to create. Possible values: `single`, `pre`, `post`.

- **-d <description>**.

  may contain any text. Use `"` if spaces or special characters are included.

- **-c <cleanup-algorithm>**.

  This option determines the rules according to which the snapshot should be automatically deleted. Possible values: `number`, `timeline`, `pre`, `post`. If this option is missing, the snapshot will be kept until the user deletes it manually.

- **-u <userdata>**

  specifies user data for the snapshot. The format must be *key=value*. Multiple user data must be separated by a comma, for example `author=Tom,↙ important=yes`.

Snapper always creates snapshots in *read-only* mode. You can change the default with the `--read-write` option. Changing data in a snapshot will lead to inconsistent data sets. We strongly advise against this unless you know exactly what you are doing.

Now we create a snapshot and display the snapshots of the same configuration.

```
$ snapper -c data_pr create -t single -d "AB finished" -c ↙
   number -u user=Pit
$ snapper -c data_pr list
 #|Typ   |Pre #|Date    |User|Cleanup |Description|Userdata
--+------+-----+--------+----+--------+----------+--------
 0|single|     |        |root|        |current   |
88|single|     |22:00:38|root|timeline|timeline  |
90|single|     |11:34:41|root|timeline|timeline  |
91|single|     |11:36:23|user|number  |AB finished|user=Pit
```

The snapshot we (user) created has # 91. Unfortunately we made the mistake to let the snapshot be handled according to the cleanup rule *number*. We change this with the *modify -c ""* option so that Snapper will not delete it automatically.

```
$ snapper -c data_pr modify -c "" 91
$ snapper -c data_pr list
 #|Typ   |Pre #|Date    |User|Cleanup |Description|Userdata
```

```
--+------+-----+--------+----+--------+----------+--------
 0|single|     |        |root|        |current   |
88|single|     |22:00:38|root|timeline|timeline  |
90|single|     |11:34:41|root|timeline|timeline  |
91|single|     |11:36:23|user|        |AB finished|user=Pit
```

Snapshot # 91 will now remain until we delete it ourselves.

Delete snapshot

We can delete any snapshot at any time as long as we have the rights to do so. Snapper does not care about the delete action, because on each run the cleanup algorithm checks which snapshots are kept. The preceding chapter Snapper Configuration also explains in detail the settings with which we can adjust the cleanup algorithm if necessary.

The following command removes snapshot # 91 from our `@data` subvolume.

```
$ snapper -c data_pr delete 91
```

The `delete 34-50` command deletes a number of snapshots.
The snapshot # 0 with the description *"current"* is not deletable. It is the snapshot that is mounted to the file tree and in which we are currently working.

7.5.4 Snapper rollback

If the system is damaged due to an action initiated by us that went completely out of control, or due to a faulty upgrade, Snapper allows you to use the *"rollback"* to return the system to one or more states that existed before the problems occurred.

Prerequisites
A *"rollback"* is only supported with Btrfs for the root file system. The root file system must be on a single device, in a single partition, and on a single subvolume. Directories that are excluded from `/` snapshots, for example `/root`, can be on separate partitions.

Performing a rollback
Before the rollback, we check if the rollback target works as expected. To do this,

we boot into the desired snapshot, for example 13, using the *"siduction snapshots"* submenu. The system boots in *read-only* mode. We ignore the error message regarding *sddm*. If it does, we reboot back to the current default subvolume. There we perform the rollback as **root**:

```
# snapper --ambit classic rollback
Ambit is classic.
Creating read-only snapshot of default subvolume. (Snapshot ↙
   15.)
Creating read-write snapshot of snapshot 13. (Snapshot 16.)
Setting default subvolume to snapshot 16.
```

Always execute rollback from the default subvolume specifying the subvolume number of the rollback target.

The output precisely describes the rollback procedure. Afterwards the grub menu file *grub.cfg* is automatically updated to show the new snapshots in the submenu and snapshot 16 is used as the default subvolume. The grub menu file is updated whenever the paths of the btrfs-default subvolume, the booted subvolume, or the grub-default menu entry differ after a snapshot, rollback, or reboot.

The `snapper list` command shows that we are currently in snapshot 12 and snapshot 16 is the new default subvolume. (The minus - after #12 and the plus + after #16.)

```
 # |Typ   |Pre #|Date     |User |Cleanup| Description   |
---+------+-----+--------+-----+-------+---------------+
 0 |single|     |         |root |       |current        |
12-|single|     |17:28:15|root |number |important      |
13 |pre   |     |11:34:41|root |number |apt            |
14 |post  |   13|11:35:56|root |number |apt            |
15 |single|     |12:05:23|root |number |rollback backup|
16+|single|     |12:05:23|root |       |r/w copy of #13|
```

We perform a reboot and select the Grub default entry. Now the * after #16 indicates that we are in this snapshot and it is the default subvolume.

```
 # |Typ   |Pre #|Date     |User |Cleanup| Description   |
```

```
16*|single|      |12:05:23|root |         |r/w copy of #13|
```

The Grub menu file is also updated automatically in the rollback target and Grub is reinstalled from snapshot #16. From now on, Grub reads the menu file from the new default subvolume #16.

The following graphic illustrates how the test-btrfs-default script adjusts the configuration of GRUB after a rollback.
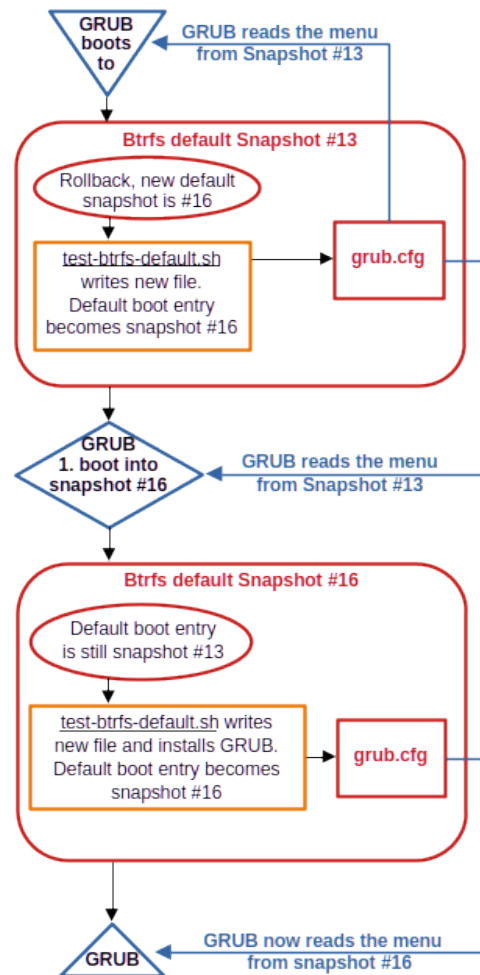
7.5.5   File rollback within the root file system

This is the undoing of changes to files. For this purpose, two shnapshots are com-
pared and then the file to be changed is picked out. Afterwards you can see the
changes and decide if you want to undo them.

The output of **snapper list** shows the currently existing snapshots of the subvol-
ume @. (The columns have been shortened.) All snapshots with a digit # greater
than zero represent the state of the file system at that exact time. The only excep-
tion is the one marked with a *. It was booted into and is the default snapshot. If
no system rollback has been performed yet, snapshot 0 takes its place.

```
  # |Typ   |Pre #|Date    |User |Cleanup |Description|Us..
---+------+-----+--------+-----+--------+-----------+----
  0 |single|     |        |root |        |current    |
 42 |single|     |09:50:36|root |        |IP pc1     |
 43 |pre   |     |11:30:18|root |number  |apt        |
 44 |post  |   43|11:34:41|root |number  |apt        |
 45*|single|     |22:00:38|root |        |           |
 46 |single|     |23:00:23|root |timeline|timeline   |
```

Two snapshots can be compared with:

```
# snapper status 42..45
[...]
c..... /etc/group
+..... /etc/group-
c..... /etc/hosts
[...]
```

Each line names a file and the type of change. A + at the beginning of the line
means that the file was created, a - that the file was deleted, and a c that the
contents of the file were changed.
If the output includes a lot of lines, we redirect it to a file with the -o </path/⤸
name> option.

Differences in a file between two snapshots can be displayed with:

```
# snapper diff 42..45 /etc/hosts
--- /.snapshots/42/snapshot/etc/hosts
+++ /.snapshots/45/snapshot/etc/hosts
@@ -5,5 +5,3 @@
 ff02::2     ip6-allrouters
 # This host address
 127.0.1.1  lap1
-# added 2022-12-02
-192.168.3.1 pc1
```

If we want to undo the change, we use the command:

```
# snapper undochange 42..45 /etc/hosts
```

A *"file rollback"* within the root file system only makes sense if a snapshot is to be prepared for a *"system rollback"*, or the snapshot into which the system was booted is involved (recognizable by the `*` mark). It may be necessary to restart services or daemons, or even to reboot.

It is also possible to include several files separated by spaces in the command.

Caution

If the command **snapper undochange 42..45** is entered without specifying a file, Snapper will undo all changes between snapshots 42 and 45. The better alternative for such an operation is a *"system rollback"*.

7.5.6   File rollback of user data

With Snapper alone

Snapper treats snapshot 0 as a snapshot, but it represents the current state of the subvolume and is thus variable. All other snapshots, as previously mentioned, represent the state of the file system at exactly its point in time. Changes between these snapshots therefore only act in the past.

For us, this means that a *"file rollback"* of user data between snapshots 15 and 17

is worthless, since the operation does not affect the current state in our subvolume. So we always need snapshot 0 as a target for changes.

We look at such an operation using the `Test.txt` file in the `@data` subvolume.

```
$ snapper -c data_pr list
  #|Typ   |Pre #|Date     |User     |Cleanup   |Descr.
---+------+-----+--------+--------+---------+--------
  0 |single|     |         |root     |          |current
 15 |single|     |12:50:48|root     |timeline  |timeline
 16 |single|     |13:51:08|root     |timeline  |timeline
 17 |single|     |14:51:26|root     |timeline  |timeline
```

The comparison between snapshot 15 and 16:

```
$ snapper -c data_pr status 15..16
[...]
+..... /data/user1/Test.txt
[...]
```

The file first appears in snapshot 16. We compare with the next snapshot.

```
$ snapper -c data_pr status 16..17
[...]
c..... /data/user1/Test.txt
[...]
```

The file was changed between snapshots 16 and 17.

This is followed by a query with `diff` that prints the changes between 16 and 17.

```
$ snapper -c data_pr diff 16..17 /data/user1/Test.txt
--- /data/.snapshots/16/snapshot/user1/Test.txt
+++ /data/.snapshots/17/snapshot/user1/Test.txt
@@ -8,6 +8,8 @@
 test file

 This text was alreadey in
 the file before the snapshot 16.
```

```
-So was this one, but it was deteted.
+
+This text was inserted after the snapshot 16.
```

Since the file has not been modified since snapshot 17, the **$ snapper -c** ↙
data_pr diff 16..0 /data/user1/Test.txt command does not produce any
other output for comparing snapshot 16 with the current contents of the file.

Now we put the undochange command between 16 and 0. After that the *Test.txt*
contains the first six lines from snapshot 16.

```
$ snapper -c data_pr undochange 16..0 /data/user1/Test.txt
create:0 modify:1 delete:0

$ cat /data/user1/Test.txt
test file

This text was alreadey in
the file before the snapshot 16.

So was this one, but it was deteted.
```

A deleted file is promoted back to the current directory with the same command.
Only the feedback from Snapper changes slightly.

```
$ snapper -c data_pr undochange 16..0 /data/user1/Test.txt
create:1 modify:0 delete:0
```

With Snapper and Meld

The preceding procedure always restores a file as a whole to the state correspond-
ing to the selected snapshot. Individual parts of the changes cannot be applied in
this way.
The comparison program **Meld** fills exactly this gap. *Meld* is additionally able to
insert parts at any place in the current document via *copy & paste* (an advantage

also towards **Kompare** of the KDE Desktop). In siduction, *Meld* is not installed by default. We will make up for this.

The actions of Snapper are always possible for the non **root** user if the key `ALLOW_GROUPS=users` is set in the configuration file for the subvolume. This is the default. However, they are denied access to the snapshot files within the file system because the `/.snapshots` directory is readable and executable only by **root**. To work with *Meld*, we change this.

Make snapshots readable for users and install *Meld*. (Run as **root**.)

```
# chmod a+rx /data/.snapshots
# apt update && apt install meld
```

As a reminder, snapshots in Btrfs are always stored in read-only mode. The only exception is the `system rollback` snapshot.

We use snapper to select file changes in the same way as before. The command **$ snapper -c data_pr diff 16..0 /data/user1/Test.txt** contains the exact path to the file `Test.txt` in the snapshot.

```
$ snapper -c data_pr diff 16..0 /data/user1/Test.txt
--- /data/.snapshots/16/snapshot/user1/Test.txt
+++ /data/user1/Test.txt
[...]
```

We start *Meld* and select the two files with their paths for the file comparison. The differences are immediately visible.
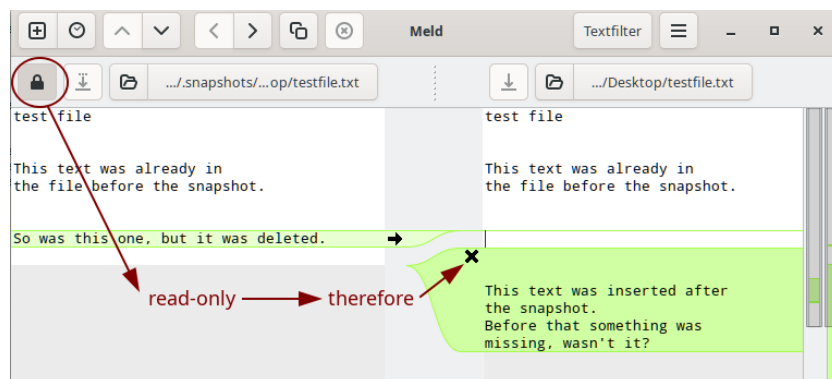
A click on the arrow transfers the line to our current file. Another click on the cross removes the other lines. A transfer to the file in the snapshot is not possible, because the file system of the snapshot is read-only.

Since Snapper shows us the exact path to our file in the snapshot, we also have the very conventional option of copying a file from the snapshot to our current working directory.

```
$ cp /data/.snapshots/16/snapshot/user1/Test.txt /home/user1/⤸
   Test.txt
```

7.5.7 Sources BTRFS and Snapper

- `man btrfs` and `man btrfs-subvolume` as well as other subpages of *"man btrfs"*

- Btrfs wiki from kernel.org

- Btrfs documentation

- Btrfs snapshot in grub menu

- `man snapper` and `man snapper-configs`

- Snapper project page

- Snapper on GitHub

Last edited: 2024-04-30

7.6   APT package management

APT is an acronym for **A**dvanced **P**ackaging **T**ool and provides a collection of programs and scripts that assist the system and administrator in installing and managing Debian packages.
A complete description of the APT system can be found in Debian's APT-HOWTO.

7.6.1   apt and apt-get

apt is intended as an end-user interface and, compared to more specialized tools such as apt-get and apt-cache, enables some options more suitable for interactive use by default. With apt not all options of apt-get and apt-cache are available. Please read the man pages of apt, apt-get, and apt-cache. The following table shows the respective commands and their basic meaning.

apt	apt-get	short info
apt update	apt-get update	Refresh the package database.
apt upgrade	apt-get upgrade	Update the system to the latest available package versions.
apt full-upgrade	apt-get dist-upgrade	Upgrade the system to the latest available package versions even if it means removing already installed packages.
apt full-upgrade -d	apt-get dist-upgrade -d	Upgrade the system as before, but only downloads without installing anything.
apt install	apt-get install	Install one or more packages.

apt	apt-get	short info
apt remove	apt-get remove	Remove one or more packages.
apt purge	apt-get purge	Remove one or more packages including configuration files.
- apt-mark hold	Prevent apt from installing another version of the package.	
-	apt-mark unhold	Cancel the 'apt-mark hold' command.
apt search	apt-get search	Search for packages according to the pattern entered (regex possible).
apt show	apt-cache show	Display the details of a package.
apt list	apt-cache policy	Show the installed or installable version of a package.

7.6.2   sources.list - List of sources

The APT system needs a configuration file which contains information about the location of installable and upgradeable packages. In general, this file is called sources.list. Modern systems use modularized sources now to improve the overview.

siduction provides the sources in this folder:

```
/etc/apt/sources.list.d/
```

Inside this directory you can find the following files by default:

```
debian.list
extra.list
fixes.list
```

This has the advantage that it is easier to automatically select from mirror servers ("mirror switching"), and it also makes it easier to add or replace source lists.

Custom source list files can be added with the naming
`/etc/apt/sources.list.d/xxxx.list`.
For example, on siduction
`/etc/apt/sources.list.d/extra.list` might look like this:

```
This is the default mirror, chosen at first boot.
# One might consider to choose the geographically nearest
 or the fastest mirror.

deb http://packages.siduction.org/extra unstable main contrib↙
    non-free

#deb-src http://packages.siduction.org/extra unstable main ↙
   contrib non-free
```

Under `/etc/apt/sources.list.d/fixes.list` it might look like this:

```
deb https://packages.siduction.org/fixes unstable main ↙
   contrib non-free

#deb-src https://packages.siduction.org/fixes unstable main ↙
   contrib non-free
```

And `/etc/apt/sources.list.d/debian.list` contains the actual Debian repo:

```
# debian loadbalancer
deb http://deb.debian.org/debian/ unstable main contrib non-↙
   free

#deb-src http://deb.debian.org/debian/ unstable main contrib ↙
   non-free
```

More entries for optional siduction repositories can be found at siduction reposito-ries.

For example, adding one or more Debian repositories would look like this:

```
#Debian
#Unstable
deb http://ftp.us.debian.org/debian/ unstable main contrib ⤶
    non-free

#deb-src http://ftp.us.debian.org/debian/ unstable main ⤶
    contrib non-free

# Testing
#deb http://ftp.us.debian.org/debian/ testing main contrib ⤶
    non-free

#deb-src http://ftp.us.debian.org/debian/ testing main ⤶
    contrib non-free

# Experimental
#deb http://ftp.us.debian.org/debian/ experimental main ⤶
    contrib non-free

#deb-src http://ftp.us.debian.org/debian/ experimental main ⤶
    contrib non-free
```

NOTE:

This example uses the US Debian mirror starting with ftp.us. This setting can be changed as **root** by adjusting the country code (for example: ftp.at, ftp.de). Most countries have local Debian mirrors available. This provides a higher connection speed for the user and also saves bandwidth.

List of currently available Debian servers and their mirrors

7.6.3   apt update

To get updated information about the packages, a database is kept with the needed entries. The apt program uses it when installing a package to resolve all dependencies and thus to guarantee that the selected packages will work. The creation or update of this database is done with the command **apt update**.

```
root@siduction# apt update
Fetch:1 http://siduction.org sid Release.gpg [189B]
Fetch:2 http://siduction.org sid Release.gpg [189B]
Fetch:3 http://siduction.org sid Release.gpg [189B]
Get:4 http://ftp.de.debian.org unstable Release.gpg [189B]
Get:5 http://siduction.org sid Release [34.1kB]
Get:6 http://ftp.de.debian.org unstable Release [79.6kB]
Fetched 404 kB in 8 s (50.8 kB/s).
Package lists are read... Done
Dependency tree is built.
Status information is read.... Done
Upgrade available for 48 packages. Run "apt list --upgradable⤾
    " to view them.
```

7.6.4   Install packages

If we know the package's name, the command **apt install <package_name>** is sufficient.
(See below for how to find a package.)

> **Warning:**
> Packages that are not installed in the multi-user.target (formerly runlevel 3) can bring big, unsupportable problems!

Therefore we recommend the following procedure:

1. Log out of the desktop environment.
2. Switch to the text console with **Ctrl**+**Alt**+**F3**
3. Log in as **root**.

Then install the desired program package:

```
init 3
apt update
apt install <package_name>
init 5 && exit
```

In the example below, the package "funtools" is installed.

```
root@siduction# apt install funtools
Reading package list... Done
Building dependency tree
Reading state information.... Done
The following additional packages will be installed:
  libfuntools1 libwcstools1
The following NEW packages will be installed:
  funtools libfuntools1 libwcstools1
0 updated, 3 reinstalled, 0 to remove and 48 not upgraded.
Need to get 739 kB of archives.
After this operation, 2,083 kB of additional disk space will ⤸
    be used.
Do you want to continue? [Y/n] y
Get:1 http://deb.debian.org/debian unstable/main amd64 ⤸
    libwcstools1 amd64 3.9.5-3 [331 kB]
Get:2 http://deb.debian.org/debian unstable/main amd64 ⤸
    libfuntools1 amd64 1.4.7-4 [231 kB]
Get:3 http://deb.debian.org/debian unstable/main amd64 ⤸
    funtools amd64 1.4.7-4 [177 kB]
Fetched 739 kB in 0 s (1,678 kB/s).
Selecting previously unselected package libwcstools1:amd64.
(Reading database ... 279741 files and directories currently ⤸
    installed).
Preparing to unpack .../libwcstools1_3.9.5-3_amd64.deb ...
Unpacking libwcstools1:amd64 (3.9.5-3) ...
Selecting previously unselected package libfuntools1:amd64.
Preparing to unpack .../libfuntools1_1.4.7-4_amd64.deb ...
Unpacking libfuntools1:amd64 (1.4.7-4) ...
```

```
Selecting previously unselected package funtools.
Preparing to unpack .../funtools_1.4.7-4_amd64.deb ...
Unpacking funtools (1.4.7-4) ...
Setting up libwcstools1:amd64 (3.9.5-3) ...
Setting up libfuntools1:amd64 (1.4.7-4) ...
funtools (1.4.7-4) is set up ...
Processing triggers for man-db (2.8.5-2) ...
Processing triggers for libc-bin (2.28-8) ...
```

7.6.5   Remove packages

The `apt remove <package_name>` command removes a package, but its dependencies remain:

```
root@siduction# apt remove gaim
Reading package lists... Done
Building dependency tree.
Reading state information.... Done
The following packages were installed automatically and are ⤸
    no longer needed:
      libfuntools1 libwcstools1
Use "sudo apt autoremove" to remove them.
The following packages will be REMOVED:
      funtools
0 updated, 0 reinstalled, 1 to remove and 48 not upgraded.
After this operation, 505 kB of disk space will be freed.
Do you want to continue? [Y/n] y
(Read database ... 279786 files and directories are currently⤸
    installed).
Removing funtools (1.4.7-4) ...
Processing triggers for man-db (2.8.5-2) ...
```

In the last case, the configuration files are not removed from the system; they can be reused in a later reinstallation of the program package (in the example: gaim). If the configuration files should also be removed, then the following call is needed:

apt purge funtools

This will also remove the configuration files. In case you want to see if configuration files of already removed programs are still on the system, you can easily get a result with dpkg:

```
dpkg -l | grep ^rc
rc colord 1.4.3-3.1 amd64 system service to manage device ↙
   color profiles -- system daemon
rc hplip 3.18.10+dfsg0-1 amd64 HP Linux Printing and Imaging ↙
   System (HPLIP)
rc libsensors4:amd64 1:3.4.0-4 amd64 library to read ↙
   temperature/voltage/fan sensors
rc sane 1.0.14-13.1 amd64 scanner graphical frontends
rc sane-utils 1.0.27-3.1 amd64 API library for scanners -- ↙
   utilities
rc systemd-coredump 240-1 amd64 tools for storing and ↙
   retrieving coredumps
```

The packages listed here were removed without purging.

7.6.6   Hold or downgrade a package

Sometimes it may be necessary to revert to an earlier version of a package because the latest version has a serious bug.

Hold

With apt-mark, apt allows you to apply various settings for a package. The hold option protects the package from changes by apt.

```
apt-mark hold <package_name>
```

To end the holding of a package:

```
apt-mark unhold <package_name>
```

This is how to search for packages that are placed on hold:

```
apt-mark showhold
```

Please keep in mind that hold is only an emergency measure. You will run into problems if you forget to release a hold in a timely manner. This is even more true the more (essential) dependencies the package has. So please use holds only in case of emergency and release them as soon as possible.

Downgrade

Debian does not support package downgrading. In simple cases, installing older versions can succeed, but it can also fail spectacularly. For more information, see the Emergency downgrading chapter in the Debian Handbook.

Although downgrading is not supported, it can succeed for simple packages. The steps for a downgrade are now demonstrated on the package kmahjongg:

The sources of unstable are stored in
`/etc/apt/sources.list.d/debian.list`
with a hash sign "#" and add the sources for testing. After that, we execute the following commands:

```
apt update
apt install kmahjongg/testing
```

The now installed package is set to hold in order to protect it from upgrades:

```
apt-mark hold kmahjongg
```

Then the sources for testing are marked with a hash "#" in
`/etc/apt/sources.list.d/debian.list`,
while the hashes in front of the sources for unstable are removed again. After saving the changes, enter:

```
apt update
```

As soon as a new, bug-free package arrives in sid, you can reinstall the latest version if you exit the hold state:

```
apt-mark unhold kmahjongg
apt update
apt install kmahjongg / apt full-upgrade
```

7.6.7   Updating the system

An upgrade of the whole system is performed with this command: **apt full-**∠
upgrade. Before such an action, the current upgrade warnings on the main siduc-
tion page should be followed to check whether packages of one's own system are
affected. If an installed package should be kept, i.e. put on hold, please refer to
the section downgrade or "hold" of a package.

A simple **apt upgrade** of Debian Sid is usually not recommended. However, it can
be helpful if there is a situation with many packages held or to be removed. Here
an **apt upgrade** can update packages not affected by the situation.

How regularly should a system upgrade be performed?
A system update should be performed regularly, every one to two weeks has
proven to be a good guideline. Even monthly system updates should not cause
any significant problems. Theoretically, the system can be updated several times
a day after mirror synchronization every 6 hours.

Experience shows that you should not wait longer than two, maximum three
months. Special attention should be paid to program packages which do not come
from the siduction or Debian repositories or which have been compiled by yourself,
as they may lose their functionality after a system update via full-upgrade due to
incompatibilities.

Update not with live media

There is no possibility to update a siduction installation using a live medium. Below
we describe in detail the upgrade process and why apt should be used.

7.6.8   Updateable packages

After updating the internal database, you can find out for which packages a newer version exists (first you need to install apt-show-versions):

```
root@siduction# apt-show-versions -u
libpam-runtime/unstable upgradeable from 0.79-1 to 0.79-3
passwd/unstable upgradeable from 1:4.0.12-5 to 1:4.0.12-6
teclasat/unstable upgradeable from 0.7m02-1 to 0.7n01-1
libpam-modules/unstable upgradeable from 0.79-1 to 0.79-3
[...]
```

The same can be achieved with:

```
apt list --upgradable
```

The upgrade of a single package (here e.g. debtags-1.6.6.0) can be done considering the dependencies with:

```
root@siduction# apt install debtags-1.6.6.0
Reading package lists... Ready
Building dependency tree... Done
The following packages will be REMOVED:
  apt-index-watcher
The following packages will be updated:
  debtags
1 updated, 0 reinstalled, 1 to remove and 0 not upgraded.
Need to get 660kB of archives.
After unpacking, 1991kB of disk space will have been freed.
Do you want to continue [Y/n]?
Get:1 http://ftp.de.debian.org unstable/main debtags 1.6.6 ↙
    [660kB]
Fetched 660kB in 1s (513kB/s)
(Reading database ... 138695 files and directories currently ↙
    installed).
Removing apt-index-watcher ...
```

```
(Reading database ... 138692 files and directories currently ↙
   installed).
Preparing to replace debtags 1.6.2 (with .../debtags_1.6.6↙
   _i386.deb) ...
Unpacking replacement for debtags ...
Setting up debtags (1.6.6) ...
Installing new version of the configuration file /etc/debtags↙
   /sources.list ...
```

Download (only)

A little known but great option is the `-d` option:

```
apt update && apt full-upgrade -d
```

`-d` allows to save the packages of a full-upgrade locally without installing them. This can be done in a console while in X. The full-upgrade itself can be done later in init 3. This also gives one the opportunity to check for any warnings and then decide whether or not to perform the upgrade:

```
root@siduction#apt full-upgrade -d
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following NEW packages will be installed:
  elinks-data
The following packages have been kept back:
  git-core git-gui git-svn gitk icedove libmpich1.0ldbl
The following packages will be upgraded:
  alsa-base bsdutils ceni configure-ndiswrapper debhelper
  discover1-data elinks file fuse-utils gnucash.........
35 upgraded, 1 newly installed, 0 to remove and 6 not ↙
   upgraded.
Need to get 23.4MB of archives.
After this operation, 594kB of additional disk space will be ↙
   used.
```

```
Do you want to continue [Y/n]?Y
```

Y downloads the packages to be updated or reinstalled without changing the installed system.

After downloading the packages with `full-upgrade -d`, they can be installed at any time according to the procedure in the following paragraph.

7.6.9   Run full-upgrade

> **Warning:**
> A system update that is not performed in the multi-user.target (formerly runlevel 3) can lead to problems when it comes to updates of the installed desktop environment or the X server!

Before updating the system, visit the siduction home page to find out about any upgrade warnings. These warnings are necessary because of the structure of Debian sid/unstable which adds new program packages to its repositories several times a day.

The following procedure should be followed:

1. Log out of the desktop environment.
   (This procedure is nowadays only recommended when updating X or the desktop environment itself, but does not hurt in other cases.)

2. Switch to the text console with **Ctrl**+**Alt**+**F3**.
3. Log in as **root**.

Then execute the following commands:

```
init 3
apt update
apt full-upgrade
apt clean
init 5 && exit
```

If a new kernel has been installed, the command **`systemctl reboot`** or **`init 6`** needs to be run instead of *"init 5"* in order to boot with the new kernel.

7.6.10 Why use apt exclusively

For installing, deleting, and carrying out a system update, *apt* should be used. Please refrain from updating the system with applications like `synaptic`, or `discover`!

The mentioned programs are excellent *Debian stable* installation and very good for searching program packages, but they are not adapted to the special tasks of the dynamic distribution Debian Sid. They cannot correctly resolve the extensive changes in Sid (changed dependencies, naming conventions, or scripts). This is not due errors in the programs or mistakes by the developers.

Package managers like synaptic, and discover are - technically speaking - non-deterministic. When using a dynamic distribution like Debian Sid with the addition of third party repositories whose quality cannot be tested by the Debian team, a system update can lead to disaster, as these package managers can make wrong decisions by automatically trying to solve the problem.

Furthermore, it should be noted that all GUI package managers must be run in X. System updates in X (even an anyway not recommended 'apt upgrade') will sooner or later cause you to irreversibly damage your system.

In contrast, apt does only what is requested. In the case of incomplete dependencies in Sid, i.e. when the system breaks (this can happen in Sid during structural changes), the causes can be precisely determined and thus repaired or bypassed. The own system does not "break". So if a system update feels like deleting half the system, apt leaves it up to the administrator to decide what to do, and does not act on its own.

This is the reason why Debian builds use apt and not other package managers.

7.6.11   Searching for program packages

The APT system provides a number of useful search commands that search the APT database and output information about packages. In addition, there are some programs that display the search graphically.

Package search in the terminal

With the simple command

apt search <search_pattern>

you get a list of all packages containing the search pattern. Searching with search allows the use of regex terms.

For example, if you search for *"gman"*, you get this result:

```
user1@pc1:~$ apt search ^gman
Sorting... Done
Full text search... Done
gman/unstable,now 0.9.3-5.3 amd64 [installed]
  small man(1) front-end for X


gmanedit/unstable 0.4.2-7 amd64
  GTK+/GNOME editor for manual pages
```

Here the "^" means that "gman" must be at the beginning of the line. Without this character, the pattern will also find khan*gman* and lo*gman*ager, for example.

If you want more information about the current versions of a package, use:

```
user1@pc1:~$ apt show gman
Package: gman
Version: 0.9.3-5.3
Priority: optional
Section: doc
Maintainer: Josip Rodin <joy-packages@debian.org>
Installed-Size: 106 kB
Provides: man-browser
Depends: libc6 (>= 2.14), libgcc1 (>= 1:3.0), libglib2.0-0
 (>= 2.12.0), libgtk2.0-0 (>= 2.8.0), libstdc++6 (>= 5),
```

```
 man-db, xterm | x-terminal-emulator Suggests: gv,
 man2html, httpd, sensible-browser, evince
Tag: implemented-in::c, interface::graphical,
 interface::web, interface::x11, role::program,
 uitoolkit::gtk, use::browsing, use::viewing, web::cgi,
 works-with-format::html, works-with-format::man,
 works-with::text, x11::application
Download-Size: 34,3 kB
APT-Manual-Installed: yes
APT-Sources: http://ftp.de.debian.org/debian unstable/main
 amd64 Packages
Description: small man(1) front-end for X
 Gman is a simple front-end for the manual page system. The
 most basic job of gman is to build a database for all the
 man pages and display them (or part of them) on the screen.
 When user decides to read a man page, gman will launch an
 external viewer to display the manual page. More than one
 external viewer windows can be launched at the same time.
[...]
```

All installable versions of the package (depending on the sources.list) can be listed as follows:

```
user1@pc1:~$ apt list gman
listing... Done
gman/unstable,now 0.9.3-5.3 amd64 [installed]
```

The command **aptitude** (in the terminal) opens the program of the same name in an ncurses environment. It is operated with the keyboard or mouse and offers various functions which can be reached via the upper menu bar. The use of APT or Aptitude is a matter of taste, but Aptitude is often "too smart" for the speed of Debian Unstable.

```
 Actions  Undo  Package  Resolver  Search  Options  Views  Help
 C-T: Menu  ?: Help  q: Quit  u: Update  g: Preview/Download/Install/Remove Pkgs
 aptitude 0.8.13 @ pcl
 i A   git-man                                        1:2.29.2-1     1:2.29.2-1
 i     gitmagic                                       20160304-1.2   20160304-1.2
 i     gman                                           0.9.3-5.3      0.9.3-5.3
 i A   gnuplot-data                                   5.4.0+dfsg1-1  5.4.0+dfsg1-1
 i A   info                                           6.7.0.dfsg.2-5 6.7.0.dfsg.2-5
 i A   install-info                                   6.7.0.dfsg.2-5 6.7.0.dfsg.2-5
 i A   kdoctools5                                     5.74.0-2       5.74.0-2
 i A   khelpcenter                                    4:20.04.2-1    4:20.04.2-1
 i     libreoffice-help-de                            1:7.0.3-4      1:7.0.3-4
 i     man-db                                         2.9.3-2        2.9.3-2
 i     man2html                                       1.6g-12        1.6g-12
 i A   man2html-base                                  1.6g-12        1.6g-12
 i     mandoc                                         1.14.4-1       1.14.4-1
 i     manpages                                       5.09-2         5.09-2
 i     manpages-de                                    4.1.0-1        4.1.0-1
 small man(1) front-end for X
 Gman is a simple front-end for the manual page system. The most basic job of gman is to build
 a database for all the man pages and display them (or part of them) on the screen. When user
 decides to read a man page, gman will launch an external viewer to display the manual page.
 More than one external viewer windows can be launched at the same time.

 The default manual page viewer is a terminal window with the original man(1). It can also
 launch gv, evince, or a link to a CGI script which utilizes man2html, for viewing manual
 pages using a web browser.

 There is an index search function to look for the man pages that one needs. It's simple, but
 it's useful.
 Tags: implemented-in::c, interface::graphical, interface::web, interface::x11,
       role::program, uitoolkit::gtk, use::browsing, use::viewing, web::cgi,
       works-with-format::html, works-with-format::man, works-with::text,
       x11::application
```

Graphical package search

The program **`packagesearch`** is very useful to search for suitable programs. Mostly packagesearch is not installed automatically; therefore:

```
apt update
apt install packagesearch
```

After the first start of packagesearch you have to select "apt" in *"Packagesearch"* > *"Preferences"* and occasionally an info window appears, which criticizes the absence of deborphan. Please use the information from deborphan with utmost caution.

Packagesearch is not intended to be used for installing files/packages, but only as a graphical search engine. Upgrading and reinstalling files without first quitting X can cause problems (see above).

The following search criteria are available:

- pattern (general search query)
- tags (search based on debtags)
- files (file names)

- installed status
- orphaned packages

In addition, a lot of information about Debian packages is provided, including which files are bundled in a package. More detailed information about using package-search can be found at *"Help" > "Contents"*. Currently the user interface of packagesearch is English only.

A complete description of the APT system can be found in Debian's APT-HOWTO.

Last edited: 2024/03/22

7.7   Local APT mirror

Apt-Cacher, a proxy server for Debian packages

Apt-Cacher is a proxy server that allows multiple local computers access to a De-
bian package cache.
The packages requested for installation from a computer at the cache only need
to be loaded once from Debian mirror servers, no matter how many devices need
these packages. This saves network bandwidth, increases speed for users, and
reduces load on the mirrors.

For users who own multiple PCs and want to conserve bandwidth and download
volume while increasing the speed of system updates, apt-cacher is the ideal so-
lution to achieve all these goals.

Apt-Cacher is not a universal proxy server. Whoever tries this anyway will experi-
ence some unpleasant surprises in the network.

Prerequisites

- a PC on which the local APT proxy server will be set up
- 6 GB free disk space for the cache on the server
- LAN connection to the other devices

Apt-Cacher Setup

The setup for apt-cacher is done in two steps.
First you install apt-cacher on the PC selected as APT proxy server, and then you
configure all client PCs to use the APT proxy server.

7.7.1   Install server

After an `apt update` the necessary packages are installed with the following com-
mand:

```
# apt install apt-cacher
[...]
The following NEW packages will be installed:
  apt-cacher ed libberkeleydb-perl
```

```
  libcompress-raw-bzip2-perl libcompress-raw-lzma-perl
  libcompress-raw-zlib-perl libfilesys-df-perl
  libio-compress-lzma-perl libio-compress-perl
  libio-interactive-perl libio-interface-perl
  libipc-shareable-perl libnetaddr-ip-perl libsocket6-perl
  libsys-syscall-perl libww-curl-perl
0 updated, 16 reinstalled, 0 to remove and 0 not updated.
992 kB of archives need to be downloaded.
After this operation, 3,205 kB of additional disk space will ↙
   be used.
Do you want to continue? [Y]
```

During the installation of apt-cacher the basic configuration is done automatically.
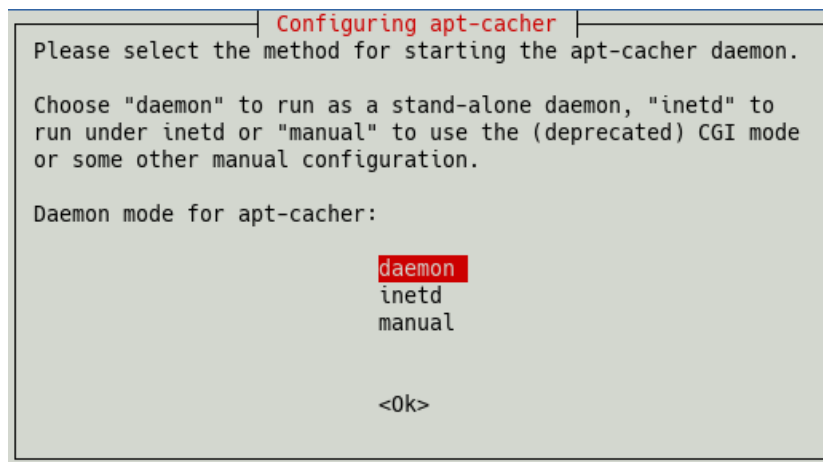
Keep and confirm the recommended daemon mode *"daemon"*.

The cache, where all downloaded packages will be stored in the future, is located in

`/var/cache/apt-cacher/`

and the configuration files in

`/etc/apt-cacher/`.

Server configuration

We change to the directory `/etc/apt-cacher/` and edit the file `apt-cacher.⤢conf`.

```
# cd /etc/apt-cacher
/etc/apt-cacher# mcedit apt-cacher.conf
```

Now, somewhere around line 160, we look for the directive *"allowed_hosts"*. Remove the comment sign (*"#"*) at the beginning of the line to allow clients to contact the APT proxy server.
For security reasons, we replace the wildcard character (*"*"*), which allows everyone to access, with the IP addresses of the clients.

```
#allowed_hosts = *
```

For example, change it to

```
allowed_hosts = '192.168.3.10-20'
```

Of course, the IP addresses must be adapted to your own circumstances. Explanations of the syntax are in the file immediately before the directive.
If there is a DHCP server operating in your own network, it is necessary to assign a fixed IP to the APT proxy server, e.g. *"192.168.3.5"*.

In the following, we need to specifiy the *"user"* and the *"group"* which the daemon runs with and the port on which the daemon listens (all of them in the `apt-cacher⤢.conf` file):

```
group = www-data
user = www-data
daemon_port = 3142
```

These are the default values, which we do not change. After saving the file we exit mcedit.

If a cache directory other than `/var/cache/apt-cacher/` is to be used, the owner and file permissions must be checked and adjusted (`chmod 644` for the files).

To make sure that the APT proxy server starts automatically every time the server is booted, we issue the following command:

```
# systemctl enable apt-cacher.service
```

The APT proxy server is now also restarted and thus the changed configuration is read in.
We check if it is active and listening on port 3142.

```
# ss -tl | grep 3142
LISTEN 0 4096 0.0.0:3142 0.0.0.0:
```

Everything is fine with this output.

Import of existing .deb's

Apt-Cacher now has an import script that imports Debian archives already present on the PC. It saves downloading the packages again. We give the existing archive directory to the call:

```
# /usr/share/apt-cacher/apt-cacher-import.pl /var/cache/apt/↵
   archives/
```

Called with `-h`, we get usage instructions and a listing of all options.

7.7.2 Client configuration

The clients accessing the APT proxy server require only minor configuration.

First we create the file `30proxy` in the directory `/etc/apt/apt.conf.d/` which instructs apt to use the server. Here we use the above mentioned IP of the server. Please adapt the IP to your own circumstances.

```
# echo "Acquire::http { proxy "http://192.168.3.5:3142"; };" ↵
   > /etc/apt/apt.conf.d/30proxy
```

Next, we change the addresses of the download mirrors in the directory `/etc/↵ apt/sources.list.d/` from "https" to "http" within the files `debian.list`, `extra↵ .list`, and `fixes.list`.

Using "https" is possible: On the one hand it requires some configuration effort, and on the other hand it is not necessary at the moment because all download mirrors still accept "http".

A subsequent

```
# apt update
```

should run without error messages.

The first call of **# apt full-upgrade** on a client loads all new packages into the cache of the APT proxy server. Thus, this process takes the same amount of time as before. The further accesses of the clients make use of the cache and then run substantially faster, without requiring bandwidth again.

Last edited: 2022/04/05

7.8 Nala package management

More user-friendly and powerful than APT

Nala is a command line frontend for the APT package manager. It uses the `python-apt` API instead of the APT libraries to manage packages. The goal of Nala is to provide a clearer and more user-friendly display of the current package inventory as well as the requested actions and their execution. It also intends to speed up package download.

Nala uses many APT commands such as `install`, `remove`, `purge`, `update`↙ , `show`, and `search`. It also implements the `history` command to see past transactions and allow the user to undo them, and the `fetch` command that displays a list of the fastest mirror servers to choose from. By default, Nala speeds up downloads by fetching three packets at a time from a server. The limit of three connections per mirror exists to minimize the load on the mirrors.

7.8.1 Use Nala

As of siduction 2022.1.0, Nala is installed automatically and can be used immediately. It is not mandatory to use Nala, you can switch between APT and Nala at will. A look at the manpage **man nala** should be mandatory. Before using it, we strongly recommend making two changes in the configuration file `/etc/nala/nala.conf`.

First, we change the value for the `auto_remove` configuration option to `false` as shown in the following listing:

```
# Set to false to disable auto auto-removing
auto_remove = false
```

Secondly, the value `full_upgrade = false`, this must be changed to `true` to automatically perform a `full-upgrade`.

```
# Set to true to make full-upgrade the default
full_upgrade = true
```

The reason for both changes is that siduction is based on *GNU Linux debian unstable/sid* and this base changes daily and therefore needs to be updated regu-

larly. When upgrading from *debian sid*, occasionally a situation may arise where significant parts of the system should be removed. With the option `auto_remove`↙ `= true` we have no way to research, check and decide for ourselves if or which packages to remove. Even in normal operation packages should not be removed with `auto_remove`, but only after a visual check.

If in an exceptional case it is necessary to `upgrade` the system without a `full-`↙ `upgrade`, `nala` can be called with the following command, **`nala upgrade --no`↙ `-full`**.

7.8.2   Commands analogous to APT

Many of the commands known from APT are identical in Nala. By default, Nala always expects confirmation before performing a requested action that changes the system.

- **`nala update`**
  Updates the package information of the configured package sources.

- **`nala install <package>`**
  Installs the named package into our system.

- **`nala remove <package>`**
  Removes the named package from our system.

- **`nala purge <package>`** or **`nala remove --purge <package>`**
  Removes the named package with its configuration files from our system.

- **`nala upgrade`**
  Runs `update` followed by `upgrade`. (default setting in `/etc/nala/nala.`↙ `conf` file)
  Runs `update` followed by `upgrade --full`. (With the above changes in the file `/etc/nala/nala.conf`)

The user-friendly formatting of the output in the terminal facilitates the overview, as the example shows.
(To gain root privileges, *"doas"* was used in the command.)

```
user1@pc1:~$ doas nala install yapf3
=========================================================
 Installing
=========================================================
  Package:              Version:                       Size:
  python3-yapf          0.32.0-1                       133 KB
  yapf3                 0.32.0-1                        30 KB


=========================================================
 Summary
=========================================================
 Install 2 Packages

 Disk space required  892 KB

Do you want to continue? [Y/n] y
  ┌─ Installing Packages ─────────────────────────────
  Unpacking:  python3-yapf (0.32.0-1)
  Unpacking:  yapf3 (0.32.0-1)
  Setting up: python3-yapf (0.32.0-1)
  Setting up: yapf3 (0.32.0-1)
  Processing: triggers for runit (2.1.2-50)
  Processing: triggers for man-db (2.11.0-1+b1)

  ✓ Running dpkg …  ──────────────── 100.0% • 0:00:00 • 5/5
  └──────────────────────────────────────────────────

Finished Successfully
```

In the first part of the output we get a list of the packages to be installed with the indication of their versions and size. After confirmation, the second part lists the actions performed.

7.8.3   Commands that APT does not include

"fetch" command

The `nala fetch` command, run without any other options, automatically determines the distribution and release of our installation, searches for the fastest mirror servers, lists them for interactive selection, and, after selecting one or more servers, creates the file `/etc/apt/sources.list.d/nala-sources.list`.

The `-c, --country` option limits the search using the ISO country code. Multiple specifications of the option are allowed.
The `--non-free` option adds contrib and non-free components to the file.

During download, up to three packages are fetched from the server simultaneously.

"history" command

The `nala history` command, called without a subcommand, shows a summary of all actions performed with Nala. Each line corresponds to an action and contains the ID, the command, the timestamp, the number of packages changed, and the user who requested the action. Actions performed by other programs are not recorded.

```
user1@pc1:~$ nala history
 ID    Command                Date and Time            Altered    Requested-By
 1     upgrade bash busybox …  2022-11-07 17:41:13 CET       72    root (0)
 2     purge libbpf0           2022-11-08 13:41:10 CET        1    root (0)
 3     install yapf3           2022-11-09 17:10:37 CET        2    user1 (1000)
user1@pc1:~$ _
```

Details about an action from the history are shown by the same command with the attached subcommand `info <ID>`.

```
user1@pc1:~$ nala history info 3
===========================================================
 Installed
===========================================================
  Package:          Version:                         Size:
  python3-yapf      0.32.0-1                         133 KB
  yapf3             0.32.0-1                          30 KB


===========================================================
 Summary
===========================================================
 Installed 2 Packages
```

If we now want to undo the installation of *"yapf3"* with its dependencies, in our case *"python3-yapf"*, we use the subcommand `undo <ID>` for this.
(Again, **user1** gets root privileges by using *"doas"*.)

```
user1@pc1:~$ doas nala history undo 3
==============================================================
 Removing
==============================================================
  Package:             Version:                          Size:
  python3-yapf         0.32.0-1                         849 KB
  yapf3                0.32.0-1                          43 KB


==============================================================
 Summary
==============================================================
 Remove 2 Packages

 Disk space to free   892 KB

Do you want to continue? [Y/n] y
┌─ History Undo 3 ─────────────────────────────────────────┐
│Removing:    yapf3 (0.32.0-1)                              │
│Removing:    python3-yapf (0.32.0-1)                       │
│Processing: triggers for runit (2.1.2-50)                 │
│Processing: triggers for man-db (2.11.0-1+b1)             │
│                                                          │
│ ✓ Running dpkg …  ─────────────  100.0% • 0:00:00 • 5/5  │
│                                                          │
└──────────────────────────────────────────────────────────┘
Finished Successfully
```

In the first part of the output, we see the packages to be removed with the indication of their versions and size. After confirmation, the second part lists the actions performed.

If we change our mind and want to use the packages again, the `nala history↙ redo <ID>` command will help us to perform the action again. The `nala ↙ history clear <ID>` command can be used to remove entries from the history, `nala history clear --all` removes all entries.

In the Nala version 0.11.1 described here, the subcommands `undo <ID>` and `redo <ID>` currently only support the actions Install or Remove. In a future version, which will be based on the Rust programming language, it should be possible to roll back complete dist upgrades.

Last edited: 2023-10-15

7.9   Kernel Upgrade

siduction provides the following kernels:

- **linux-image-siduction-amd64 + linux-headers-siduction-amd64** - Linux kernel for 64-bit PCs with AMD64 or Intel 64 CPU
- 32 bit kernel are not provided anymore. Here you can use the Debian kernel or, alternatively, the Liquorix kernel (https://liquorix.net/).

The siduction kernels are located in the siduction repository as .deb and are automatically included in a system update, provided that the metapackages for image and headers are installed.

7.9.1   Kernel Update without System Update

1. updating the package database:

```
apt update
```

2. installation of the current kernel:

```
apt install linux-image-siduction-amd64 linux-headers-↲
   siduction-amd64
```

3. reboot of the computer to load the new kernel

If you encounter problems with the new kernel, you can choose an older kernel after rebooting.

7.9.2   Modules

The kernel usually comes with all the required kernel modules. For 3rd party modules, dkms is recommended in siduction. For this it is necessary to install the package `build-essential`. Since 3rd party modules are often non-free modules, it is necessary to make sure that contrib and non-free are enabled in the sources.

7.9.3 Removing old kernels

After successfully installing a new kernel, old kernels can be removed. However, it is recommended to keep old kernels for a few days. If problems occur with the new kernel, you can boot into one of the old kernels listed in the grub startup screen.

To remove old kernels the script `kernel-remover` is installed:

```
kernel-remover
```

Last edited: 2022/04/05

7.10   Systemd - the system and services manager

Note:
The following general introduction to systemd was mainly taken from the extensive systemd manpages.

systemd is a system and service manager that runs as the first process (as PID 1) at system startup and thus acts as an **init system**, booting the system and managing services at the application level.
It is lead developed by Red Hat developers Lennart Poettering and Kay Sievers.

In Debian, the introduction of systemd as the default init system was discussed long, controversially, and emotionally until the Technical Committee voted in favor of systemd in February 2014.

siduction has been using systemd as default init system since release 2013.2 "December".

7.10.1   Concept of systemd

Systemd provides a dependency system between different *"units"* of 11 different types (see below). Units encapsulate various objects relevant to system startup and operation.
Units can be *"active"* or *"inactive"*, as well as in the process of *"activation"* or *"deactivation"*, i.e. between the two former states. A special state *"failed"*, which is very similar to *"inactive"*, is also available. When this state is reached, the cause is logged for later inspection. See the manual page systemd-journal.
With systemd, many processes can be controlled in parallel because the unit files declare possible dependencies and systemd adds required dependencies automatically.

The units managed by systemd are configured using unit files.
The unit files are pure text files in INI format, divided into different sections. This makes their contents easy to understand and edit without knowledge of a scripting language. All unit files must have a section corresponding to the unit type and may contain the generic sections [Unit] and [Install].

The manual page systemd unit file explains the basic structure of the unit files, as well as many options of the generic sections [Unit] and [Install].

7.10.2　Unit types

Before we turn to the unit types, it is advisable to read the manual page systemd unit file to understand the operation of the generic sections and their options. The following unit types are available, and if a link is available, it will take you to a more detailed description in our manual:

1. **Service units** (systemd.service) start and control daemons as well as the processes that make them up.

2. **Socket units** (systemd.socket) encapsulate local IPC or network sockets in the system (useful for socket-based activation).

3. **Target units** (systemd.target) are useful for grouping units. They also provide synchronization points known as runlevels during system startup.

4. **Device units** (systemd.device) expose kernel devices (all block and network devices) in systemd and can be used to implement device-based activation.

5. **Mount units** (systemd.mount) control mount points in the file system.

6. **Automount units** (systemd.automount) provide self-mount capabilities for on-demand file system mounts and parallelized boot.

7. **Timer units** (systemd.timer) are useful for triggering the activation of other units based on timers.

8. **Swap units** (systemd.swap) are similar to mount units and encapsulate memory swap partitions or files of the operating system.

9. **Path units** (systemd.path) can be used to enable other services when file system objects change or are modified.

10. **Slice units** (systemd.slice) can be used to group units that manage system processes (such as service and scope units) in a hierarchical tree for resource management reasons.

11. **Scope units** (systemd.scope) are similar to service units, but manage foreign processes instead of starting them as well.

7.10.3   Systemd in the file system

The unit files installed by the distribution's package manager are located in the `/lib/systemd/system/` directory. Self-created unit files are placed into the directory `/usr/local/lib/systemd/system/`. (If necessary, create the directory beforehand with the command **`mkdir -p /usr/local/lib/systemd/system/`**.) You can control the status (enabled, disabled) of a unit via symlink in the directory `/etc/systemd/system/`.
The directory `/run/systemd/system/` contains unit files created at runtime.

7.10.4   Further functions of systemd

Systemd provides other functions as well. One of them is logind as a replacement for the no longer maintained *ConsoleKit*. With this, systemd controls sessions and power management. Last but not least systemd offers a lot of other possibilities like spinning up a container (similar to a chroot) using systemd-nspawn and many more. A look at the link list on Freedesktop allows further discoveries, including the extensive documentation on systemd by lead developer Lennart Poettering.

7.10.5   Handling services

One of the jobs of systemd is to start, stop, or otherwise control services. For this purpose the command `systemctl` can be used.

- systemctl –all - lists all units, active and inactive.
- systemctl -t [NAME] - lists only units of the specified type.
- systemctl list-units - lists all active units.
- systemctl start [NAME…] - starts one or more units.
- systemctl stop [NAME…] - stops one or more units.
- systemctl restart [NAME] - stops a unit and restarts it immediately. Used e.g. to re-read the changed configuration of a service.
- systemctl status [NAME] - shows the current status of a unit.

- systemctl is-enabled [name] - shows only the value "enabled" or "disabled" of a unit's status.

The following two commands integrate or remove the unit based on the configuration of its unit file. Dependencies to other units are taken into account and default dependencies are added if necessary so that systemd can execute the services and processes without errors.

- systemctl enable [NAME] - adds a unit to systemd.
- systemctl disable [NAME] - removes a unit from systemd.

It is often necessary to perform `systemctl start` and `systemctl enable` on a unit to make it available both immediately and after a reboot. Both options are combined by the command:

- systemctl enable –now [NAME]

The following are two commands whose function is described on our manual page systemd-target.

- systemctl reboot - performs a reboot.
- systemctl poweroff - shuts down the system and turns off the power if technically possible.

Example
Using Bluetooth we demonstrate systemd's functionality.
First the status query in short format:

```
# systemctl is-enabled bluetooth.service
enabled
```

Now we search for the unit files, combining `systemctl` with `grep`:

```
# systemctl list-unit-files | grep blue
bluetooth.service enabled enabled
dbus-org.bluez.service alias -
bluetooth.target static -
```

Then we disable the unit *"bluetooth.service"*.

```
# systemctl disable bluetooth.service
Synchronizing state of bluetooth.service with SysV service
 script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable ⤸
   bluetooth
Removed /etc/systemd/system/dbus-org.bluez.service.
Removed /etc/system/system/bluetooth.target.wants/bluetooth.⤸
   service.
```

In the output you can clearly see that the links (not the unit file itself) have been removed. This means that the *"bluetooth.service"* will no longer start automatically when booting the PC/laptop. We check the status after a reboot.

```
# systemctl is-enabled bluetooth.service
disabled
```

To disable a unit only temporarily, we use the command

```
# systemctl stop <unit>
```

This will keep the configuration in systemd. We reactivate the unit with the corresponding `start` command.

7.10.6   Sources systemd

```
man systemd
man systemd.unit
man systemd.syntax
```

Last edited: 2024/08/30

7.11   systemd unit file

The basic and introductory information about systemd can be found on the manual page systemd-start.

On the following manual page we explain the structure of the **unit files** and the generic sections *[Unit]* and *[Install]*.

The unit file is a plain text file in INI format. It contains configuration statements of the type *key=value* in various sections. Empty lines and those starting with "#" or ";" are ignored. All unit files must contain a section corresponding to the unit type. The generic sections [Unit] at the beginning and [Install] at the end of the file are optional, but the [Unit] section is strongly recommended.

7.11.1   Loading path of the unit files

The output shows the order of the directories from which the unit files are loaded.

```
# systemd-analyze unit-paths
/etc/systemd/system.control
/run/systemd/system.control
/run/systemd/transient
/run/systemd/generator.early
/etc/systemd/system
/etc/systemd/system.attached
/run/systemd/system
/run/systemd/system.attached
/run/systemd/generator
/usr/local/lib/systemd/system
/lib/systemd/system
/usr/lib/systemd/system
/run/systemd/generator.late
```

Unit files found in directories listed earlier override files with the same name in directories further down the list. For example, a file in `/etc/systemd/system`↵ overrides one with the same name in `/lib/systemd/system`.

Only some of the previously listed directories exist in siduction by default:

- `/lib/systemd/system/`

  contains system units installed by the distribution's package manager and any unit files created by the administrator.

- `/etc/systemd/system/`

  contains symlinks to unit files in `/lib/systemd/system/` for enabled units and administrator-created unit files, if any.

- `/usr/local/lib/systemd/system/`

  must be created and is meant to contain administrator-created unit files.

- `/run/systemd/`

  contains runtime units and dynamic configuration for volatile units. For the administrator, this directory has informal value only.

We recommend storing your own unit files in `/usr/local/lib/systemd/system↲ /`.

7.11.2 Activating the unit file

To make the configuration of a unit accessible to systemd, the unit file must be activated. This is done with the call:

```
# systemctl daemon-reload
# systemctl enable --now <UNIT_FILE>
```

The first command reloads the complete daemon configuration, the second one starts the unit immediately (option `--now`) and incorporates it into systemd so that it is executed every time the PC is rebooted.
The command

```
# systemctl disable <UNIT_FILE>
```

causes it to stop running every time the PC is rebooted. However, it can still be started manually with the command **systemctl start <UNIT_FILE>** and stopped with **systemctl stop <UNIT_FILE>**.

If a unit file is empty (i.e. has size 0) or is a symlink on `/dev/null`, its configuration will not be loaded and it will appear with the load state "masked" and cannot be activated. This is an effective way to completely disable a unit and also make it impossible to start it manually.

7.11.3   Sections of the unit file

The unit file usually consists of the [Unit] section, the type-specific section, and the [Install] section. The type-specific section is included as a suffix in the file name. For example, a unit file that configures a timer always has the extension ".timer" and must contain [Timer] as the type-specific section.

7.11.3.1   Section Unit   This section contains general information about the unit, defines dependencies to other units, evaluates conditions, and takes care of the enumeration in the boot process.

1. General options

   a. `Description=`

      identifies the unit by a human readable name, which is used by systemd as a description for the unit and thus appears in the systemjournal ("Starting *description*…") and can be used as a search pattern there.

   b. `Documentation=`

      is a reference to a file or web page that references documentation for this unit or its configuration, e.g. "Documentation=man:cupsd(8)" or "Documentation=http://www.cups.org/doc/man-cupsd.html".

2. Binding dependencies to other units

   a. `Wants=`

      Units listed here are started with the configured unit.

   b. `Requires=`

      Similar to Wants=, but declares a stronger binding to the listed Units. When this unit is activated, the listed units are also activated. If activation of one of the other units fails **and** the order dependency

`After=` is set on the failed unit, then that unit will not be started.

If one of the other units becomes inactive, this unit will remain active. Only if one of the other units is stopped, this unit will also be stopped.

c. `Requisite=`

Similar to Requires=. The start of this unit will fail immediately if the units listed here have not been started yet. Requisite= should be combined with the order dependency `After=` to ensure that this unit is not started before the other unit.

d. `BindsTo=`

BindsTo= is the strongest dependency type: it causes, in addition to the properties of Requires=, that the bound unit must be in active state for this unit to be active.

When the bound unit is stopped or in an inactive state, this unit will always be stopped.

To prevent the start of this unit from failing when the bound unit is not (or not yet) in an active state, BindsTo= is best combined with the order dependency `After=`.

e. `PartOf=`

works similar to Requires=, but limited to stopping and restarting units. When systemd stops or restarts the units listed here, the action is forwarded to that unit.

This is a one-way dependency. Changes to this Unit do not affect the units listed.

f. `Conflicts=`

Declares negative request dependencies. It is possible to specify a space-separated list.

Conflicts= causes the listed unit to stop when this unit starts and vice versa.

Since Conflicts= does not include an order dependency, an *After=* or `Before=` dependency must be declared to ensure that the conflicting unit is stopped before the other unit is started.

3. Order dependencies to other units

   a. `Before=`

      This setting configures order dependencies between units. Before= ensures that the listed unit will only start after the configured unit has finished starting.
      Specifying a space-separated list is possible.

   b. `After=`

      This setting ensures the opposite of Before=. The listed unit must have been completely started before the configured unit is started.

   c. `OnFailure=`

      specifies units to be activated when this unit takes the "failed" state.

4. Conditions
   Unit files can also contain a set of conditions.
   Before starting the unit, systemd will check if the specified conditions are true.
   If not, the start of the unit (almost without output) will be skipped.
   Failing conditions will not cause the unit to enter the "failed" state.
   In case multiple conditions are specified, the unit will be executed if all of them are true.
   In this section, we list only conditions that we think are useful for user-created units, because many conditions are used to skip units that do not apply on the local system.
   The command `systemd-analyze verify <UNIT_FILE>` can be used to test conditions.

   a. `ConditionVirtualization=`

      checks if the system is running in a virtualized environment and optionally tests if it is a specific implementation.

   b. `ConditionACPower=`

      checks if the system is on the mains or running solely on battery power at the time the unit is activated.

c. ConditionPathExists=

   checks for the existence of a file. With an exclamation mark ("!") in front
   of the path, the test is negated.

d. ConditionPathExistsGlob=

   As before, except that a search pattern is specified. With an exclamation
   mark ("!") in front of the path, the test is negated.

e. ConditionPathIsDirectory=

   tests for the existence of a directory. With an exclamation mark ("!") in
   front of the path, the test is negated.

f. ConditionPathIsSymbolicLink=

   checks if a given path exists and is a symbolic link. With an exclamation
   mark ("!") in front of the path, the test is negated.

g. ConditionPathIsMountPoint=

   checks if a given path exists and is a mount point. With an exclamation
   mark ("!") in front of the path, the test is negated.

h. ConditionPathIsReadWrite=

   checks if the underlying file system is readable and writable. With an
   exclamation mark ("!") in front of the path, the test is negated.

i. ConditionDirectoryNotEmpty=

   checks if a given path exists and is a non-empty directory. With an ex-
   clamation mark ("!") in front of the path the test is negated.

j. ConditionFileNotEmpty=

   checks if a given path exists and refers to a normal file with a non-
   zero size. With an exclamation mark ("!") in front of the path, the test
   is negated.

k. ConditionFileIsExecutable=

   checks if a given path exists and refers to a normal file marked as ex-
   ecutable. With an exclamation mark ("!") in front of the path, the test is
   negated.

For full documentation on all options of the "[Unit]" section, please refer to `man` ↙
`systemd.unit`.

7.11.3.2   Type-specific section   This section contains the special options of the
eleven possible types. Detailed descriptions can be found in the linked manual
pages, or on the respective manpage.

- [Service] configures a service.

- [Socket] configures a socket (`man systemd.socket`).

- [Device] configures a device (`man systemd.device`).

- [Mount] configures a mount point.

- [Automount] configures a self-mount point.

- [Swap] configures a swap file or partition (`man systemd.swap`).

- [Target] configures a start target.

- [Path] configures a monitored file path.

- [Timer] configures a timer controlled and monitored by systemd.

- [slice] configures a resource management slice (`man systemd.slice`).

- [Scope] configures a group of externally created processes (`man systemd`↙
  `.scope`).

7.11.3.3   Install section   Unit files may contain this section.
The options in the [Install] section are related by the **`systemctl enable <`**↙
`UNIT_FILE>` and **`systemctl disable <UNIT_FILE>`** commands during installa-
tion of a unit.
Unit files without [Install] section can be started manually with the command
`systemctl start <UNIT_FILE>` or from another unit file.

Description of options:

- `Alias=`

  A list of additional names under which this unit should be installed. The names listed here must have the same extension as the unit file.

- `WantedBy=`

  This option can be used multiple times or contain a space-separated list.
  A symbolic link is created in the `.wants/` directory of each of the listed units during installation. This adds a dependency of the type `Wants=` from the listed unit to the current unit. The main result is that the current unit is started when the listed unit is started.
  Behaves like the Wants= option in the [Unit] section.

  Example:
  WantedBy=graphical.target

  This tells systemd to launch the unit when starting graphical.target (formerly "init 5").

- `RequiredBy=`

  This option can be used multiple times or contain a space-separated list.
  A symbolic link is created in the `.requires/` directory of each of the listed units during installation. This adds a dependency of type `Requires=` from the listed unit to the current unit. The main result is that the current unit is started when the listed unit is started.
  Behaves like the Requires= option in the [Unit] section.

- `Also=`

  specifies additional units to be installed/uninstalled when this unit is installed/uninstalled.

- `DefaultInstance=`

  This option has effect only for template unit files.
  Declares which instance of the unit should be released. The specified string must be suitable for identifying an instance.

Hint: To verify the configuration of a unit file, the **systemd-analyze verify <⤶ UNIT_FILE>** command is suitable.

7.11.4   Example cupsd

cupsd, the job scheduler for the Common UNIX Printing System, is controlled by systemd through its three unit files `cups.socket`, `cups.service`, and `cups.↙ path` and is well suited to illustrate the dependencies.

Here are the three files.

```
File /lib/system/system/cups.service:


[Unit]
Description=CUPS Scheduler
Documentation=man:cupsd(8)
After=network.target sssd.service ypbind.service nslcd.↙
    service
Requires=cups.socket
    After=cups.socket
      (not in the file, because implicitly present)
    After=cups.path
      (not in the file, because implicitly present)


[Service]
ExecStart=/usr/sbin/cupsd -l
Type=notify
Restart=on-failure


[Install]
Also=cups.socket cups.path
WantedBy=printer.target
```

```
File /lib/system/system/cups.path:


[Unit]
Description=CUPS Scheduler
PartOf=cups.service
    Before=cups.service
      (not in the file, because implicitly present)
```

```
[Path]
PathExists=/var/cache/cups/org.cups.cupsd

[Install]
WantedBy=multi-user.target
```

```
File /lib/system/system/cups.socket:

[Unit]
Description=CUPS Scheduler
PartOf=cups.service
    Before=cups.service
      (not in the file, because implicitly present).

[Socket]
ListenStream=/run/cups/cups.sock

[Install]
WantedBy=sockets.target
```

The [Unit] section

contains the same description for all three files. The files `cups.path` and `cups`↙
`.socket` additionally contain the binding dependency `PartOf=cups.service`↙
, which means that these two units are stopped or restarted depending on
cups.service.

The socket unit as well as the path unit include the order dependency `Before`↙
`=` to their service unit with the same name. Therefore it is not necessary to
include the order dependencies After=cups.socket and After=cups.path in the
cups.service unit (see below the output of *"systemd-analyze dump"* with the nota-
tion *"destination-implicit"*). The effect of both dependencies together is that regard-
less of which unit starts first, all three units will always start, and the cups.service
unit will only start after the cups.path unit and the cups.socket unit have success-
fully started.

We get the units' complete configuration with the command **systemd-analyze**↙
 dump, which prints a very, very long list (> 32000 lines) of the systemd server
state.

```
# systemd-analyze dump
[...]
-> Unit cups.service:
    Description: CUPS Scheduler.service
    [...]
    WantedBy: printer.target (destination-file)
    ConsistsOf: cups.socket (destination-file)
    ConsistsOf: cups.path (destination-file)
    Before: printer.target (destination-default)
    After: cups.socket (destination-implicit)
    After: cups.path (destination-implicit)
[...]
-> Unit printer.target:
    Description: Printer
    [...]
    Wants: cups.service (origin-file)
    After: cups.service (origin-default)
[...]
```

The [Install] section

of the cups.service unit contains the option `Also=cups.socket cups.path`↙
, i.e. the instruction to install these two units as well and all three units have different
`WantedBy=` options:

- cups.socket: WantedBy=sockets.target
- cups.path: WantedBy=multi-user.target
- cups.service: WantedBy=printer.target

To understand why different values are used for the *"WantedBy="* option, we need
additional information, which we can obtain with the **systemd-analyze dot** and
systemd-analyze plot commands.

```
    (enter in a single line)
```

```
$ systemd-analyze dot --to-pattern='*.target'
 --from-pattern='*.target' | dot -Tsvg > targets.svg


$ systemd-analyze plot > bootup.svg
```

The first one gives us a flowchart with the dependencies of the different targets to each other and the second one a graphical listing of the boot process with when a process was started, how much time it took, and its activity state.

From the `targets.svg` and the `bootup.svg` we can see that

1. *sysinit.target*
   is activated and

2. *basic.target*
   will not start until sysinit.target has been reached.

   2.1. *sockets.target*
   is requested by basic.target,

   2.1.1. *cups.socket*
   and all other .socket units are fetched from sockets.target.

   2.2 *paths.target*
   is requested by basic.target,

   2.2.1. *cups.path*
   and all other .path units are fetched from paths.target.

3. *network.target*
   will not start until basic.target has been reached.

4. *cups.service*
   will not start until network.target has been reached.

5. *multi-user.target*
   will not start until network.target has been reached.

6. *multi-user.target*
   is not reached until cups.service has been started successfully. (Strictly

speaking, this is because the cups-browsed.service, which depends on the cups.service, must have been started successfully.)

7. *printer.target*

becomes active only when systemd dynamically generates device units for the printers. For this to happen, the printers must be connected and turned on.

Further above we noted that starting a cups.xxx unit is sufficient to bring in all three units. If we look again at the *"WantedBy="* options in the [Install] section, we have the cups.socket unit being brought in via the sockets.target already during the basic.target, the cups.path unit being brought in during the multi-user.target, and the cups.service being brought in by the printer.target.

Throughout the boot process, the three cups.xxx units are repeatedly requested from systemd for activation. This hardens the cupsd against unforeseen errors, but does not matter to systemd because it does not matter how many times a service is requested if it is in the queue.

Additionally, the printer.target requests the cups.service whenever a printer is newly detected by systemd.

7.11.5   Tools

Systemd includes some useful tools for analyzing, checking, and editing unit files. Please also refer to the man pages `man systemd-analyze` and `man systemctl`.

- edit

```
# systemctl edit <UNIT_FILE>
# systemctl edit --full <UNIT_FILE>
# systemctl edit --full --force <UNIT_FILE>
```

"systemctl edit" opens the selected unit file in the configured editor.

- **systemctl edit <UNIT_FILE>** creates a new directory under `/etc/`↙ `systemd/system/` named `<UNIT_FILE>.d` and in it the file `override`↙ `.conf` which contains only the changes from the original unit file. This

applies to all unit files in the directories entered in the Hirarchy of load paths including `/etc/systemd/system/` downwards.

- **systemctl edit - -full <UNIT_FILE>** creates a new file with the same name in the `/etc/systemd/system/` directory. This applies to all unit files in the directories entered in the Hirarchy of load paths below `/etc/systemd/system/`. Files already existing in the `/etc/systemd⤸ /system/` directory will be overwritten.

- **systemctl edit - -full - -force <UNIT_FILE>** creates a new file in the directory `/etc/systemd/system/`. Without the `--full` option, only an `override.conf` file would be generated in the new directory `/etc/systemd/system/<UNIT_FILE>.d/`, which lacks the associated unit file.

When the editor is terminated, systemd automatically executes the command **systemctl daemon-reload**.

- revert

```
# systemctl revert <UNIT_FILE>
```

reverts the changes made to unit files with `systemctl edit` and `systemctl edit --full`. This does not apply to changed unit files that were already existing in the `/etc/systemd/system/` directory.
In addition, the command undoes the changes made with `systemctl mask`.

- daemon-reload

```
# systemctl daemon-reload
```

reloads the system administrator configuration. This re-runs all generators, reloads all unit files, and rebuilds the entire dependency tree.

- cat

```
$ systemctl cat <UNIT_FILE>
```

prints the contents of the unit file and all associated changes according to the console command `cat`.

- analyze verify

```
$ systemd-analyze verify <UNIT_FILE>
```

checks the configuration settings of a unit file and prints hints. This is a very useful command to check the configuration of self created or changed unit files.

- systemd-delta

```
$ systemd-delta
```

presents in the output unit files and the changes made to them. The keyword at the beginning of the line defines the type of change or configuration. Here is an example:

```
$ systemd-delta --no-pager
[MASKED] /etc/sysctl.d/50-coredump.conf → /usr/lib/sysctl↙
    .d/50-coredump.conf

[OVERRIDDEN] /etc/tmpfiles.d/screen-cleanup.conf → /usr/↙
    lib/tmpfiles.d/screen-cleanup.conf

[MASKED] /etc/systemd/system/NetworkManager-wait-online.↙
    service → /lib/systemd/system/NetworkManager-wait-↙
    online.service

[EQUIVALENT] /etc/systemd/system/tmp.mount → /lib/systemd↙
    /system/tmp.mount

[EXTENDED] /lib/systemd/system/rc-local.service → /lib/↙
    systemd/system/rc-local.service.d/debian.conf
```

```
[EXTENDED] /lib/systemd/system/systemd-localed.service → ↵
    /lib/systemd/system/systemd-localed.service.d/locale-↵
    gen.conf

6 overridden configuration files found.
```

- analyze dump

```
$ systemd-analyze dump > systemd_dump.txt
```

creates the text file *"systemd_dump.txt"* with the complete configuration of all systemd units. The very long text file gives information about all configuration settings of all systemd units and can be easily searched with a text editor and using regex patterns.

- analyze plot

```
$ systemd-analyze plot > bootup.svg
```

creates the file *"bootup.svg"* with the chronological sequence of the boot process. It is a graphical listing of the boot process with the start and end times of all units, what time they took, and their activity states.

- analyze dot (Enter the command on one line).

```
$ systemd-analyze dot --to-pattern='*.target'
  --from-pattern='*.target' | dot -Tsvg > targets.svg

  Color legend: black     = Requires
                dark blue = Requisite
                dark grey = Wants
                red       = Conflicts
                green     = After
```

creates the *"targets.svg"* flowchart that shows the dependencies of the targets used in the boot process. The relationships of the .target units are shown in color for a better overview.

The tools mentioned here represent only a part of the tools shipped with systemd. Please refer to the man pages for full documentation.

7.11.6 Sources systemd-unit file

We recommend to read the following manpages:

```
man systemd.unit
man systemd.syntax
man systemd.device
man systemd.scope
man systemd.slice
man systemd.socket
man systemd.swap
man systemd-analyze
man systemctl
```

Last edited: 2022/04/08

7.12 systemd-service

The basic and introductory information about systemd is contained on the manual page systemd-start. The sections *[Unit]* and *[Install]* concerning all unit files are covered by our manual page systemd unit file.

On this manual page we explain the function of the unit **systemd.service**. The unit file with the ".service" name extension is the most commonly encountered unit type in systemd.

The service unit file must contain a [Service] section that configures information about the service and the process it is monitoring.

7.12.1 Create service unit

We prefer to place self-created unit files in the `/usr/local/lib/systemd/`↙ `system/` directory. (If necessary, create the directory with the command **mkdir**↙ **-p /usr/local/lib/systemd/system/**.) This has the advantage of giving them priority over system units installed by the distribution's package manager, while placing control links and change files created with **systemctl edit <**↙ **UNIT_FILE>** in the directory `/etc/systemd/system/`, which itself has a higher priority. See: Hirarchy of load paths.

7.12.2 Service section

There are over thirty options available for this section, of which we describe particularly frequently used ones here.

```
Type=              PIDFile=
RemainAfterExit=   GuessMainPID=
ExecStart=         Restart=
ExecStartPre=      RestartSec=
ExecStartPost=     SuccessExitStatus=
ExecCondition=     RestartPreventExitStatus=
ExecReload=        RestartForceExitStatus=
ExecStop=          NonBlocking=
ExecStopPost=      NotifyAccess=
```

```
TimeoutStopSec=    RootDirectoryStartOnly=
TimeoutStartSec=   FileDescriptorStoreMax=
TimeoutAbortSec=   USBFunctionDescriptors=
TimeoutSec=        USBFunctionStrings=
RuntimeMaxSec=     Sockets=
WatchdogSec=       BusName=
                   OOMPolicy=
```

- `Type=`
  Defines the process startup type and is therefore one of the most important options.
  The possible values are: simple, exec, forking, oneshot, dbus, notify, or idle. The default simple is used if `ExecStart=` is set, but neither `Type=` nor `BusName=` are.

  - `simple`
    systemd considers a simple type unit as successfully started as soon as the main process specified with `ExecStart=` has been started by *fork*. Then systemd immediately starts subsequent units, regardless of whether the main process can be called successfully.

  - `exec`
    Similar to simple, but systemd waits to start subsequent units until the main process has finished successfully. This is also the time when the unit reaches the "active" state.

  - `forking`
    Here systemd considers the service as started as soon as the process specified with `ExecStart=` branches to the background and the parent system terminates. This type is often used with classic daemons. The option `PIDFile=` should also be specified here so that the system can continue to follow the main process.

  - `oneshot`
    The Type=oneshot option is similar to exec and often used with scripts or commands that do a single job and then exit. However, the service

never reaches the "active" state, but goes from the "activating" to "deactivating" or "dead" state immediately after the main process terminates. Therefore it is often useful to use this option with `RemainAfterExit=`↙ `yes` to reach the "active" state.

– `dbus`

behaves similarly to simple. systemd starts subsequent units after the D-Bus bus name has been obtained. Units with this option implicitly get a dependency on the unit dbus.socket.

– `notify`

The type=notify is very similar to the type simple, with the difference that the daemon sends a signal to systemd when it is ready.

– `idle`

The behavior of idle is very similar to simple. However, systemd delays the actual execution of the service until all active jobs are completed. This type is not useful as a general tool for sorting units, because it is subject to a 5 s timeout, after which the service is executed in any case.

• `RemainAfterExit=`

expects a logical value (default: no) that determines whether the service, even if all its processes have terminated, should be considered active. See Type=oneshot.

• `GuessMainPID=`

expects a logical value (default: yes). systemd uses this option only if `Type`↙ `=forking` is set and `PIDFile=` is not, and then tries to guess the main PID of a service if it cannot determine it reliably. For other types or with `PIDFile=` set, the main PID is always known.

• `PIDFile=`

accepts a path to the service's PID file. For services of `Type=forking` the use of this option is recommended.

- `BusName=`

  The D-Bus bus name under which this service can be reached must be specified here. The option is mandatory for services of `Type=dbus`.

- `ExecStart=`

  contains commands with their arguments that are executed when this unit is started. Exactly one command must be specified, unless the `Type=↙ oneshot` option is set, in which case ExecStart= can be used multiple times. The value of ExecStart= must conform to the rules described in detail in the man page `man systemd.service`.

- `ExecStop=`

  can be used multiple times and contains commands to stop a service started by ExecStart=. The syntax is identical to ExecStart=.

- `ExecStartPre=`, `ExecStartPost=`, `ExecStopPost=`

  are additional commands that are started before or after the command in `ExecStart=` or `ExecStop`. Again, the syntax is identical to ExecStart=. Multiple command lines are allowed and the commands are executed serially one after the other. If one of these commands (not preceded by "-") fails, the unit is immediately considered to have failed.

- `RestartSec=`

  specifies the sleep time before restarting a service. A unit-free integer defines seconds, a specification of "3min 4s" is also possible.
  The type of time value definition applies to all timed options.

- `TimeoutStartSec=`, `TimeoutStopSec=`, `TimeoutSec=`

  define the time to wait for starting or stopping. TimeoutSec= combines the two previously mentioned options.
  TimeoutStopSec= additionally configures the time to wait for each `ExecStop=` command, if any.

- `Restart=`

  configures whether the service should be restarted when the service process terminates, kills itself, or times out. If the process' death is the result of a systemd action, the service will not be restarted.

The allowed values are: no, always, on-success, on-failure, on-abnormal, on-abort, or on-watchdog.

The following table shows the effect of the Restart= setting on the exit reasons.

► Restart= ► ▼ Exit-Grund ▼	always	on success	on failure	on abnormal	on abort	on watchdog
Sauberer Exit	X	X				
Unsauberer Exit	X		X			
Unsauberes Signal	X		X	X	X	
Zeitüberschreitung	X		X	X		
Watchdog	X		X	X		X

The options

`RestartPreventExitStatus=` and `RestartForceExitStatus=`

change this behavior.

Examples

Some self created service units can be found on our manual pages

service-unit for systemd timer,

service-unit for systemd Path,

and with the preferred search engine on the Internet.

7.12.3 Sources systemd-service

```
man systemd.service
```

LinuxCommunity, Create systemd units yourself

Last edited: 2022/04/08

7.13 systemd-mount

The basic and introductory information about systemd can be found on the manual page systemd-start. The sections *[Unit]* and *[Install]* concerning all unit files are covered on our manual page systemd unit file.

On this manual page we explain the function of the systemd units **mount** and **automount**. They are used by systemd to manage mount points for drives and their partitions, which can be accessible both locally and over the network.

The **mount** unit is a configuration file that provides systemd with information about a mount point.

The **automount** unit monitors the file system and activates the .mount unit of the same name if the file system designated therein is available.

For drives and their partitions directly installed in the PC we only use the mount unit. It is enabled and started to mount the drives at each boot.

For network file systems, the mount unit has the advantage of being able to declare dependencies so that the unit only becomes active when the network is ready. Again, we use only the mount unit and activate and start it to mount the network file system at each boot. The mount unit supports all types of network file systems (NFS, SMB, FTP, WEBDAV, SFTP, SSH).

Removable devices, such as USB sticks and network file systems that are not permanently accessible, must always be attached to a .automount unit. In this case, the mount unit must not be activated and should not contain an [Install] section.

mount and automount units must be named after the mount point they control. For example, the mount point "/home/exampleuser" must be configured in a unit file "home-musteruser.mount", or "home-musteruser.automount".

The devices declared in `/etc/fstab` and their mount points are translated into native mount units by systemd in the early boot phase using the systemd-fstab-generator.

7.13.1 Contents of the mount unit

The mount unit has the following options in the mandatory [Mount] section:

- `What=` (mandatory)

  contains the absolute path of the mounted device, e.g., disk partitions such as `/dev/sda8` or a network share such as NFSv4 or Samba.

- `Where=` (mandatory)

  Here you specify the mount point, i.e. the folder where the partition, network drive, or device should be mounted. If it does not exist, it will be created during the mount process.

- `Type=` (optional)

  Here the type of the file system is specified, according to the mount parameter -t.

- `Options=` (optional)

  contains all used options in a comma separated list, according to the mount parameter -o.

- `LazyUmount=` (default: off)

  If set to true, the filesystem will be unmounted as soon as it is no longer needed.

- `SloppyOptions=` (default: off)

  If true, a relaxed evaluation of the options specified in `Options=` is performed and unknown mount options are tolerated. This is equivalent to the mount parameter -s.

- `ReadWriteOnly=` (default: off)

  If false, the file system or device that should be mounted read-write, but could not be mounted successfully, is attempted to be mounted read-only. If true, the process immediately ends with an error if the read-write mount fails. This is equivalent to the -w mount parameter.

- `ForceUnmount=` (default: off).

  If true, unmounting is forced if, for example, an NFS file system is unreachable. This corresponds to the mount parameter -f.

- `DirectoryMode=` (default: 0755)
  The, if necessary, automatically created directories of mount points get the declared file system access mode. Accepts an access mode in octal notation.

- `TimeoutSec=` (default value from the `DefaultTimeoutStartSec=` option in systemd-system.conf).
  Configures the time to wait for the mount command to finish. If a command does not finish within the configured time, the mount is considered to have failed and is shut down again. Accepts a unit-free value in seconds or a duration value such as "5min 20s". Passing "0" will disable the timeout logic.

7.13.2   Contents of automount unit

The automount unit has the following options in the mandatory [Automount] section:

- `Where=` (mandatory)
  This specifies the mount point, i.e. the folder where the partition, network drive, or device is to be mounted. If it does not exist, it will be created during the mount process.

- `DirectoryMode=` (default: 0755)
  The, if necessary, automatically created directories of mount points get the declared file system access mode. Accepts an access mode in octal notation.

- `TimeoutIdleSec=` (default: 0)
  specifies the time of inactivity after which systemd attempts to unmount the file system. Accepts a unitless value in seconds or a duration value such as "5min 20s". The value "0" disables the option.

7.13.3   Examples

Systemd reads the mount point from the name of the mount and automount units. Therefore, they must be named after the mount point they control.
Make sure not to use hyphens "-" in the filenames, because they declare a new subdirectory in the directory tree. Some examples:

- invalid: /data/home-backup
- allowed: /data/home_backup
- allowed: /data/home\x2dbackup

To get an error-free file name for the mount and automount units, we use the `systemd-escape` command in the terminal.

```
$ systemd-escape -p --suffix=mount "/data/home-backup"
  data/home\x2dbackup.mount
```

Disk partition
A partition should be accessible under `/disks/TEST` after every system start.
We create with a text editor the file "disks-TEST.mount" in the directory `/usr/↵ local/lib/systemd/system/`. (If necessary, create the directory beforehand with the command **`mkdir -p /usr/local/lib/systemd/system/`**.)

```
[Unit]
Description=Mount /dev/sdb7 at /disks/TEST
After=blockdev@dev-disk-by\x2duuid-a7af4b19\x2df29d\x2d43bc\↵
    x2d3b12\x2d87924fc3d8c7.target
Requires=local-fs.target
Wants=multi-user.target

[Mount]
Where=/disks/TEST
What=/dev/disk/by-uuid/a7af4b19-f29d-43bc-3b12-87924fc3d8c7
Type=ext4
Options=defaults,noatime

[Install]
WantedBy=multi-user.target
```

Then we activate and start the new mount unit.

```
# systemctl enable --now disks-TEST.mount
```

NFS

The "document-root" directory of an Apache web server in the home network is to be mounted into the home directory of the workstation computer using NFS.

We create the file `home-<user>-www_data.mount` in the `/usr/local/lib/⤸ systemd/system/` directory using a text editor.

Please replace "<user>" with your own name.

```
[Unit]
Description=Mount server1/var/www/ using NFS
After=network-online.target
Wants=network-online.target

[Mount]
What=192.168.3.1:/
Where=/home/<user>/www_data
Type=nfs
Options=nfsvers=4,rw,users,soft
ForceUnmount=true
```

This file does not contain an [Install] section and will not be activated. The control is taken over by the now following file `home-<user>-www_data.automount` in the same directory.

```
[Unit]
Description=Automount server1/var/www/ using NFS
ConditionPathExists=/home/<user>/www_data
Requires=NetworkManager.service
After=network-online.target
Wants=network-online.target

[Automount]
Where=/home/<user>/www_data
TimeoutIdleSec=60

[Install]
WantedBy=remote-fs.target
```

```
WantedBy=multi-user.target
```

Afterwards:

```
# systemctl enable --now home-<user>-www_data.automount
```

Now the "document-root" directory of the Apache web server will be mounted as soon as we switch to the `/home/<user>/www_data` directory.
The status prompt confirms the action.

```
# systemctl status home-<user>-www_data.mount
home-<user>-www_data.mount Mount server1/var/www/ using NFS
     Loaded: loaded (/usr/local/lib/systemd/system/home-<user↙
        >-www_data.mount; disabled; vendor preset: enabled)
     Active: active (mounted) since Wed 2021-03-10 [...]
TriggeredBy: ● home-<user>-www_data.automount
      Where: /home/<user>/www_data
       What: 192.168.3.1:/
      Tasks: 0 (limit: 4279)
     Memory: 120.0K
        CPU: 5ms
      CGroup: /system.slice/home-<user>-www_data.mount
[...]


# systemctl status home-<user>-www_data.automount
home-<user>-www_data.automount Automount server1/var/www/ ↙
   usuing NFS
  Loaded: loaded (/usr/local/lib/systemd/system/home-<user>-↙
     www_data.automount; enabled; vendor preset: enabled)
  Active: active (running) since Wed 2021-03-10 [...]
Triggers: ● home-<user>-www_data.mount
   Where: /home/<user>/www_data
[...]
```

The journal excerpt vividly logs the operation of *"TimeoutIdleSec=60"* to unmount the file system and mount it again by starting the file manager Thunar and a call to `/home/<user>/www_data` in the terminal.

```
# journalctl -f -u home-<user>-www_data.*
[...]systemd[1]: Mounted Mount server1/var/www/ using NFS
[...]systemd[1]: Unmounting Mount server1/var/www/ using NFS
[...]systemd[1]: home-<user>-www_data.mount: Succeeded.
[...]systemd[1]: Unmounted Mount server1/var/www/ using NFS
[...]systemd[1]: home-<user>-www_data.automount: Got
                automount request for /home/<user>/www_data
                triggered by 2500 (Thunar)
[...]systemd[1]: Mounting Mount server1/var/www/ using NFS
[...]systemd[1]: Mounted Mount server1/var/www/ using NFS
[...]systemd[1]: Unmounting Mount server1/var/www/ using NFS
[...]systemd[1]: home-<user>-www_data.mount: Succeeded.
[...]systemd[1]: Unmounted Mount server1/var/www/ using NFS
[...]systemd[1]: home-<user>-www_data.automount: Got
                automount request for /home/<user>/www_data
                triggered by 6582 (bash)
[...]systemd[1]: Mounting Mount server1/var/www/ using NFS
[...]systemd[1]: Mounted Mount server1/var/www/ using NFS
[...]systemd[1]: Unmounting Mount server1/var/www/ using NFS
[...]systemd[1]: home-<user>-www_data.mount: Succeeded.
[...]systemd[1]: Unmounted Mount server1/var/www/ using NFS
```

More examples

Using your favorite search engine, you can find many examples on how to use mount and automount units on the Internet. The chapter "Sources" contains some websites with lots of further examples. We urgently recommend to also read the man pages.

7.13.4   Sources systemd-mount

```
man systemd.mount
man mount
```

Manjaro Forum, systemd.mount

Manjaro Forum, Use systemd to mount ANY device

Last edited: 2022/04/09

7.14   systemd-target - target unit

The basic and introductory information about Systemd can be found on the manual page systemd-start. The sections *[Unit]* and *[Install]* concerning all unit files are covered by our manual page systemd unit file.
Now the function of the **systemd.target** unit will be explained in more detail, which is similar to the commonly known runlevels.

The different runlevels that are booted or switched to are described by systemd as target units. They have the extension ".target".

The old sysvinit commands are still supported. (For this a quote from `man` ↙
`systemd`: "… is provided for compatibility reasons and because it is easier to type.")

target unit	description
emergency.target	launches into an emergency shell on the main console. It is the most minimal version of a system boot to get an interactive shell. This unit can be used to guide the boot process step by step.
rescue.target	starts the base system (including system mounts) and an emergency shell. Compared to multi-user.target, this target could be considered as single-user.target.
multi-user.target	starts a multi-user system with a working network, without graphics server X. This unit is used when you want to stop X or not to boot into it. This unit is used in special cases (when X itself or the desktop environment are upgraded) to perform a system upgrade (dist-upgrade).
graphical.target	is the unit for multi-user mode with network capability and a running X window system.
default.target	is the default unit that systemd launches at system startup. In siduction this is a symlink to graphical.target (except for the noX variant).

A look at the documentation `man SYSTEMD.SPECIAL(7)` is mandatory to understand the relationships of the different target units.

7.14.1   Special features

There are three special features to be considered for the target units:

1. The use on the kernel command line during the boot process
   In order to get into the edit mode in the boot manager Grub, you must press the **e** key when the boot selection appears. Then append the desired target to the kernel command line with the following syntax: "systemd.unit=xxxxx.target". The table lists the kernel commands and their still valid numeric equivalents.

target unit	kernel command	kernel command old
emergency.target	systemd.unit=emergency.target	-
rescue.target	systemd.unit=rescue.target	1
multi-user.target	systemd.unit=multi-user.target	3
graphical.target	systemd.unit=graphical.target	5

   The old runlevels 2 and 4 refer to multi-user.target

2. The use in the terminal during a running session
   Provided you are in a running graphical session, you can switch to the virtual terminal tty3 with the key combination **CTRL**+**ALT**+**F3**. Here you log in as **root** user. The following table lists the terminal commands, where the expression *"isolate"* ensures that all services not requested by the target unit are terminated.

target unit	terminal command	init command alt
emergency.target	systemctl isolate emergency.target	-
rescue.target	systemctl isolate rescue.target	init 1
multi-user.target	systemctl isolate multi-user.target	init 3
graphical.target	systemctl isolate graphical.target	init 5

3. Target units that should not be called directly
   A number of target units are used to group intermediate steps with dependencies during the boot process or the .target change. The following list shows

three frequently used commands that **should not** be called with the syntax "isolate xxxxx.target".

target	terminal command	init command alt
halt	systemctl halt	-
poweroff	systemctl poweroff	init 0
reboot	systemctl reboot	init 6

"halt", *"poweroff"*, and *"reboot"* fetch several units in the correct order to terminate the system in an orderly fashion and to reboot if necessary.

7.14.2 Sources systemd-target

```
man systemd.target
```

Last edited: 2022/04/09

7.15   systemd-path

The basic and introductory information about systemd can be found on the manual page systemd-start. The sections *[Unit]* and *[Install]* concerning all unit files are covered by our manual page systemd unit file.

On this manual page, we explain the function of the **systemd.path** unit, which systemd uses to monitor paths and trigger path-based actions.

The path unit makes it possible to trigger an action when files and directories (paths) are changed.

Once an event occurs, systemd can execute a command or script through a service unit. The path unit is not able to monitor directories recursively. However, multiple directories and files can be specified.

The path-specific options are configured in the [Path] section.

7.15.1   Required files

The **systemd-path** unit requires at least two files with preferably the same name but different extensions in the directory `/usr/local/lib/systemd/system/` for its function. (If necessary, create the directory beforehand with the command `mkdir -p /usr/local/lib/systemd/system/`.) These are

- the path unit file (<name>.path), which contains the monitoring and the trigger for the service unit
  and

- the service unit file (<name>.service), which contains the action to be started. For more extensive actions, you also create a script in `/usr/local/bin/↵` that is executed by the service unit.

7.15.2   Path unit options

The path unit must contain the [Path] section, which defines how and what to monitor.

The special options are:

- `PathExists=`

  checks if the path in question exists. If it does, the corresponding unit will be activated.

- `PathExistsGlob=`

  As above; supports file glob expressions (see also `man glob`).

- `PathChanged=`

  observes a file or path and activates the associated unit when changes occur. Action-triggering changes are:

  - creation and deletion of files

  - attributes, permissions, ownership

  - closing the file being watched after write access and closing any file after write access when the path is watched

- `PathModified=`

  As before, but in addition the associated unit is activated on simple write accesses, even if the file is not closed.

- `DirectoryNotEmpty=`

  activates the corresponding unit if the directory is not empty.

- `Unit=`

  activates the associated unit to be activated. It should also be noted that the path unit activates the service unit with the same name by default. Only in case of deviations from this, the `Unit=` option within the [Path] section is necessary.

- `MakeDirectory=`

  The directory to be watched will be created before watching.

- `DirectoryMode=`

  sets the access mode in octal notation when used for the previously created directory (default: 0755).

An example

Based on the Apache web server configuration according to our manual page LAMP - Apache, users and rights, let's illustrate the interaction of path unit with another systemd unit.

The figure *Path unit function* represents the dependencies of the systemd units of our example.
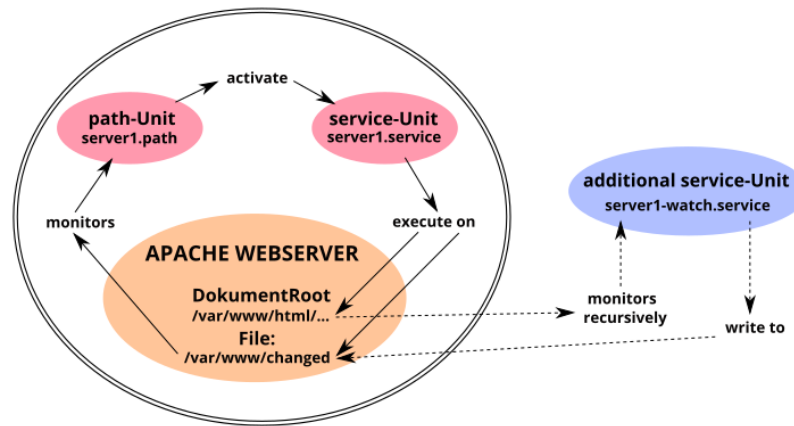
The double-bordered part in the graphic illustrates the path unit's core function. The server1.path unit monitors the file `/var/www/changed` and activates the corresponding server1.service unit in case of changes. This in turn then performs the desired actions in the directory `/var/www/html/` and restores the file `/var/www↙ /changed`.

The server1-watch.service unit outside the outline takes over the recursive monitoring of the Apache web server's `DocumentRoot`.

7.15.3   Create path unit

We create the file `server1.path` in the directory `/usr/local/lib/systemd/↙ system/`, which monitors the file `/var/www/changed` for changes, with the following content:

```
[Unit]
Description=Monitoring "changed" file!
BindsTo=server1-watch.service
After=server1-watch.service
```

```
[Path]
PathModified=/var/www/changed

[Install]
WantedBy=multi-user.target
```

Explanations

[Unit] section:

The *"BindsTo="* option represents the strongest available binding of two systemd units to each other. If one of them enters an error state during startup or operation, the other one will also be terminated immediately.

Together with the *"After="* option, it is achieved that the server1.path unit starts only after the server1-watch.service unit reports its successful start back to systemd.

[Path] section:

"PathModifid=" is the correct choice. The option reacts to changes in the file `/⤸ var/www/changed`, even if the file is not closed.

The *"PathModifid="* option (or others, see above) can be specified multiple times.

7.15.4   Service unit for path

The server1.service unit is activated and controlled by the server1.path unit and therefore does not need an [Install] section. Thus, the unit's description in the [Unit] section and the commands to be executed in the [Service] section are sufficient.

We create the file `server1.service` in the directory `/usr/local/lib/systemd⤸ /system/` with the following content.

```
[Unit]
Description=Change permissions in server1 folder

[Service]
Type=oneshot
ExecStartPre=/usr/bin/truncate -s 0 /var/www/changed
ExecStart=/usr/bin/chown -R www-data /var/www/html/
```

```
ExecStart=/usr/bin/chmod -R g+w /var/www/html/
ExecStart=/usr/bin/chmod -R o-r /var/www/html/
```

Explanations

[Service] section:

"ExecStart=" commands are executed only after all *"ExecStartPre="* commands have completed successfully. First the file `/var/www/changed` is reset to 0 byte and then the rest is executed.

Create additional service unit

Since the path unit cannot recursively monitor directories, we need an additional service unit for our example. We create the file `server1-watch.service` in the directory `/usr/local/lib/system/system/` with the following content.

```
[Unit]
Description=Watching server1 folder
Before=server1.path
Wants=server1.path

[Service]
Type=forking
ExecStart=inotifywait -dqr -e move,create -o /var/www/changed↵
    /var/www/html/

[Install]
WantedBy=multi-user.target
```

Remark:

Interestingly, systemd internally uses the inotify API for path unit to monitor filesystems, but does not implement its recursive function.

Explanations

[Unit] section:

"Before=" and *"Wants="* are the corresponding correlations to *"BindsTo="* and *"After="* from the server1.service unit.

[Service] section:

"inotifywait" logs to the `/var/www/changed` file located outside of the Apache web server's `DocumentRoot`.

7.15.5 Include path unit

Due to the dependency, we first incorporate the server1.path unit and then the server1-watch.service unit into systemd. The server1.service unit does neither need nor contain an [Install] section. When trying to include it, we receive an error message.

```
# systemctl enable server1.path
Created symlink /etc/system/system/multi-user.target.wants/↲
    server1.path /usr/local/lib/system/system/server1.path.

# systemctl enable server1-watch.service
Created symlink /etc/system/system/multi-user.target.wants/↲
    server1-watch.service /usr/local/lib/system/system/server1↲
    -watch.service.
```

Now the monitoring is also immediately active, as the status outputs of all three units show us.

```
# systemctl status server1-watch.service
server1-watch.service - Watching server1 folder.
  Loaded: loaded (/usr/local/lib/systemd/system/server1-watch↲
     .service; enabled; vendor preset: enabled)
  Active: active (running) since Sun 2021-02-21 [...]
 Process: 23788 ExecStart=inotifywait -dqr -e move,create
         -o /var/www/changed /var/www/html/ (code=exited
         status=0/SUCCESS)
Main PID: 23790 (inotifywait)
   Tasks: 1 (limit: 2322)
  Memory: 216.0K
     CPU: 5ms
   CGroup: /system.slice/server1-watch.service└─
```

```
           23790 inotifywait -dqr -e move,create
               -o /var/www/changed /var/www/html/


[...]systemd[1]: Starting Watching server1 folder....
[...]systemd[1]: Started Watching server1 folder..


# systemctl status server1.path
server1.path - Monitoring "changed" file!
  Loaded: loaded (/usr/local/lib/systemd/system/server1.path
         enabled; vendor preset: enabled)
  Active: active (waiting) since Sun 2021-02-21 [...]
Triggers: ● server1.service


Feb 21 19:25:20 lap1 systemd[1]: Started Monitoring "changed"↙
    file!.


# systemctl status server1.service
server1.service - Change permissions in server1 folder
     Loaded: loaded (/usr/local/lib/systemd/system/server1.↙
        service; static)
     Active: inactive (dead)
TriggeredBy: ● server1.path
```

The *"Active: inactive (dead)"* status of the last output is the normal state for
the server1.service unit, because this unit is only active if it was triggered by
server1.path to execute its command chain. After that, it returns to the inactive
state.

7.15.6 Execute service unit manually

Should it ever be helpful or necessary to manually change the file permissions in
`DocumentRoot` of the Apache web server, we simply issue this command:

```
# systemctl start server1.service
```

A new status query generates some additional log lines, from which we can see the successful completion of the command chain.

```
# systemctl status server1.service
server1.service - Change permissions in server1 folder
     Loaded: loaded (/usr/local/lib/systemd/system/server1.↵
        service; static)
     Active: inactive (dead) since Mon 2021-02-22 [...]
TriggeredBy: ● server1.path
  Process: 2822 ExecStartPre=truncate -s 0 /var/www/changed
           (code=exited, status=0/SUCCESS)
  Process: 2823 ExecStart=chown -R www-data /var/www/html1/
           (code=exited, status=0/SUCCESS)
  Process: 2824 ExecStart=chmod -R g+w /var/www/html1/
           (code=exited, status=0/SUCCESS)
  Process: 2825 ExecStart=chmod -R o-r /var/www/html1/
           (code=exited, status=0/SUCCESS)
 Main PID: 2825 (code=exited, status=0/SUCCESS)
      CPU: 19ms


[...]systemd[1]: Starting Change permissions in server1
[...]systemd[1]: server1.service: Succeeded.
[...]systemd[1]: Finished Change permissions in server1
```

7.15.7   Sources systemd-path

```
man systemd.path
```

Last edited: 2022/04/10

7.16   systemd-timer

The basic and introductory information about systemd can be found on the manual page systemd-start. The sections *[Unit]* and *[Install]* concerning all unit files are dealt with on our manual page systemd unit file.
On this manual page we explain the function of the unit **systemd.timer**, which can be used to trigger time-controlled actions.

The timer unit is mostly used to do regularly occurring actions. For this a service unit of the same name is necessary, in which the actions are defined. As soon as the system timer matches the time defined in the timer unit, the latter activates the service unit of the same name.
If configured accordingly, missed runs while the machine was off can be made up. It is also possible for a timer unit to trigger the desired actions only once at a previously defined time.

7.16.1   Required files

The **systemd-timer** unit needs two files with the same base name in the directory `/usr/local/lib/systemd/system/` for its function. (If necessary, create the directory beforehand with the command `mkdir -p /usr/local/lib/systemd↙ /system/`.) These are

- the timer unit file (xxxxx.timer), which contains the timing and trigger for the service unit
  and

- the service unit file (xxxxx.service), which contains the action to be started.

For more extensive actions, you can create a script in `/usr/local/bin/` as a third file that is executed by the service unit.

In the example we create a regular backup with `rsync`.

7.16.2   Service unit for timer

The service unit that executes the backup is activated and controlled by the timer unit and therefore does not need an [Install] section. Thus the unit's description in the [Unit] section is sufficient. Your [Service] section contains the command to be executed after the option `ExecStart=`.

We create the file `backup.service` in the directory `/usr/local/lib/systemd↲` `/system/` with the following content:

```
[Unit]
Description="Command to backup my home directory"

[Service]
Type=oneshot
ExecStart=/usr/bin/rsync -a --exclude=.cache/* /home/<user> /↲
    mnt/sdb5/backup/home/
```

Please replace the <user> string with your own user.

7.16.3   Create timer unit

We create the file `backup.timer` in the directory `/usr/local/lib/system/↲` `system/` with the following content:

```
[Unit]
Description="Backup my home directory"

[Timer]
OnCalendar=*-*-* 19:00:00
Persistent=true

[Install]
WantedBy=timers.target
```

Explanations

The timer unit must contain the [Timer] section, which defines when and how the

corresponding service unit is triggered.

There are two available timer types:

1. Realtime timers,

   which define a realtime (i.e. wall clock) timer with the `OnCalendar=` option.
   (the example *"OnCalendar=*-*-* 19:00:00"* means "daily at 19:00"),

   and

2. Monotonic timers,

   which define a timer relative to one of the options `OnActiveSec=`↙
   , `OnBootSec=`, `OnStartupSec=`, `OnUnitActiveSec=`, `OnUnitInactiveSec`↙
   `=`.
   "OnBootSec=90" means "90 seconds after bootup" and
   "OnUnitActiveSec=1d" means "One day after the timer was last activated".
   Both options together trigger the associated service unit 90 seconds after
   boot and then exactly every 24 hours as long as the machine is not shut
   down.

The `Persistent=` option included in the example saves the time when the ser-
vice unit was last triggered as an empty file in the `/var/lib/systemd/timers/`
directory. This is useful for catching up on missed runs when the machine was off.

Include timer unit

We include the timer unit into systemd with the following command:

```
# systemctl enable backup.timer
Created symlink
 /etc/systemd/system/timers.target.wants/backup.timer →
   /usr/local/lib/systemd/system/backup.timer.
```

The analogous command for the service unit is not necessary and would also lead
to an error, since there is no [Install] section in it.

Trigger timer unit manually.

Not the timer unit, but the service unit to be triggered by it is called.

```
# systemctl start backup.service
```

7.16.4 Timer unit as cron replacement

"cron" and "anacron" are the best known and widely used job timers. systemd timers can be an alternative. We briefly look at the benefits of and caveats to systemd timers.

Benefits

- Jobs can have dependencies (depend on other systemd services).
- Timer units are logged in the systemd journal.
- You can easily call a job independently from its timer.
- You can give timer units a nice value or use cgroups for resource management.
- systemd timer units can be triggered by events like booting or hardware changes.
- They can be easily enabled or disabled with systemctl.

Caveats

- Configuring a cron job is a simple process.
- cron can send emails using the MAILTO variables.

7.16.5 Sources systemd-timer

```
man systemd.timer
```

Archlinux Wiki, Timers
PRO-LINUX.DE, Systemd Timer Units…

Last edited: 2022/04/10

7.17   systemd-boot

Although it was included in systemd more than ten years ago, the boot manager systemd-boot (sd-boot for short) is rarely found on desktop systems. If you decide to use sd-boot with a suitable setup and after thorough testing, the changeover is not a major challenge for a somewhat experienced user.

Features

The boot manager sd-boot was developed with the aim of making the boot process fast, simple and secure. The user receives a text-based boot menu. The display does not need any bells and whistles.

In keeping with its name, it is deeply integrated into systemd and accesses the services available there. sd-boot creates a visually minimalist boot menu. Each menu entry is based on a single, separate text file. The configuration for the user interface is rudimentary compared to GRUB and the installation scope, with 32 files and a twentieth of the data volume, is very small.

sd-boot is only available for UEFI hardware. For this, sd-boot requires an ESP (**E**fi **S**ystem **P**partition). An XBOOTLDR partition is also recommended.

If only the ESP is present, it is mounted under `/boot`.

For ESP and XBOOTLDR partitions, the ESP is mounted under `/efi` and the XBOOTLDR partition under `/boot`.

File system drivers for the XBOOTLDR partition may have to be stored under `/↵ efi`.

sd-boot can only boot operating systems from partitions of the same medium. To boot operating systems from other media, use the ChainLoader method to GRUB or another boot loader.

siduction installs the boot manager GRUB automatically. It is not possible to select sd-boot during the installation. If you switch to sd-boot later, sd-boot creates customized initrd for each kernel present in the system. Only with these initrd will sd-boot boot.

systemd-boot functions:

- Boot from fully encrypted hard disk.

- Support for the XBOOTLDR partition.

- Loading of drop-in drivers.

- Registering SecureBoot keys.

- Create a menu entry when installing new kernels.

- Boot counting
  In connection with failed boot processes, the boot entry can be removed automatically.

- Support for passing a random seed to the OS.
  This serves to protect against the use of manipulated OS images.

It is also possible to pass commands to sd-boot before a reboot or during the boot process. Information on this can be found in `man systemd-boot`.

Application scenarios

Suitability with different system configurations and in comparison with GRUB.

System configuration	sd-boot	GRUB	Remark
An immutable OS	++	o	Only a minimalist boot manager without a menu is required. GRUB is oversized for this purpose.
One OS on one HD	++	o	Boot menu only necessary with multiple kernels.
Several Linux OS on one HD	+	+	sd-boot necessary on all OS. GRUB requires external module *os-prober*.
Dual boot with WIN / MAC on one HD	+	+	As before. Both can be booted using ChainLoader.

System configuration	sd-boot	GRUB	Remark
Several Linux OS on several HD	-	+	sd-boot can only boot OS from one HD. Two instances in UEFI with selection via firmware necessary.
Dualboot with WIN / MAC on several HD	-	+	As before.
Several variants of a Linux OS on one HD	o	o	With sd-boot manual adjustment of the menu entries necessary. For GRUB, create or modify the file `/etc/default/grub.d/xxxx.cfg`.
Linux OS on Btrfs file system with support for snapper	-	o	sd-boot creates menu entries only once when installing the kernel, regardless of the subvolume. Other subvolumes do not receive an entry. GRUB relies on different additional software depending on the distribution.
A fully encrypted HD	++	+	sd-boot passes the tasks to the kernel and the user space and is therefore more efficient. GRUB requires additional software.

sd-boot fully demonstrates its advantages with a simple hardware setup, but fails with operating systems on multiple media.

Grub, on the other hand, can be used more universally, is therefore heavyweight and still requires external software. GRUB is oversized for simple hardware setups.

7.17.1   Installing systemd-boot

> **Attention**
>
> It is essential to make a data backup on an external medium.
> Work on partitions is necessary to install sd-boot according to the recommen-
> dations.

The instructions are based on the standard installation of *siduction* with an ESP
(**E**fi **S**ystem **P**partition), which is mounted under `/boot/efi`. This is also the stan-
dard for Debian and many Debian derivatives. To switch to sd-boot, only the two
packages `systemd-boot` and `systemd-boot-efi` are required.
But be careful, first some work on the system is necessary.

7.17.2   System preparation

Partitioning

A significant change of sd-boot compared to the standard installation of siduction
with GRUB is the relocation of the directory `/boot` into one or better two separate
partitions. The recommendations of sd-boot are:

- **Both ESP and XBOOTLDR:**
  ESP
  gdisk partition type "EF00", Part-GUID code: C12A7328-F81F-11D2-BA4B-
  00A0C93EC93
  File system: VFAT, mounted under /efi/
  Size: 100 MB
  XBOOTLDR
  gdisk Partition type "EA00", Part-GUID code: BC13C2FF-59E6-4262-A352-
  B275FD6F7172
  File system: Any file system that the UEFI implementation can read, mounted
  under /boot/
  Size: at least 1 GB

- **Only ESP:**
  (Conditionally recommended, see below.)

gdisk partition typ "EF00", Part-GUID code: C12A7328-F81F-11D2-BA4B-00A0C93EC93

File system: VFAT, mounted under /boot/

Size: at least 1 GB

- **Not recommended:**
  ESP mounted under /boot/efi/

Quotes from the boot_loader_specification.

[Quote start]

Note: These partitions are shared among all OS installations on the same disk. Instead of maintaining one boot partition per installed OS (as /boot/ was traditionally handled), all installed OSes use the same place for boot loader menu entries.

[…]

Mounting the ESP to /boot/efi/, as was traditionally done, is not recommended. Such a nested setup complicates an implementation via direct autofs mounts — as implemented by systemd for example —, as establishing the inner autofs will trigger the outer one. Mounting the two partitions via autofs is recommended because the simple VFAT file system has weak data integrity properties and should remain unmounted whenever possible.

[…]

For systems where the firmware is able to read file systems directly, the ESP must — and the MBR boot and GPT XBOOTLDR partition should — be a file system readable by the firmware. For most systems this means VFAT (16 or 32 bit). Applications accessing both partitions should hence not assume that fancier file system features such as symlinks, hardlinks, access control or case sensitivity are supported.

[Quote end]

Minimum size

If you follow the recommendations of sd-boot and use both the ESP and the XBOOTLDR partition, you can keep the small size of the ESP and use a more powerful file system for the XBOOTLDR partition. Necessary file system drivers for the XBOOTLDR partition are to be fetched from akeo.ie and copied to `/efi⤸`

`/EFI/systemd/drivers/`.

The XBOOTLDR partition should be at least 1 GB, as sd-boot stores all kernels and initrd in it twice. As prescribed by the standard, once directly under `/boot/` and once again under `/boot/<entry-token-or-machine-id>/<version>/`. This results in 70 to 100 MB for a kernel with initrd. The reason for this is probably the use of VFAT, which does not support symlinks. With several OSes, each with several kernels, the size of 1 GB is borderline and the usual 200 to 300 MB of the ESP are completely unsuitable. This requires a change in partitioning.

We keep the ESP in its previous size and create the XBOOTLDR partition (gdisk type EA00) anywhere on the same medium. As already mentioned, at least 1 GB in size, preferably 2 GB. If the UUID of an existing partition changes, the `/etc/⤸ fstab` file must be adjusted. See: Customize the fstab.

Copy data and adapt files

We open a terminal and become ROOT with `su`.
If `gdisk` is not installed, we do this and read the *Partition GUID code* of the ESP and XBOOTLDR partition. The device file can of course also be '/dev/sda' and the number after the `-i` option names the partition on the medium.

```
# sgdisk -i1 /dev/nvme0n1
Partition GUID code: C12A7328-F81F-11D2-BA4B-00A0C93EC93B (⤸
   EFI system partition)
[...]
# sgdisk -i4 /dev/nvme0n1
Partition GUID code: BC13C2FF-59E6-4262-A352-B275FD6F7172 (⤸
   XBOOTLDR partition)
[...]
```

The first line of the output of `sgdisk` must look as shown. If not, use `gdisk` to change the partition type to *EF00* (ESP) and *EA00* (XBOOTLDR).

In the next step we create new directories and copy the directory `/boot`.

```
# cd /
# mkdir /efi
# mkdir /grub
```

```
# cp -a /boot/* /grub/
# umount /boot/efi/
# rm -r /boot/*
```

We also backed up the ESP data with the command sequence.
Now we adapt the file `/etc/fstab`. (Create a backup copy beforehand.)

```
# cp /etc/fstab /etc/fstab_$(date +%F)
```

We create a copy of the line with the mount point `/boot/efi` and only remove the string "/boot" from the original line.
In the copy we write the data for the XBOOTLDR partition with the mount to `/boot`. Finally, the changes should look like this.

```
[...]
UUID=FA62-156D            /efi    vfat   defaults 0 2
UUID=<uuid_of_xbootldr> /boot   ext4   defaults 0 2
[...]
```

Now we mount the partitions and copy the kernel with accessories to `/boot`. Then we create the directory for the file system drivers of the XBOOTLDR partition.

```
# systemctl daemon-reload
# mount /boot/
# mount /efi/
# cp -a /grub/vmlinuz-6.8.9-1-siduction-amd64 /boot/
# cp -a /grub/initrd.img-6.8.9-1-siduction-amd64 /boot/
# cp -a /grub/System.map-6.8.9-1-siduction-amd64 /boot/
# cp -a /grub/config-6.8.9-1-siduction-amd64 /boot/
   # or more simply: cp -a /grub/*-6.8.9-1-siduction-amd64 /↲
      boot/
# mkdir -p /efi/EFI/systemd/drivers/
```

Finally, we fetch the file system drivers from the website akeo.ie and save them in the directory created under `/efi/`. Pay attention to execution rights.

Installation

Once the preparations have been completed and the result has been checked, e.g. with the commands `ls -l /boot lsblk` or `sgdisk -i...`, a command line is sufficient to add the two required packages to our system.

```
# apt install systemd-boot-efi systemd-boot
[...]
Copied "/usr/lib/systemd/boot/efi/systemd-bootx64.efi" to "/↵
   efi/EFI/systemd/systemd-bootx64.efi".
Copied "/usr/lib/systemd/boot/efi/systemd-bootx64.efi" to "/↵
   efi/EFI/BOOT/BOOTX64.EFI".
Created "/boot/e5cc6ff820c1450c93a29d8723c78cd1".
! Mount point '/efi' which backs the random seed file is ↵
   world accessible, which is a security hole!
! Random seed file '/efi/loader/random-seed' is world ↵
   accessible, which is a security hole!
Random seed file /efi/loader/random-seed successfully ↵
   installed (32 bytes).
Created EFI boot entry "Linux Boot Manager".
```

In the firmware UEFI menu, we now find the *Linux Boot Manager* in first place, which, as the output shows, has been written to the ESP. At the same time, sd-boot has created the boot entries for the *Linux Boot Manager* in the XBOOTLDR partition.
For each additional OS on the medium, the 'System preparation' and the 'Installation' are necessary.

7.17.3   Configuration

There are only two places where the configuration for sd-boot is located.
For the *Linux Boot Manager* it is the file `/efi/loader/loader.conf` and for the *menu entries* it is the files with the extension `.conf` in the directory `/boot/↵ loader/entries/`.

Linux Boot Manager

The configuration of the *Linux Boot Manager* only affects the following points
- Display of the menu and timeout - no/yes/duration
- Console mode - value
- Editing the kernel line - yes/no
- Set default boot entry - value

For only one system with one kernel, the configuration file has the following content:

```
#timeout 3
#console-mode keep
```

It boots directly into the operating system without displaying a menu, because 'time-out' is commented out. The reason: Only one menu item exists so far. After removing the comment character, the menu of a siduction XFCE installation appears.

```
                  siduction XFCE
          Reboot Into Firmware Interface
```

If there is more than one OS on the medium, the file `/efi/loader/loader.conf` contains an additional line `default <entry-token-or-machine-id>-*` to define the default entry.

```
timeout 3
#console-mode keep
default e5cc6ff820c1450c93a29d8723c78cd1-*
```

Menu entries

The configuration files are located in the directory `/boot/loader/entries/⤶` and their name corresponds to the format `<entry-token-or-machine-id>-<⤶ kernelversion>.conf`.
sd-boot automatically adds the entries to the menu when additional kernels are installed. With an additional kernel, the menu items also receive the kernel version.

```
          siduction XFCE (6.8.10-1-siduction-amd64)
          siduction XFCE (6.8.9-1-siduction-amd64)
```

```
                    Reboot Into Firmware Interface
```

Of **several OS** on different partitions, only the one from which sd-boot was installed is automatically listed in the menu. The kernels available there appear in the selection. If you install sd-boot in every OS, these boot entries also find their way into the menu. Here, for example, is UBUNTU.

```
            siduction XFCE (6.8.10-1-siduction-amd64)
            siduction XFCE (6.8.9-1-siduction-amd64)
                      Ubuntu 24.04 LTS
                Reboot Into Firmware Interface
```

If, for whatever reason, a menu entry is missing, the generation of the entry can be restarted at any time. It must be ensured that the kernel, the vmlinuz and the config file of the corresponding version are located in the `/boot/` directory. From the corresponding OS, the command

```
 dpkg-reconfigure linux-image-6.8.9-1-siduction-amd64
```

also creates the desired menu entry.

7.17.4   Remove GRUB

sd-boot cannot currently (08-2024) be selected as the default boot manager when installing siduction. If extensive tests with sd-boot were successful, we have to remove GRUB from our system to avoid errors during update and upgrade.

Packages

We open a terminal, become ROOT with `su` and use the command `apt purge` to remove the configuration files as well.

```
01|# apt purge grub*
02| Package lists are read...
03| Dependency tree is built...
04| Status information is read in...
05| The following packages are REMOVED:
```

```
06| grub-pc-bin* os-prober* grub-btrfs* siduction-btrfs* grub↵
   -efi-amd64-bin*
07| grub2-common* grub-common* grub-efi-ia32-bin* memtest86+*↵
    grub-pc*
08| 0 updated, 0 reinstalled, 10 to be removed.
09| After this operation 0 B disk space will be used ↵
   additionally.
10| Do you want to continue? [Y/n] y
11| (Read database ... 249262 files and directories are ↵
   currently installed).
12| Delete the configuration files of grub-btrfs (4.11-0~1↵
   siduction3) ...
13| /var/lib/dpkg/info/grub-btrfs.postrm: 23: update-grub: ↵
   not found
14| dpkg: Error while removing the package grub-btrfs (--↵
   purge):
15| "installed post-removal script of grub-btrfs "package ↵
   subprocess returned error value 127
16| An errors occurred while removing:
17|  grub-btrfs
```

The *grub-btrfs* package throws an error. In line 13 we read that the post-removal
script can no longer find the command from line 23 (update-grub), which is of
course correct, because *grub-pc-bin* has already been removed.
Consequently, we comment out the line and try apt again.

```
# sed -i '23s!\(.*\)!#\1!' /var/lib/dpkg/info/grub-btrfs.↵
   postrm
# apt purge grub-btrfs
```

After this action, the backup copies we created and two GRUB directories remain.
These are
the directory `/grub/`,
the backup file `/etc/fstab_*`,

the directory `/usr/share/grub/`

and the directory `/etc/default/grub.d/`.

Let's get rid of the leftovers with

```
# rm -r /grub/
# rm /etc/fstab_*
# rm -r /usr/share/grub/
# rm -r /etc/default/grub.d/
```

Clean up UEFI

The siduction boot entry from the installation with GRUB still exists in the firmware.
We display the content in the terminal with ROOT rights. (Shortened version)

```
# efibootmgr
 BootCurrent: 0001
 Timeout: 0 seconds
 BootOrder: 0001,0000,0002,2001,2002
 Boot0000* siduction
 Boot0001* Linux Boot Manager
 Boot0002* EFI Hard Drive
 Boot2001* EFI USB Device
 Boot2002  EFI Network RC
```

The first command removes the GRUB entry.
The second command then deletes the corresponding directory from `/efi/`.

```
# efibootmgr -b 0000 -B
# rm -r /efi/EFI/siduction/
```

7.17.5   systemd-boot and Btrfs

The Btrfs file system, especially in conjunction with snapper, offers the possibility
of restoring a defective system to a previous state. In this context, the relocation
of the `/boot` directory to a separate partition required by sd-boot is a hindrance.
The `/boot` directory is an essential part of the operating system and should not

be moved to a partition or subvolume in connection with Btrfs. This is because it is not covered by snapshots of the root file system.

With the *siduction-btrfs* package, siduction is able to create menu entries for all kernels contained in the rollback target during a *rollback*. Menu entries for other snapshots are not available. The user can therefore decide for himself whether he wants to use sd-boot when installing the system on Btrfs.

7.17.6   Further information

`man systemd-boot`
boot_loader_specification
File system driver by akeo.ie

Last edited: 2024/09/21

7.18   System journal

The system journal consists of the *systemd-journald*, or **journald** for short, which collects and stores log messages, and the **journalctl**, which is used to manage, query, and output the collected log messages.

7.18.1   journald

journald is a system service that collects and stores log messages using the *systemd-journald.service* unit (and its associated socket units).
It creates and maintains structured, indexed journals based on log messages from:

- kernel log messages
- simple system log messages
- structured system log messages via the native journal API
- standard output and standard error output from service units
- audit records coming from the kernel audit subsystem

journald allows journal "namespaces". They are both a mechanism for logically isolating a log stream from the rest of the system, and a mechanism for improving performance. Journal namespaces exist concurrently and side-by-side. Each has its own independent log data stream. After siduction has been installed, only the system's default namespace exists.

By default, journald stores the log data permanently in
`/var/log/journal/MACHINE_ID`.

Log data for other namespaces can be found in
`/var/log/journal/MACHINE_ID.NAMESPACE`.

The `systemd-cat` command provides two ways to pass data from a process to the journal independently of systemd units.

1. `systemd-cat <program> <option(s)>`
   Used with a program call or command, systemd-cat redirects all standard input, standard output, and standard error output from a process to the journal.

2. **Used in a pipe,**

   systemd-cat serves as a filtering tool to send the previously created output to the journal.

If no parameter is passed, systemd-cat will send whatever it reads from standard input to the journal. The man-page `man systemd-cat` provides more information.

7.18.2   journald over the network

The systemd-journal modules *upload*, *remote*, and *gatewayd* allow system log data to be sent and received between different computers over the network. With their help remote computers can be monitored continuously. In this installation it makes sense to set up namespaces on the remote computer for the log data of the other computers.

For more information please read the man-pages journal upload, journal remote, and journal gatewayd.

7.18.3   journald.conf

The following files configure various parameters of the systemd journal service.

- /etc/systemd/journald.conf
- /etc/systemd/journald.conf.d/*.conf
- /etc/systemd/journald@NAMESPACE.conf (optional)
- /run/systemd/journald.conf.d/*.conf (optional)
- /usr/lib/systemd/journald.conf.d/*.conf (optional)

The default namespace managed by the systemd-journald.service (and its associated socket units) is configured in `/etc/systemd/journald.conf` and associated additions.

The configuration file contains the defaults as a commented out note to the administrator. To change settings locally, simply edit this file.

Instances that manage other namespaces are only needed if there is a need to deviate from the defaults. Their configuration file is to be created according to the pattern `etc/systemd/journald@NAMESPACE.conf`.

Service units can be assigned to a particular journal namespace using the unit file setting `LogNamespace=`.

By default, only the default namespace collects kernel and audit log messages.

Rank order

If packages need to customize configuration, they can install configuration snippets in `/usr/lib/systemd/*.conf.d/` or `/usr/local/lib/systemd/*.conf.d/`.

The main configuration file is read before any other from the configuration directories and has the lowest priority. Entries in a file in any of the configuration directories override entries in the main configuration file. Files in the `*.conf.d/` subdirectories are sorted by their file name, regardless of which subdirectory they are located in. If separate configuration files are necessary, it is recommended that all file names in these subdirectories be preceded by a two-digit number and a hyphen to simplify file sorting.

7.18.4 journalctl

journalctl is used to query the journal created by systemd-journald.
When called without parameters, the entire contents from all accessible sources of the journal are displayed, starting with the oldest entry.
The output is directed page by page by *less*. Long lines can be viewed using the **arrow-left** and **arrow-right** keys. The `--no-pager` option disables page-by-page viewing, shortening the lines to the width of the terminal.

journalctl offers, in addition to the options described below, a whole range of other options for filtering and formatting the output. Please also read the man page `man↲ journalctl`.

Rights

The user **root** and all users who are members of the groups **systemd-journal**, **adm**, and **wheel** are granted access to the system journal and the other users' journals. siduction adds all configured users to the **systemd-journal** group.

The journal contains trusted fields, i.e. fields that are implicitly added by the journal and cannot be changed by client code. They start with an underscore (e.g.: _PID=, _UID=, _GID=, _COMM=, _EXE=, _CMDLINE=).

Filter output.

- options: `--user`, `--system`, `--directory=`, `--file=`, `--namespace=`
  These options limit the source of the output to the named scope, directory, or file.

- options: `-b`, `-k`, `-u`, `-p`, `-g`, `-S`, `-U`
  The outputs of these options use all available journal files, unless one of the previously mentioned options is used in addition.

  - `-b` (`--boot=`)
    shows messages from a particular system boot. Without any argument, the logs for the current system startup are displayed. The argument "-1" prints the messages of the system startup before the current one. The argument "5" presents the messages of the fifth system start since the beginning of the records.

  - `-k` (`--dmesg`)
    displays only kernel messages. This includes the `-b` option so that only kernel messages since the current system start are printed.

  - `-u` (`--unit=`)
    This option requires the specification of a UNIT or a PATTERN.
    Prints the journal entries for the specified systemd unit UNIT or for all units that match the PATTERN.

  - `-p` (`--priority=`)
    filters the output by message priorities or priority ranges. Requires specification of a single protocol level or a range of protocol levels in the form FROM…TO.
    The log levels are the normal syslog log levels:
    "emerg" (0), "alert" (1), "crit" (2), "err" (3), "warning" (4), "notice" (5), "info" (6), "debug" (7).

Both the names and the digits of the protocol levels can be used as arguments. If a single protocol level is specified, all messages with this or a lower protocol level will be displayed.

- `-g` (`--grep=`)

  Requires the specification of a PERL-compatible regular expression to filter the output. The regular expression is applied to the "MESSAGE=" field in the journal entries.

- `-S` (`--since=`) and `-U` (`--until=`)

  The display will start with newer entries from the specified date or older entries up to the specified date. The date format should be "2012-10-30 18:17:16", but parts of it can be omitted. Alternatively, the strings "yesterday", "today", "tomorrow" are possible. The argument "now" refers to the current time. The specification of relative times allow a preceding "-" or "+", which refer to times before or after the specified time.

Control output

- options: `-f`, `-n`, `-r`, `-o`, `-x`, `--no-pager`

  - `-f` (`--follow`)

    displays only the newest journal entries and continuously outputs new entries. This includes the `-n` option. The output is similar to the old known command **tail -f /var/log/messages**.

  - `-n` (`--lines=`)

    shows the latest journal entries and limits the number of events to show. The argument is a positive integer. The default value is 10 if no argument is given.

  - `-r` (`--reverse`)

    The output starts with the newest entry.

  - `-o` (`--output=`)

    controls the formatting of the displayed journal entries. A number of other options are subordinate to this option, of which we will only consider the "short-full" option here.

`-o short-full`

The output is mostly identical to the formatting of classic syslog files. It displays one line per journal entry, but the timestamp is output in the format that the –since= and –until= options accept. Therefore, this output is very suitable to create a time-based filtering of journal entries in the following.

– `-x` (`--catalog`)

adds explanatory help text to journal lines where available.

– `--no-pager`

This option disables page-by-page display, shortening the lines to the width of the terminal. Using it is only useful if merely a small number of lines is expected for the output.

Control journalctl

The following options handle the management of data written by journald.

• `--disk-usage`

displays the current disk space usage of all journal files.

• `--vacuum-size=`, `--vacuum-time=`, `--vacuum-files=`

removes the oldest archived journal files until the disk space they use falls below the specified size, or all archived journal files that do not contain data older than the specified time period, or so that no more than the specified number of separate journal files remain. Executing `--vacuum-xxx` does not include the active journal files.

• `--rotate`

asks the journal daemon to rotate the journal files. Journal files rotation has the effect of marking all currently active journal files as archived and renaming them so that in the future they will never be written to again. Then new (empty) journal files will be created instead. This action can be combined with `--`⤸ `vacuum-xxx` in a single command to actually achieve the arguments given to `--vacuum-xxx`.

- `--verify`

  checks the journal files for internal consistency.

7.18.5   Mastering journalctl

As described above under permissions, you can use the journal as a simple user. Here are some examples:

command	display
journalctl	the full journal of all users, oldest entries first
journalctl -r	as before, newest entries first
journalctl -b	the log of the last boot
journalctl -b -1 -k	all kernel messages from the next to last boot (-1)
journalctl -b -p err	limited to the last boot and the priority ERROR
journalctl –since=yesterday	the journal since yesterday
journalctl /dev/sda	the journal of the device file /dev/sda
journalctl /usr/bin/dbus-daemon	all logs of the D-Bus daemon
journalctl -f	live view of the journal (formerly: tail -f /var/log/messages)

The option `--list-boots` prints the corresponding list.

```
# journalctl --list-boots --no-pager
[...]
 -50 8fc07f387... Sun 2021-02-28 11:07:05 CET-Sun [...] CET
 -49 aa49cb3af... Mon 2021-03-01 17:49:58 CET-Mon [...] CET
 -48 3a6e55a4a... Tue 2021-03-02 12:18:46 CET-Tue [...] CET
 -47 a46150a19... Wed 2021-03-03 11:06:29 CET-Wed [...] CET
 -46 d42ed8b05... Thu 2021-03-04 10:59:56 CET-Thu [...] CET
 -45 566f65991... Thu 2021-03-04 19:53:52 CET-Thu [...] CET
 -44 8e2da4a61... Fri 2021-03-05 10:15:18 CET-Fri [...] CET
[...]
```

Afterwards you can use the command `journalctl -b -47` to display the messages of the boot process of 3.3.2021.

Another new feature in logging is the tab completion for journalctl. If you type *journalctl* and press the **TAB** key twice, a list of possible completions appears:

```
$ journalctl
_AUDIT_FIELD_APPARMOR=          _KERNEL_SUBSYSTEM=
_AUDIT_FIELD_CAPABILITY=        KERNEL_USEC=
_AUDIT_FIELD_CAPNAME=           LEADER=
_AUDIT_FIELD_DENIED_MASK=       LIMIT=
_AUDIT_FIELD_INFO=              LIMIT_PRETTY=
_AUDIT_FIELD_NAME=              _LINE_BREAK=
_AUDIT_FIELD_OPERATION=         _MACHINE_ID=
_AUDIT_FIELD_OUID=              MAX_USE=
_AUDIT_FIELD_PEER=              MAX_USE_PRETTY=
_AUDIT_FIELD_PROFILE=           MESSAGE=
_AUDIT_FIELD_REQUESTED_MASK=    MESSAGE_ID=
_AUDIT_FIELD_SIGNAL=            NM_CONNECTION=
_AUDIT_ID=                      NM_DEVICE=
_AUDIT_LOGINUID=                NM_LOG_DOMAINS=
_AUDIT_SESSION=                 NM_LOG_LEVEL=
_AUDIT_TYPE=                    N_RESTARTS=
_AUDIT_TYPE_NAME=               _PID=
AVAILABLE=                      PRIORITY=
AVAILABLE_PRETTY=               SEAT_ID=
_BOOT_ID=                       _SELINUX_CONTEXT=
_CAP_EFFECTIVE=                 SESSION_ID=
_CMDLINE=                       SHUTDOWN=
CODE_FILE=                      SLEEP=
CODE_FUNC=                      _SOURCE_MONOTONIC_TIMESTAMP=
CODE_LINE=                      _SOURCE_REALTIME_TIMESTAMP=
_COMM=                          _STREAM_ID=
COMMAND=                        SYSLOG_FACILITY=
CONFIG_FILE=                    SYSLOG_IDENTIFIER=
CONFIG_LINE=                    SYSLOG_PID=
```

```
CURRENT_USE=                    SYSLOG_RAW=
CURRENT_USE_PRETTY=             SYSLOG_TIMESTAMP=
DISK_AVAILABLE=                 _SYSTEMD_CGROUP=
DISK_AVAILABLE_PRETTY=          _SYSTEMD_INVOCATION_ID=
DISK_KEEP_FREE=                 _SYSTEMD_OWNER_UID=
DISK_KEEP_FREE_PRETTY=          _SYSTEMD_SESSION=
ERRNO=                          _SYSTEMD_SLICE=
_EXE=                           _SYSTEMD_UNIT=
EXECUTABLE=                     _SYSTEMD_USER_SLICE=
EXIT_CODE=                      _SYSTEMD_USER_UNIT=
EXIT_STATUS=                    THREAD_ID=
_FSUID=                         TIMESTAMP_BOOTTIME=
_GID=                           TIMESTAMP_MONOTONIC=
GLIB_DOMAIN=                    _TRANSPORT=
GLIB_OLD_LOG_API=               _UDEV_DEVNODE=
_HOSTNAME=                      _UDEV_SYSNAME=
INVOCATION_ID=                  _UID=
JOB_ID=                         UNIT=
JOB_RESULT=                     UNIT_RESULT=
JOB_TYPE=                       USER_ID=
JOURNAL_NAME=                   USER_INVOCATION_ID=
JOURNAL_PATH=                   USERSPACE_USEC=
_KERNEL_DEVICE=                 USER_UNIT=
```

Most of these are self-explanatory. For example COMM, which stands for *command*, provides a lot of options:

journalctl _COMM= lists the possible applications after another hit of **TAB**:

```
$ journalctl _COMM=
acpid           hddtemp      ntpdate       systemd
acpi-fakekey    hdparm       ntpd          systemd-fsck
acpi-support    hp           ofono         systemd-hostnam
alsactl         hpfax        ofonod        systemd-journal
anacron         ifup         pkexec        systemd-logind
apache2         irqbalance   polkitd       systemd-modules
```

```
backlighthelper kbd           pulseaudio     systemd-shutdow
bash            kdm           pywwetha       systemd-udevd
bluetoothd      keyboard-setup pywwetha.py   teamviewerd
chfn            loadcpufreq   resolvconf     udev-configure-
chrome          logger        rpcbind        udisksd
console-kit-dae login         rpc.statd      udisks-daemon
console-setup   lvm           samba-ad-dc    umount
cpufrequtils    lvm2          saned          uptimed
cron            mbd           sensors        useradd
cups            mbmon         sh             usermod
dbus-daemon     mdadm         smartmontools  vboxdrv
ddclient        mdadm-raid    smbd           VBoxExtPackHelp
docvert-convert mtp-probe     ssh            vdr
glances         mysql         sshd           winbind
gpasswd         networking    su
gpm             nfs-common    sudo
groupadd        ntp           sysstat
```

With **journalctl _COMM=su** you can now see which user got root privileges with
su and when.

```
# journalctl _COMM=su
-- boot 1b5d2b3fcd9043d88d8abce665b75ed4 --
Mar 10 16:27:22 pc1 su[105197]: (to root) siduser on pts/1
Mar 10 16:27:22 pc1 su[105197]: pam_unix(su:session):
     session opened for user root(uid=0) by (uid=1000)
Mar 10 17:54:33 pc1 su[105197]: pam_unix(su:session):
     session closed for user root

-- boot 37b19f6321814620be1ed4deb3be467f --
Mar 10 17:56:35 pc1 su[3381]: (to root) siduser on pts/1
Mar 10 17:56:35 pc1 su[3381]: pam_unix(su:session):
     session opened for user root(uid=0) by (uid=1000)
Mar 10 19:07:17 pc1 su[3381]: pam_unix(su:session):
     session closed for user root
```

Another example:

You can additionally narrow the output by time.

```
# journalctl _COMM=dbus-daemon --since=2020-04-06 --until↙
   ="2020-04-07 23:40:00"
[...]
Apr 07 22:59:04 pc1 org.gtk.Private.GPhoto2VolumeMonitor↙
   [2006]: ### debug: in handle_supported
Apr 07 22:59:04 pc1 org.gtk.Private.GPhoto2VolumeMonitor↙
   [2006]: ### debug: in handle_list
Apr 07 22:59:04 pc1 org.gtk.Private.GoaVolumeMonitor[2006]: #↙
   ## debug: in handle_supported
Apr 07 22:59:04 pc1 org.gtk.Private.GoaVolumeMonitor[2006]: #↙
   ## debug: in handle_list
Apr 07 23:03:09 pc1 org.gtk.Private.GPhoto2VolumeMonitor↙
   [2006]: ### debug: Name owner ':1.4320' vanished
Apr 07 23:03:09 pc1 org.gtk.Private.GoaVolumeMonitor[2006]: #↙
   ## debug: Name owner ':1.4320' vanished
Apr 07 23:03:09 pc1 org.gtk.Private.AfcVolumeMonitor[2006]: #↙
   ## debug: Name owner ':1.4320' vanished
Apr 07 23:03:09 pc1 org.gtk.Private.MTPVolumeMonitor[2006]: #↙
   ## debug: Name owner ':1.4320' vanished
```

Many of the above options can be combined to display only the journal entries you are looking for. The man-page `man journalctl` describes all options in detail.

7.18.6   Sources journald

```
man systemd-journald
man journald.conf
man journalctl
man systemd-cat
```

and online for packages not installed by default

journal gatewayd

journal remote

journal upload

Last edited: 2022/04/10