Überblick



Linux Workshop

Handout: Interaktives Lernen

Wozu dieses Handout?

Dieses Handout begleitet das Arbeitsheft *Interaktives Lernen* und die Folien des 5-Tage-Kurses. Die Spalte *Linux kompakt* nennt die Kapitel des Handbuchs zum Nachlesen. Es fasst für jedes der sieben Module und das optionale Zusatzmodul die wichtigsten **Befehle, Dateien, Lösungswege und Merksätze** auf einer bis zwei Seiten zusammen. Nutzen Sie es zum Nachschlagen und zum Vergleichen mit Ihren eigenen Einträgen im Arbeitsheft. Es ersetzt die Übungen nicht: Verstehen entsteht durch eigenes Ausprobieren.

Die Module auf einen Blick

Modul	Thema	Inhalt in Stichworten	*Linux kompakt*
1	Login Prompt	Standard In / Out / Error	
2	Das System kennenlernen	Steckbrief, Befehle finden, `find`, `.bashrc` und Aliase	2, 3, 11, 53
3	Benutzer, Gruppen, SUID/SGID	`useradd`, `/etc/passwd`, SGID-Verzeichnis, SUID, Sticky-Bit	20, 22
4	`sudo`: `visudo` und `sudoers.d`	Regeln delegieren, Drop-ins, riskante Freigaben	18, 21, 51
5	SSH: Schlüssel und Härtung	`ssh-keygen`, `sshd_config.d`, Root sperren, `scp`	41, 51
6	`systemctl`, `journalctl`, `hostnamectl`	`systemd` ist der Init-Prozess (**PID!** 1) und Dienstverwalter für fast aller großen Distributionen.	32, 32.4 32.6
7	Webserver: Apache und Nextcloud	Virtueller Host, MariaDB, fehlende PHP-Module selbst finden, Nextcloud	45, 50, 57
8	Troubleshooting: Selbsthilfe	Checkliste, Symptome, Fehlerbilder und Lösungen	53, 54
Z	Festplatten Formatieren, mounten	Zusätzliche Festplatten Formatieren und Einfügen.	26
Z	LVM und RAID	`mdadm`, Ausfall und Reparatur, `pvcreate`, `lvextend`, `fstab`	27
Z	ACL auf Dateien und Ordnern	`setfacl`, `getfacl`, Default-ACL, Maske, Sichern	22
Z	Firewall einrichten und Konfigurieren	`ufw`, `firewall-ctl`	39, 43

Testumgebung

- Debian oder Ubuntu in einer virtuellen Maschine.
- **Benutzername = user01**
- **Passwort = Start1234**

Workshop-Vorbereitung: Szenario und Zugangsdaten

Für die folgenden Übungen und Praxisteile verwenden wir eine fiktive Unternehmensstruktur, um reale Berechtigungs- und Verwaltungsszenarien (z. B. Dateifreigaben und Gruppenstrukturen) praxisnah abzubilden.

Fiktive Unternehmensstruktur

Das Unternehmen ist in folgende Abteilungen und Rollen unterteilt:

- **Leitung:** Gesamtzugriff und administrative Überprüfung
- **Office:** Verwaltung, Dokumentation und interne Organisation
- **Lager:** Logistik, Bestandsverwaltung und Dateiaustausch
- **Vertrieb:** Kundenkontakt, Angebote und externer Datenaustausch

Standard-Zugangsdaten für Nextcloud

Um während des Workshops einheitliche Tests durchzuführen, nutzen wir für die benötigte MariaDB den Datenbankuser und Passwort.

Kategorie	**Standard**
MariaDB Benutzername	`nc_user`
MariaDB Passwort	`nc_passwort`
Administrator	admin
Administrator PW	Start1234

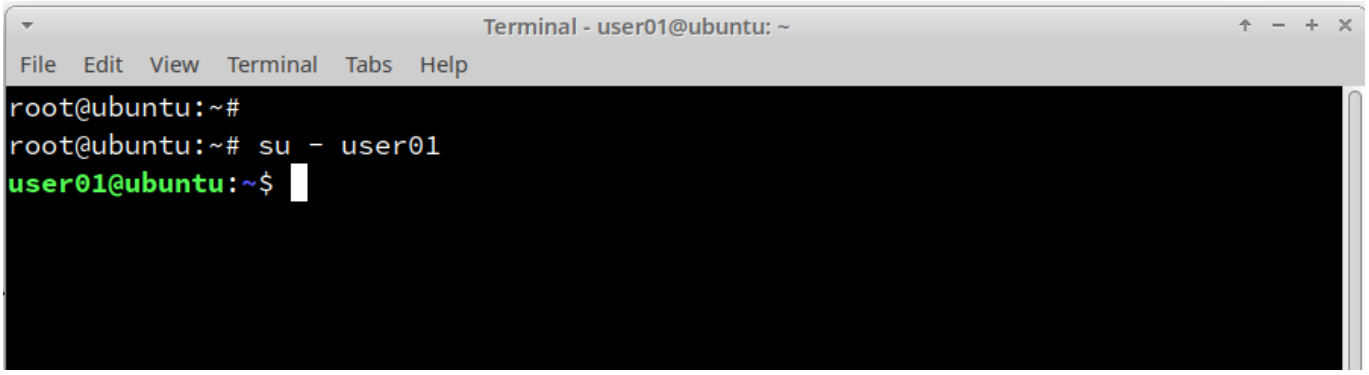
Merke: Diese vereinfachten Standard-Passwörter dienen ausschließlich dem reibungslosen Ablauf im Workshop-Umfeld und entsprechen ausdrücklich nicht den Best Practices für reale Produktionsumgebungen!

Legende

`befehl`	Befehle und Dateinamen stehen in Schreibmaschinenschrift.
Merke	Grüner Kasten: Kernaussage, die Sie sich einprägen sollten.
Achtung	Roter Kasten: Risiko für Datenverlust oder Aussperren.

Modul 1 - Prompt

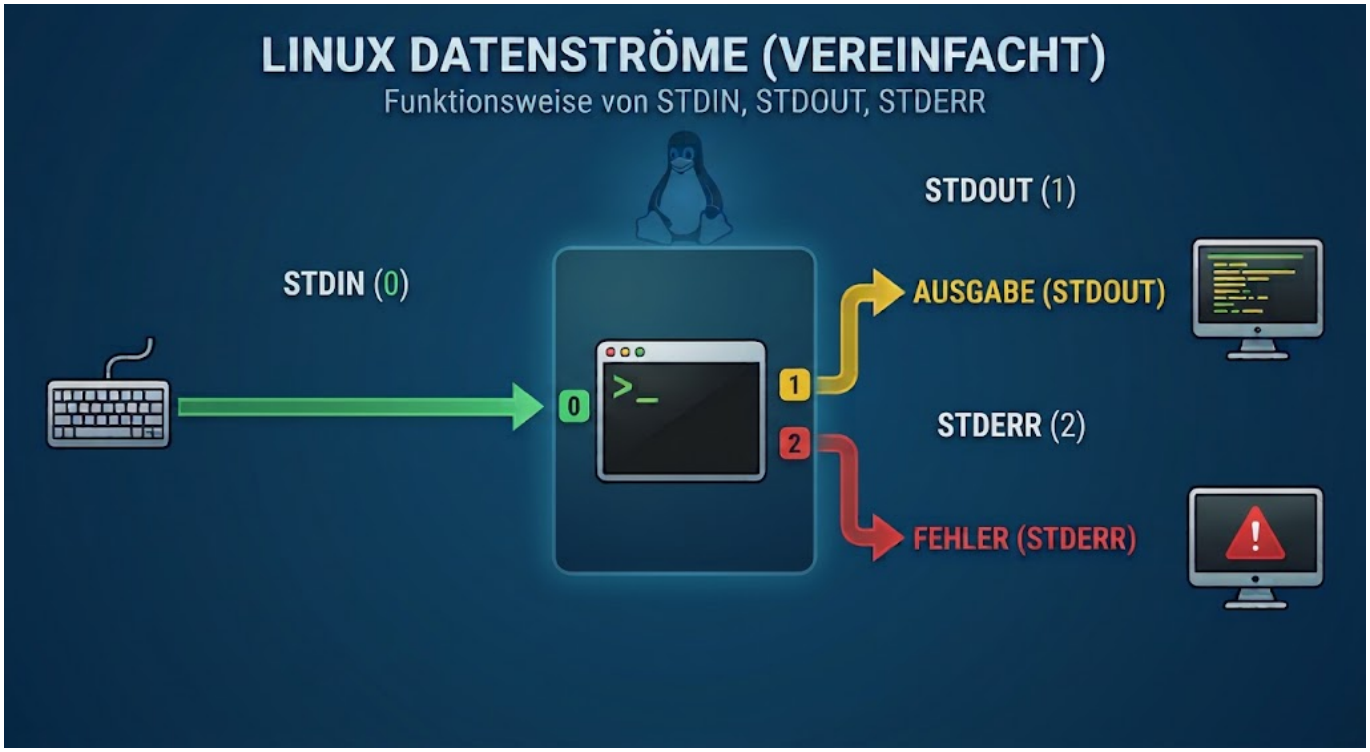
Login Prompt



Unter Linux unterscheidet man den `root`-Benutzer (den Administrator mit allen Rechten) und einen normalen Benutzer direkt am Login-Prompt bzw. an der Eingabeaufforderung (Shell) an einem ganz bestimmten Symbol am Ende der Zeile:

- Normaler Benutzer (user): Die Zeile endet mit einem Dollar-Zeichen ($).
- Root-Benutzer (root): Die Zeile endet mit einer Raute bzw. einem Hashtag (#).

Standard In / Out / Error



Unter Linux kommunizieren Prozesse über drei standardisierte Datenströme (auch Kanäle oder File Descriptors genannt) miteinander:
Standard Input (`stdin` / Kanal 0): Der Eingang für Daten in einen Befehl oder Prozess. Die Standardquelle ist in der Regel die Tastatur.
Standard Output (`stdout` / Kanal 1): Der reguläre Ausgabekanal für erfolgreiche Programmergebnisse. Das Standardziel ist der Bildschirm (das Terminal).
Standard Error (`stderr` / Kanal 2): Der separate Kanal für Fehlermeldungen und Diagnosedaten. Auch hier ist das Standardziel der Bildschirm, allerdings getrennt von der normalen Ausgabe, damit Fehler gezielt abgefangen oder umgeleitet werden können.

Modul 2 – Das System kennenlernen

Steckbrief: Wer bin ich, wo bin ich & was habe ich?

Frage	**Befehl**
Distribution und Version	`cat /etc/os-release`
Kernel	`uname -r`
Rechnername, Systeminfo	`hostnamectl`
Laufzeit	`uptime -p`
CPU-Kerne, Arbeitsspeicher	`nproc`, `free -h`
Platten und freier Platz	`lsblk -f`, `df -h /`
Mountpunkte und die Parameter	`mount | column -t`
IP-Adressen (kurz)	`ip -br a`
Routingtabellen	ip r
Dateibaum anzeigen	ls -la
Benutzer und Gruppen	`id`

Befehle selbst finden

Frage	**Befehl**
Laufende Dienste	`systemctl list-units --type=service --state=running`
Lauschende Ports mit Prozess	`ss -tulpn`
Anzahl installierter Pakete	`dpkg -l | wc -l`
Die drei größten Speicherfresser	`ps aux --sort=-%mem | head -4`
Konto von `root`	`getent passwd root`
Anzahl der Benutzerkonten	`getent passwd | wc -l`

Suchen mit `find`

Aufgabe	**Befehl**
Dateien auf `.conf` unter `/etc` zählen	`find /etc -name '*.conf' | wc -l`
Die drei größten Dateien in `/var/log`	`du -ah /var/log | sort -rh | head -3`
Programme mit SUID-Bit	`find / -perm -4000 -type f 2>/dev/null`

Suchen mit `grep`

Aufgabe	**Befehl**
Nach einem bestimmten Begriff in einer Datei suchen	`grep "error" /var/log/syslog`
Rekursiv nach Text in Ordnern suchen (ignoriert Groß-/Kleinschreibung)	`grep -rnwi "config" /etc/`
Zeilen anzeigen, die ein bestimmtes Muster nicht enthalten (invertiert)	`grep -v "root" /etc/passwd`

Der Verzeichnisbaum nach FHS

Der **Filesystem Hierarchy Standard** (**FHS!**) legt fest, was wo liegt.
Die Struktur ist auf allen Distributionen sehr ähnlich.

Verzeichnis	Inhalt
`/`	Wurzel des gesamten Baums
`/bin`, `/sbin`	Grundlegende Programme (Benutzer / Administration); auf modernen Systemen Links nach `/usr/bin` und `/usr/sbin`
`/boot`	Kernel, Initramfs, Bootloader-Dateien
`/dev`	Gerätedateien (`sda`, `tty`, `null`, `random`)
`/etc`	**Systemweite Konfigurationsdateien** (nur Text)
`/home`	Heimatverzeichnisse der Benutzer (`/home/tux`)
`/lib`, `/lib64`	Bibliotheken und Kernelmodule (Link nach `/usr/lib`)
`/media`	Einhängepunkte für Wechselmedien (**USB!**, **DVD!**)
`/mnt`	Manuell eingehängte Dateisysteme
`/opt`	Zusatzsoftware einzelner Hersteller
`/proc`	Pseudo-Dateisystem: Prozess- und Kernelinformationen
`/root`	Heimatverzeichnis des Benutzers `root`
`/run`	Laufzeitdaten seit dem Systemstart (**PID!**-Dateien, Sockets)
`/srv`	Daten von Diensten (Web, **FTP!**)
`/sys`	Pseudo-Dateisystem: Geräte und Treiber
`/tmp`	Temporäre Dateien (bei Neustart gelöscht, für alle beschreibbar)
`/usr`	Programme, Bibliotheken, Dokumentation (`/usr/bin`, `/usr/share`, `/usr/local`)
`/var`	Veränderliche Daten: Logs (`/var/log`), Mail, Caches, Datenbanken (`/var/lib`)

Merke

Konfiguration liegt in `/etc`, **Programme** in `/usr`, **Veränderliches** in `/var`, **Benutzerdaten** in `/home`. Wer diese vier Orte kennt, findet auf jedem Linux-System das Wichtigste.

Pseudo-Dateisysteme

`/proc` und `/sys` existieren nur im Arbeitsspeicher.
Sie machen Kerneldaten als Dateien lesbar, teilweise auch schreibbar:

```
$ cat /proc/cpuinfo | head -5              # Prozessorinformationen
$ cat /proc/meminfo | head -3              # Speicher
$ cat /proc/uptime                         # Laufzeit in Sekunden
$ cat /proc/$$/status | head -3            # Daten der eigenen Shell ($$ = deren PID)
$ ls /sys/class/net                        # Netzwerkschnittstellen
# echo 1 > /proc/sys/net/ipv4/ip_forward   # Kernelparameter (bis zum Neustart)
```

Die Shell anpassen: `.bashrc` und Aliase

Ein Alias ist eine Abkürzung für einen längeren Befehl. Damit er in jedem neuen Terminal gilt, steht er in `~/.bashrc`. Die Bash liest diese Datei bei jeder *interaktiven* Shell, also bei jedem neuen Terminalfenster.

```
# --- Meine Aliase (am Ende von ~/.bashrc) ---
alias ll='ls -lah'
alias ..='cd ..'
alias ports='ss -tlnp'
alias meinip='ip -br a'
alias upd='sudo apt update'
```

Befehl	**Wirkung**
`cp ~/.bashrc ~/.bashrc.bak`	Sicherung vor der Änderung
`alias`	alle Aliase anzeigen
`source ~/.bashrc` (oder `. ~/.bashrc`)	Datei im laufenden Terminal neu einlesen
`type ll`	zeigt, ob `ll` Alias, Builtin oder Programm ist
`\ll`	Alias einmalig umgehen
`unalias ll`	Alias für diese Sitzung entfernen

Merke: Ein nur im Terminal definierter Alias gilt *nicht* in neuen Fenstern. Aliase werden in Skripten nicht ausgewertet, weil Skripte nicht interaktiv laufen. Dort nutzen Sie den vollen Befehl oder eine Funktion.

Modul 2 – Benutzer, Gruppen und SUID/SGID

Benutzer und Gruppe anlegen

```
sudo useradd -m -s /bin/bash anna               # ebenso ben und cora
sudo useradd -m -d /home/anna -s /bin/bash anna # Explizit angabe von /home
sudo passwd anna
sudo groupadd projekt
sudo usermod -aG projekt anna         # -aG haengt an, ohne andere Gruppen zu loeschen
sudo usermod -aG projekt ben          # cora bleibt ausserhalb
id anna
```

Nach einer Gruppenänderung braucht der Benutzer eine neue Anmeldung, damit die Gruppe wirkt.

Die Kontodateien

Ein Eintrag in `/etc/passwd` hat sieben durch Doppelpunkte getrennte Felder:

```
tux:x:1000:1000:Tux Pinguin,,,:/home/tux:/bin/bash
 |  |  |    |        |            |          +- Login-Shell
 |  |  |    |        |            +------------ Heimatverzeichnis
 |  |  |    |        +------------------------- Kommentar (GECOS): vollstaendiger Name
 |  |  |    +---------------------------------- primaere GID
 |  |  +--------------------------------------- UID
 |  +------------------------------------------ Passwort ('x': Hash steht in /etc/shadow)
 +--------------------------------------------- Benutzername
```

Merke: `/etc/passwd` ist für alle lesbar (`644`), weil Programme UIDs in Namen umsetzen. Die Hashes stehen deshalb in `/etc/shadow` (`640`, nur `root` und Gruppe `shadow`).

`/etc/shadow` hat neun Felder, getrennt durch Doppelpunkte:

```
root:$6$rounds=5000$xyz...:19800:0:99999:7:::
  |      |               |     |     |    | |+--- Reserviert (leer)
  |      |               |     |     |    | +---- Inaktiven-Tage nach Ablauf
  |      |               |     |     |    +------ Warn-Tage vor Ablauf
  |      |               |     |     +----------- Maximale Gültigkeit in Tagen
  |      |               |     +----------------- Minimale Tage bis Passwort-Änderung
  |      |               +----------------------- Letzte Änderung (Epoche-Tage seit 1970)
  |      +--------------------------------------- Verschlüsselter Passwort-Hash (inkl. Salt)
  +---------------------------------------------- Benutzername
```

Merke: `/etc/shadow` ist für normale Benutzer nicht lesbar (Rechte `640`, Besitzer `root`, Gruppe `shadow`). Mit `sudo` läuft der Befehl als `root` und gelingt.

SGID am Gruppenverzeichnis

```
sudo mkdir /srv/projekt
sudo chgrp projekt /srv/projekt
sudo chmod 770 /srv/projekt
sudo chmod g+s /srv/projekt           # oder: chmod 2770
```

Ohne SGID erhält eine neue Datei die *Primärgruppe* des Erzeugers, mit SGID die Gruppe des Verzeichnisses (`projekt`). `cora` wird mit `Permission denied` abgewiesen.

Sondermodi auf einen Blick

Bit	Wirkung und Beispiel
SUID (`4000`, `u+s`)	Programm läuft mit den Rechten des *Besitzers*. Beispiel: `/usr/bin/passwd` ändert `/etc/shadow`.
SGID (`2000`, `g+s`)	Programm läuft mit der Gruppe des Programms; an einem *Verzeichnis* erben neue Dateien die Gruppe.
Sticky (`1000`, `+t`)	In einem Verzeichnis darf nur der Besitzer (oder `root`) eigene Dateien löschen. Beispiel: `/tmp`.

Oktal	Symbolisch und Wirkung
`4755`	`rwsr-xr-x`: SUID, ausführbar für alle
`2770`	`rwxrws---`: SGID-Verzeichnis, nur Besitzer und Gruppe
`1777`	`rwxrwxrwt`: Sticky-Verzeichnis wie `/tmp`
`4644`	`rwSr--r--`: großes `S` heißt SUID gesetzt, aber *ohne* Ausführungsrecht, also wirkungslos

SUID-Experiment und Sticky-Test

```
sudo cp /usr/bin/id /usr/local/bin/id_suid
sudo chmod u+s /usr/local/bin/id_suid
/usr/local/bin/id_suid                # zeigt euid=0(root), uid bleibt anna
sudo rm /usr/local/bin/id_suid        # Experiment aufraeumen

sudo mkdir /srv/tausch && sudo chmod 1777 /srv/tausch
ls -ld /tmp /srv/tausch               # beide mit t am Ende
```

Im Sticky-Verzeichnis darf `ben` die Datei von `anna` *nicht* löschen, obwohl er dort schreiben darf.

Achtung: Geben Sie niemals SUID an Shells, Editoren oder Interpreter (`bash`, `vim`, `python`). Wer sie startet, erhält Root-Rechte. Alle SUID-Programme finden Sie mit `find / -perm -4000 -type f 2>/dev/null`.

Modul 4 – **sudo**: **visudo** und **sudoers.d**

Achtung: Ein Syntaxfehler in der sudoers-Konfiguration kann sudo für *alle* unbrauchbar machen. Bearbeiten Sie die Dateien **nur mit visudo**, nie direkt mit nano oder vim, und halten Sie eine zweite Root-Sitzung (sudo -i) offen. Sollte nano als default Editor eingestellt sein so kann man das anpassen sudo update-alternatives --config editor

Modul 3 - Benuzter & Gruppen

```
ben   ALL=(root)   /usr/bin/systemctl restart apache2
```

Teil	Bedeutung
ben	wer: Benutzer (Gruppen mit %, zum Beispiel %projekt)
ALL (vor dem =)	wo: auf welchen Rechnern die Regel gilt
(root)	als wen: Zielbenutzer ((ALL:ALL) = beliebiger Benutzer und Gruppe)
/usr/bin/systemctl restart apache2	was: Befehl mit *vollem Pfad* und genauen Argumenten

Ausgangslage: Die Gruppe %sudo (Debian, Ubuntu) beziehungsweise %wheel (Red Hat) hat in /etc/sudoers die Regel ALL=(ALL:ALL) ALL. Die Zeile @includedir /etc/sudoers.d (älter #includedir) liest das Verzeichnis /etc/sudoers.d ein. Ein Benutzer ohne Regel, etwa ben, erhält bei sudo ls /root die Meldung „is not in the sudoers file“; der Versuch wird im Journal und in /var/log/auth.log protokolliert.

Regel mit **visudo** als Drop-in anlegen

```
sudo visudo -f /etc/sudoers.d/20-ben-web
# Inhalt:
ben ALL=(root) /usr/bin/systemctl restart apache2, /usr/bin/systemctl reload apache2

command -v systemctl           # absoluten Pfad ermitteln
sudo visudo -c                 # gesamte Konfiguration pruefen
ls -l /etc/sudoers.d           # Rechte 0440, Besitzer root
sudo -l -U ben                 # Rechte von ben als Administrator anzeigen
```

Bei einem Syntaxfehler meldet visudo den Fehler und fragt mit „What now?“: e (erneut bearbeiten), x (beenden ohne Speichern) oder Q (trotzdem speichern, gefährlich). Die Datei wird erst nach erfolgreicher Prüfung übernommen. visudo sperrt die Datei außerdem gegen gleichzeitiges Bearbeiten.

Befehl als ben	Ergebnis
sudo systemctl restart apache2	erlaubt
sudo systemctl stop apache2	abgelehnt, anderer Unterbefehl
sudo systemctl restart mariadb	abgelehnt, anderer Dienst
sudo cat /etc/shadow	abgelehnt
sudo -l	listet die erlaubten Befehle

Apache wird erst in Modul 6 installiert. Die Regeln prüfen Sie hier mit sudo visudo -c und sudo -l -U ben; die Neustart-Befehle der Tabelle testen Sie nach Modul 6.

Merke: Dateien in /etc/sudoers.d mit einem *Punkt* oder *Tilde* im Namen werden ignoriert (zum Beispiel 20-ben-web.bak), damit Paketsicherungen nicht wirken. Verwenden Sie daher Namen ohne Punkt, mit Zahlenpräfix für die Reihenfolge.

Gruppen, Aliase und NOPASSWD

```
# /etc/sudoers.d/30-webadmins   (mit: sudo visudo -f ...)
Cmnd_Alias WEBDIENST = /usr/bin/systemctl restart apache2, \
                       /usr/bin/systemctl reload apache2
%projekt   ALL=(root)   WEBDIENST

# Einzelne Regel ohne Passwort:
anna ALL=(root) NOPASSWD: /usr/bin/systemctl reload apache2
```

anna und ben (Gruppe `projekt`) dürfen `reload`, `cora` nicht. Nach einer Gruppenänderung ist eine neue Anmeldung nötig. `NOPASSWD` setzen Sie sparsam und nur für genau benannte, ungefährliche Befehle ein.

Riskante Freigaben erkennen

Freigabe für ben	**Gefährlich?**	**Begründung**
`ALL=(ALL) ALL`	ja	volle Root-Rechte
`ALL=(root) NOPASSWD: ALL`	ja	volle Root-Rechte ohne jede Rückfrage
`ALL=(root) /usr/bin/vim`	ja	Shell mit `:!sh` oder `:shell`, dann Root
`ALL=(root) /usr/bin/less /var/log/syslog`	ja	Shell mit `!sh` im Pager
`ALL=(root) /usr/bin/find`	ja	`find . -exec /bin/sh \;` startet eine Root-Shell
`ALL=(root) /usr/bin/systemctl restart apache2`	nein	genau ein Befehl mit festen Argumenten

Merke: Je genauer der Befehl, desto kleiner die Angriffsfläche. Programme, die Shells, Befehle oder Pager starten können (Editoren, `less`, `find`, `awk`), geben praktisch Root. Für einzelne Dateien nutzen Sie `sudoedit` mit genau benannter Datei.

Protokoll, Zeitstempel und Rettung

```
sudo -k                                          # Passwort-Zeitstempel loeschen
sudo true ; sudo true                            # nur der erste Aufruf fragt nach dem Passwort
sudo journalctl _COMM=sudo -n 10 --no-pager      # wer hat wann was als root getan
```

Wenn `sudo` wegen eines Fehlers ausfällt, kommen Sie so wieder an Root: eine vorher geöffnete Root-Sitzung nutzen, `su -` mit dem Root-Passwort, `pkexec visudo`, oder über den Wiederherstellungsmodus beziehungsweise das Boot-Menü (`init=/bin/bash`), dann die fehlerhafte Datei korrigieren.

Modul 5 – SSH: Schlüssel und Härtung

Achtung: Halten Sie bei jeder Änderung am SSH-Server **eine zweite Sitzung offen**. Prüfen Sie mit `sudo sshd -t`, laden Sie neu und testen Sie die Anmeldung in einem *neuen* Fenster. Schließen Sie die alte Sitzung erst nach erfolgreichem Test.

Server, Fingerabdruck, erste Verbindung

```
sudo apt install openssh-server
systemctl status ssh ; ss -tlnp                          # sshd lauscht auf Port 22
ssh anna@IP-des-Servers                                  # erste Frage: Fingerabdruck
ssh-keygen -lf /etc/ssh/ssh_host_ed25519_key.pub         # auf dem Server vergleichen
```

Die Rückfrage beim ersten Verbinden schützt vor einem untergeschobenen Server (Man-in-the-Middle). Bestätigte Server stehen in `~/.ssh/known_hosts`.

Wichtige Dateien

Datei	Seite	Zweck
`/etc/ssh/sshd_config`	Server	Hauptkonfiguration des Servers
`/etc/ssh/sshd_config.d/*.conf`	Server	Drop-ins, eingelesen über `Include`
`/etc/ssh/ssh_config`	Client	systemweite Client-Einstellungen
`~/.ssh/config`	Client	persönliche Einträge (`Host labserver`)
`~/.ssh/known_hosts`	Client	Fingerabdrücke bekannter Server
`~/.ssh/authorized_keys`	Server	öffentliche Schlüssel, die sich anmelden dürfen
`~/.ssh/id_ed25519`, `.pub`	Client	privater (geheim) und öffentlicher Schlüssel

Schlüsselanmeldung einrichten

```
ssh-keygen -t ed25519 -C 'anna@lab'        # mit Passphrase
ssh-copy-id anna@IP-des-Servers            # einmal Kontopasswort noetig
ssh anna@IP-des-Servers                    # fragt jetzt nach der Passphrase
eval "$(ssh-agent)" ; ssh-add              # optional: Passphrase nur einmal
```

Merke: Der *private* Schlüssel verlässt nie Ihren Rechner (Rechte `600`). `~/.ssh` braucht `700`, `authorized_keys` `600`. Bei `666` lehnt `sshd` den Schlüssel ab; die Meldung steht in `sudo journalctl -u ssh -n 5`.

`sshd_config` und `sshd_config.d`

```
grep -vE '^#|^$' /etc/ssh/sshd_config      # aktive Zeilen, Include-Zeile suchen
ls /etc/ssh/sshd_config.d/
sudo sshd -T | grep -i maxauthtries        # tatsaechlich wirksamer Wert
```

Merke: `sshd` verwendet bei doppelten Direktiven den *ersten* gelesenen Wert. Steht die `Include`-Zeile am Anfang der Hauptdatei (Debian 12, Ubuntu), gewinnt das Drop-in; die Dateien werden alphabetisch gelesen.

Härtung: Root sperren, nur Schlüssel

```
# /etc/ssh/sshd_config.d/10-haertung.conf
PermitRootLogin        no
PasswordAuthentication no
PubkeyAuthentication   yes
MaxAuthTries           3
AllowUsers             anna ben
X11Forwarding          no
```

```
sudo sshd -t && sudo systemctl reload ssh      # erst Syntax, dann neu laden
sudo journalctl -u ssh -n 20                   # oder /var/log/auth.log
```

Versuch (neues Fenster)	Ergebnis
`ssh root@IP`	abgelehnt (`PermitRootLogin no`)
`ssh anna@IP` mit Schlüssel	gelingt
`ssh -o PubkeyAuthentication=no anna@IP`	abgelehnt (keine Passwörter erlaubt)
`ssh cora@IP`	abgelehnt (nicht in `AllowUsers`)

Aktivieren Sie `PasswordAuthentication no` erst, wenn der Schlüssel-Login nachweislich funktioniert.

Komfort, Tunnel und scp

```
# ~/.ssh/config auf dem Client
Host labserver
    HostName     192.0.2.10
    User         anna
    IdentityFile ~/.ssh/id_ed25519
```

Befehl	Wirkung
`ssh labserver`	Anmeldung über den Eintrag
`ssh -L 8080:localhost:80 labserver`	Tunnel: `http://localhost:8080` zeigt den Apache des Servers (nach Modul 6)
`scp hallo.txt anna@IP:/tmp/`	Datei hochladen
`scp anna@IP:/etc/hostname .`	Datei herunterladen
`scp -r testordner anna@IP:/tmp/`	Verzeichnis kopieren
`scp hallo.txt labserver:`	in das Home-Verzeichnis, mit dem Eintrag
`scp -P 2222 datei anna@IP:`	anderer Port (großes `P`)

Merke: Bei `scp` trennt der *Doppelpunkt* Rechner und Pfad. Ohne Doppelpunkt (`scp hallo.txt anna@IP`) entsteht lokal nur eine neue Datei namens `anna@IP`.

Modul 6 – Dienstverwalter

`systemd` ist der Init-Prozess (**PID!** 1) und Dienstverwalter fast aller großen Distributionen. Er startet Dienste parallel, überwacht sie, protokolliert ihre Ausgaben (`journald`) und bietet einheitliche Verwaltung über das Kommando `systemctl`.

```
$ systemctl status apache2              # Zustand, PID, letzte Logzeilen
# systemctl start apache2               # jetzt starten
# systemctl stop apache2                # jetzt stoppen
# systemctl restart apache2             # stoppen und starten
# systemctl reload apache2              # Konfiguration neu laden, ohne Neustart
# systemctl enable apache2              # beim Systemstart starten
# systemctl disable apache2             # nicht mehr beim Start
# systemctl enable --now apache2        # aktivieren und sofort starten
# systemctl mask apache2                # komplett sperren (auch manuell nicht startbar)
# systemctl unmask apache2
$ systemctl is-active apache2           # active / inactive / failed
$ systemctl is-enabled apache2
```

```
$ systemctl list-units --type=service              # laufende Dienste
$ systemctl list-unit-files --type=service         # alle installierten Dienste + Autostart
$ systemctl --failed                               # fehlgeschlagene Units
$ systemctl list-dependencies multi-user.target    # Abhaengigkeitsbaum
$ systemctl get-default                            # Standard-Target
# systemctl set-default multi-user.target          # Standard-Target aendern
# systemctl isolate rescue.target                  # jetzt in anderes Target wechseln
```

Modul 7 – Webserver: Apache und Nextcloud

Pakete und Apache-Aufbau (Debian und Ubuntu)

```
sudo apt install apache2 mariadb-server php libapache2-mod-php curl bzip2
systemctl status apache2 ; curl -I http://localhost      # Statuscode 200
php -v                                                    # PHP ist da
```

Installiert werden hier nur Apache, MariaDB und PHP selbst. Die zusätzlichen PHP-Module, die Nextcloud braucht, fehlen *absichtlich*: Sie finden sie weiter unten selbst.

Frage	**Antwort**
Hauptkonfiguration	`/etc/apache2/apache2.conf`
Aktive Sites, Module	`ls /etc/apache2/sites-enabled`, `apache2ctl -M`
Virtuelle Hosts im Überblick	`apache2ctl -S`
Protokolle	`/var/log/apache2/` (`access.log`, `error.log`)
Lauschender Port und Prozess	`sudo ss -tlnp | grep :80`
Syntax prüfen, dann neu laden	`sudo apache2ctl configtest && sudo systemctl reload apache2`

Eigener virtueller Host

```
sudo mkdir /var/www/hallo        # index.html mit den Namen des Paars anlegen
# /etc/apache2/sites-available/hallo.conf
<VirtualHost *:80>
    ServerName   hallo.lab.local
    DocumentRoot /var/www/hallo
    ErrorLog  ${APACHE_LOG_DIR}/hallo_error.log
    CustomLog ${APACHE_LOG_DIR}/hallo_access.log combined
    <Directory /var/www/hallo>
        Require all granted
    </Directory>
</VirtualHost>
```

```
sudo a2ensite hallo && sudo apache2ctl configtest && sudo systemctl reload apache2
# /etc/hosts:  <eigene-IP>  hallo.lab.local
curl -H 'Host: hallo.lab.local' http://localhost
```

Merke: Apache wählt den virtuellen Host nach dem *Host-Header* der Anfrage. `curl http://localhost` landet im Standard-Host, `curl -H 'Host: hallo.lab.local'` in Ihrer Site.

Datenbank-Setup (MariaDB für Nextcloud)

Für die automatisierte bzw. vereinfachte Einrichtung der MariaDB-Datenbank dienen folgende Befehle:

```
sudo mysql
CREATE DATABASE nextcloud CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
CREATE USER 'nc_user'@'localhost' IDENTIFIED BY 'nc_passwort';
GRANT ALL PRIVILEGES ON nextcloud.* TO 'nc_user'@'localhost';
FLUSH PRIVILEGES;
```

Datenbanken und Benutzer in MariaDB überprüfen

Überprüfung der korrekten Erstellung von Benutzer und Datenbank:

```
SHOW DATABASES;
SELECT host, user FROM mysql.user;
```

MariaDB-Einträge bereinigen und zurücksetzen

Bereinigung (Benutzer, Datenbank und Berechtigungen in (`sudo mysql`):

```
DROP DATABASE IF EXISTS nextcloud;
DROP USER IF EXISTS 'nc_user'@'localhost';
FLUSH PRIVILEGES;
```

Nextcloud herunterladen

```
cd /tmp
curl -LO https://download.nextcloud.com/server/releases/latest.tar.bz2
sudo tar -xjf latest.tar.bz2 -C /var/www
sudo chown -R www-data:www-data /var/www/nextcloud
```

Fehlende PHP-Module selbst finden

Die Grundinstallation von PHP reicht für Nextcloud nicht. Finden Sie heraus, welche PHP-Module fehlen, und installieren Sie die passenden Pakete. Vorgehen:

- ☐ **Anforderungen klären:** Systemvoraussetzungen im Nextcloud-Admin-Handbuch nachlesen, oder Nextcloud selbst fragen (Schritt 3)
- ☐ **Ist-Zustand erfassen:** welche Module PHP schon hat
- ☐ **Nextcloud prüfen:** die Fehlermeldung nennt fehlende Module
- ☐ **Paket suchen:** zu jedem fehlenden Modul das Debian-Paket finden und installieren
- ☐ **Apache neu starten** und erneut prüfen, bis keine Meldung mehr kommt

Frage	**Befehl**
Welche PHP-Module sind geladen?	`php -m`
Ist ein bestimmtes Modul da?	`php -m | grep -i gd`
Welche PHP-Pakete gibt es?	`apt search '^php-'`
Ist das Paket installiert?	`apt policy php-gd`
Was enthält ein Paket?	`apt show php-xml`, `dpkg -L php-xml`
Fehlendes Modul von Nextcloud melden lassen	`sudo -u www-data php /var/www/nextcloud/occ status`

```
sudo apt install php-MODUL                # Paket aus der Suche
sudo systemctl restart apache2            # Apache lädt neue Module erst jetzt
php -m | grep -i MODUL                    # Modul vorhanden?
sudo -u www-data php /var/www/nextcloud/occ status
```

Merke: Modulname und Paketname stimmen nicht immer überein: Ein Paket kann mehrere Module liefern (`php-xml` enthält `dom`, `simplexml`, `xmlreader` und `xmlwriter`), und das Datenbankmodul `pdo_mysql` steckt im Paket `php-mysql`. Das Kommandozeilen-PHP (`php -m`) und das Apache-PHP können unterschiedliche Konfigurationen haben, die Paketliste ist aber dieselbe.

Nextcloud einrichten

```
sudo a2enmod rewrite headers env dir mime
sudo a2ensite nextcloud
sudo apache2ctl configtest && sudo systemctl restart apache2
```

```
# /etc/apache2/sites-available/nextcloud.conf
<VirtualHost *:80>
    ServerName   cloud.lab.local
    DocumentRoot /var/www/nextcloud
    <Directory /var/www/nextcloud>
        Require all granted
        AllowOverride All
        Options FollowSymLinks MultiViews
    </Directory>
</VirtualHost>
```

Erstinstallation auf der Kommandozeile

Die Erstinstallation (Datenbank anbinden, Administrator anlegen) ersetzt den Installationsassistenten im Browser. Es ist *ein* Weg: Alternativ rufen Sie `http://cloud.lab.local` im Browser auf und füllen das Formular aus. Beide Wege liefern dasselbe Ergebnis.

```
# memory_limit = 512M in /etc/php/<Version>/apache2/php.ini, dann Apache neu starten
cd /var/www/nextcloud
sudo -u www-data php occ maintenance:install \
  --database mysql --database-name nextcloud \
  --database-user nc --database-pass 'DB-PASSWORT' \
  --admin-user admin --admin-pass 'ADMIN-PASSWORT'
sudo -u www-data php occ config:system:set trusted_domains 1 --value=cloud.lab.local
```

Achtung: Passwörter in der Befehlszeile landen in Shell-Historie und Prozessliste. Im Übungssystem vertretbar, im Betrieb nicht: dort interaktiv eingeben oder geschützte Dateien nutzen.

Merke: Hochgeladene Dateien gehören im Dateisystem dem Benutzer `www-data`, nicht dem Nextcloud-Benutzer: Nur der Webserver-Prozess greift auf `/var/www/nextcloud` zu, die Benutzerverwaltung liegt in der Datenbank.

Bonus: HTTPS mit selbstsigniertem Zertifikat

```
sudo a2enmod ssl
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
  -keyout /etc/ssl/private/cloud.key -out /etc/ssl/certs/cloud.crt \
  -subj "/CN=cloud.lab.local"
curl -kI https://cloud.lab.local
```

Im `VirtualHost *:443` stehen `SSLEngine on`, `SSLCertificateFile` und `SSLCertificateKeyFile`. Der Browser warnt, weil das Zertifikat von keiner vertrauenswürdigen Zertifizierungsstelle signiert ist.

Stolpersteine und erste Hilfe

Symptom	**Erster Blick**	**Typische Ursache**
Fehler 500	`/var/log/apache2/error.log`	PHP-Modul fehlt, Rechte, `.htaccess`
„Zugriff über nicht vertrauenswürdige Domain“	`config.php`, `trusted_domains`	Hostname fehlt in der Liste
Seite nicht erreichbar	`ss -tlnp`, Firewall, `systemctl status`	Dienst gestoppt oder Port gesperrt
Datenbankfehler	`systemctl status mariadb`	Dienst aus, falsches Passwort

Zum Abgleich: alle benötigten PHP-Module

Erst nach der eigenen Suche vergleichen. Die Spalte *Paket* zeigt, was Sie installieren mussten.

PHP-Modul	**Paket**	**Einstufung**	**Wofür**
`pdo_mysql`	`php-mysql`	erforderlich	Zugriff auf MariaDB
`gd`	`php-gd`	erforderlich	Bilder, Vorschaubilder
`curl`	`php-curl`	erforderlich	HTTP-Anfragen, App-Store
`zip`	`php-zip`	erforderlich	Archive, Apps entpacken
`dom`, `simplexml`, `xmlreader`, `xmlwriter`	`php-xml`	erforderlich	XML verarbeiten
`mbstring`	`php-mbstring`	erforderlich	Zeichensätze (UTF-8)
`intl`	`php-intl`	empfohlen	Sprachen, Sortierung
`bcmath`	`php-bcmath`	empfohlen	Rechnen mit großen Zahlen
`gmp`	`php-gmp`	empfohlen	große Zahlen, Verschlüsselung
`apcu`	`php-apcu`	empfohlen	lokaler Zwischenspeicher

Die Module `ctype`, `fileinfo`, `posix`, `session`, `openssl` und `zlib` sind bereits durch `php` und `php-common` da.

Modul 8 – Troubleshooting: Selbsthilfe

Die Selbsthilfe-Checkliste

☐	**1. Problem beschreiben**	Was sollte passieren? Was passiert? Seit wann? Was wurde zuletzt geändert?
☐	**2. Meldung lesen**	Fehlermeldung *wörtlich* lesen und notieren, nicht interpretieren.
☐	**3. Protokolle prüfen**	`journalctl -xeu dienst`, `/var/log/`, `dmesg | tail`
☐	**4. Zustand prüfen**	`systemctl status`, `ss -tlnp`, `df -h`, `df -i`, `free -h`, `id`, `ls -l`
☐	**5. Hilfe holen**	`man`, `--help`, `/usr/share/doc`, exakte Fehlermeldung suchen, Team fragen
☐	**6. Eine Änderung nach der anderen**	Backup (`cp -a datei datei.bak`), eine Änderung, testen, notieren, bei Misserfolg zurücknehmen

Symptome und erste Prüfungen

Symptom	**Erste Prüfung**	**Mögliche Ursache**
Dienst startet nicht	`systemctl status`, `journalctl -xeu dienst`	Syntaxfehler, Port belegt, Rechte
`Permission denied`	`ls -l`, `id`, `getfacl`	Besitzer, Gruppe, ACL-Maske, SELinux/AppArmor
Webseite nicht erreichbar	`ss -tlnp`, `curl -I`, `systemctl status`	Dienst aus, Port, Firewall, falsche Adresse
Platte „voll“, obwohl Platz da ist	`df -h`, `df -i`, `lsof +L1`	Inodes erschöpft, gelöschte aber geöffnete Dateien
System ist langsam	`top`, `free -h`	Speicher, Swap, einzelner Prozess, Plattenzugriffe
Name wird nicht aufgelöst	`getent hosts name`, `/etc/hosts`	falscher Eintrag, DNS-Server, `/etc/resolv.conf` !!nicht editieren, wird von SystemD verwaltet!!

Symptom	**Erste Prüfung**	
Keine Netzwerkverbindung, DNS-Namen werden nicht gefunden	`ip a`, `ip r`	Überprüfen ob die Netzwerkinterface und Routing angezeigt wird
	`nmcli connection show`, `nmcli device status`	NetworkManager Kommandos wenn aktiv
Verbindung Prüfen	`nmap IP-Adresse`	nmap Überprüft die offenen Ports des Zielsystems
	`nmap -sT -p 22 IP-Adresse`	Überprüft gezielt nur den TCP Port 22
	dig	DNS lookup utility

Fehlerbilder und ihre Lösung

Achtung: Starten Sie das System nicht neu, solange `/etc/fstab` geändert ist. Testen Sie fstab nur mit `sudo mount -a`.

Nr.	Symptom	Prüfung und Lösung
1	Apache lädt nicht neu	`apache2ctl configtest` zeigt den Tippfehler (`DocumentRot`); korrigieren, neu laden.
2	Apache startet nicht	`ss -tlnp | grep :80`: Port 80 ist belegt; fremden Prozess beenden, Apache starten.
3	Fehler 403	`ls -l` zeigt Rechte `000`; `sudo chmod 644 /var/www/hallo/index.html`.
4	Nextcloud meldet Datenbankfehler	`systemctl status mariadb`; `sudo systemctl start mariadb`.
5	Volume lässt sich nicht einbinden (Zusatzmodul)	`blkid` vergleichen, UUID in `/etc/fstab` korrigieren, `sudo mount -a`.
6	Hochladen scheitert, Volume voll (Zusatzmodul)	`df -h`, `du -ah`; Datei `fuell.img` löschen.
7	Warnung, RAID eingeschränkt (Zusatzmodul)	`cat /proc/mdstat`, `mdadm --detail`; Platte mit `--remove` entfernen und mit `--add` wieder aufnehmen.
8	`ben` und `cora` verlieren Zugriff	`getfacl /srv/doku` zeigt `mask::---`; `sudo setfacl -m m::rwx /srv/doku`.
9	Webseite nicht erreichbar	`systemctl status apache2`; `sudo systemctl enable --now apache2`.
10	Browser hängt, Zeitüberschreitung	`getent hosts cloud.lab.local`: falsche IP in `/etc/hosts`; Eintrag korrigieren.
11	SSH-Dienst startet nicht	`sudo sshd -t` nennt die Zeile (`PermitRootLogn`); korrigieren, `sudo systemctl restart ssh`.
12	Schlüssel-Login wird abgelehnt	`ls -l ~/.ssh/authorized_keys` zeigt `666`; `chmod 600`, Meldung in `journalctl -u ssh`.
13	`ben` verliert seine sudo-Rechte	`sudo -l -U ben` leer; Datei heißt jetzt `20-ben-web.bak` (Punkt im Namen wird ignoriert), zurückbenennen.

Merke: Prüfen Sie immer zuerst die Konfiguration, bevor Sie einen Dienst neu starten: `apache2ctl configtest`, `sshd -t`, `visudo -c`, `mount -a`. Eine Änderung nach der anderen, und jede mit einem Test.

Modul 3 – ACL auf Dateien und Ordnern

Eine ACL (Access Control List) vergibt Rechte an *einzelne* Benutzer oder Gruppen, ohne neue Gruppen anzulegen. Ein `+` am Ende der Rechte-Spalte von `ls -l` zeigt, dass eine ACL vorhanden ist.

Ausgangslage und Rechte vergeben

```
sudo mkdir /srv/doku
sudo chown anna /srv/doku && sudo chmod 700 /srv/doku
sudo setfacl -m u:ben:rx   /srv/doku      # ben: lesen und betreten
sudo setfacl -m u:cora:rwx /srv/doku      # cora: lesen, schreiben, betreten
getfacl /srv/doku
```

Befehl	Wirkung
`setfacl -m u:name:rechte pfad`	Eintrag für einen Benutzer setzen oder ändern
`setfacl -m g:name:rechte pfad`	Eintrag für eine Gruppe
`setfacl -x u:ben pfad`	nur den Eintrag von `ben` entfernen
`setfacl -b pfad` (mit `-R` rekursiv)	alle ACL-Einträge entfernen
`getfacl pfad`	ACL anzeigen

Default-ACL und Maske

Befehl	Wirkung
`setfacl -d -m u:ben:r /srv/doku`	Default-ACL: gilt für *künftige* Dateien und Ordner im Verzeichnis, nicht für bestehende
`setfacl -m m::r datei`	Maske setzen: Obergrenze der effektiven Rechte für alle Einträge außer Besitzer und Andere

Merke: `chmod` auf einer Datei mit ACL ändert die *Maske*. Nach `chmod 600` ist die Maske `---`, und die Rechte der Einträge sind nur noch auf dem Papier vorhanden (`#effective:---`). Mit `setfacl -m m::r datei` stellen Sie die Maske wieder her.

ACL lesen: Rätsel

```
# file: plan.txt
# owner: anna
# group: projekt
user::rw-
user:ben:rw-          #effective:r--
group::r--
mask::r--
other::---
```

- Effektive Rechte: `anna` `rw-` (Besitzer, von der Maske nicht begrenzt), `ben` nur `r--` (Maske), ein Fremder `---`.
- Schreibrechte für `ben` zurück: `setfacl -m m::rw plan.txt`.

Sichern, entfernen, wiederherstellen

```
cd /srv
getfacl -R doku > /tmp/acl.bak          # sichern (Pfade relativ!)
sudo setfacl -R -b doku                 # alle ACLs entfernen, + verschwindet
sudo setfacl --restore=/tmp/acl.bak     # wiederherstellen (im selben Verzeichnis)
sudo setfacl -x u:ben doku              # nur ben entfernen
```

Rechte-Matrix für `/srv/doku`

Benutzer	**Auflisten**	**Anlegen**
`anna` (Besitzer)	ja	ja
`ben` (`r-x`)	ja	nein
`cora` (`rwx`)	ja	ja
Fremder	nein	nein

Zusatzmodul – Festplattenpartitionierung und Dateisysteme

Merke: Dieses Modul behandelt die klassische Einrichtung von Speicher: Vom Auffinden der Geräte über die Partitionierung mit `cfdisk` und das Formatieren bis hin zum permanenten Einhängen über die `/etc/fstab`.

Achtung: Prüfen Sie vor jedem Befehl mit `lsblk`, welche Platte gemeint ist. `/dev/sdb`, `/dev/sdc` sind Platzhalter. Verwenden Sie niemals die aktive Systemplatte (z. B. `sda` oder NVMe-Laufwerke).

Übersicht der Schritte

Schritt	Werkzeug / Befehl	Beschreibung
1. Identifikation	`lsblk`, `fdisk -l`	Festplatten anzeigen und richtige Gerätedatei ermitteln
2. Partitionierung	`cfdisk /dev/sdb`	Interaktiver Partitionierer (GPT/MBR erstellen)
3. Formatierung	`mkfs.xfs`, `mkfs.ext4`	Dateisystem auf die neue Partition schreiben
4. Einhängen	`mount`, `/etc/fstab`	Temporär testen und permanent ins System einbinden

1. Festplatte partitionieren mit cfdisk

```
lsblk                                   # Blockgeraete und Partitionen auflisten
sudo cfdisk /dev/sdb                    # Interaktiven Partitionierer starten
```

- **Bedienung in cfdisk:** Mit den Pfeiltasten (oben/unten) die Platte/den freien Speicher auswählen, mit **[New]** eine neue Partition anlegen, die Größe bestimmen (z. B. `500M` oder komplett leer lassen für den Rest) und unter **[Write]** mit `yes` bestätigen. Über **[Quit]** wird das Programm verlassen.

2. Dateisystem erstellen (Formatieren)

Nach dem Anlegen der Partition (z. B. `/dev/sdb1`) wird das Dateisystem aufgespielt. Hier stehen u. a. `xfs` und `ext4` zur Auswahl:

```
sudo mkfs.xfs /dev/sdb1                 # XFS-Dateisystem erstellen (z.B. für große Daten)
sudo mkfs.ext4 /dev/sdb2                # EXT4-Dateisystem erstellen (Standard für Linux)
```

3. Permanent einbinden über /etc/fstab

Damit die Partitionen nach einem Neustart automatisch an ihrem Ort verfügbar sind, müssen sie in der Datei `/etc/fstab` registriert werden. Dazu kann die eindeutige UUID verwendet werden.

```
sudo blkid /dev/sdb1                    # UUID ermitteln (z.B. UUID="xxxx-xxxx-...")
sudo mkdir -p /mnt/daten                # Einhängepunkt (Mountpoint) erstellen
sudo nano /etc/fstab                    # Eintrag hinzufügen:
```

Fügen Sie am Ende der Datei `/etc/fstab` folgende Zeile ein (Beispiel für XFS mit Standard-Optionen):

```
UUID=hier-kommt-die-uuid-hin  /mnt/daten  xfs  defaults  0  2
```

Spalte in fstab	Bedeutung
`UUID=...`	Eindeutige ID (stabiler als `/dev/sdb1`, falls sich Gerätenamen ändern)
`/mnt/daten`	Zielverzeichnis im Dateisystem (Mountpoint)
`xfs / ext4`	Verwendetes Dateisystem
`defaults`	Standard-Einhängeoptionen (rw, suid, dev, exec, auto, nouser, async)
`0  2`	Dump-Flag (`0` = kein Backup) und Dateisystem-Prüfung beim Booten (`2` = nach Root prüfen)

Achtung: Führen Sie nach dem Bearbeiten der `fstab` unbedingt einen Testlauf durch, um Syntaxfehler abzufangen, die zu einem Boot-Fehler führen könnten!

```
sudo mount -a                           # Testet alle Einträge in /etc/fstab (kein Neustart nö
    tig)
df -h                                   # Prüfen, ob das Dateisystem erfolgreich eingehängt
    wurde
```

Zusatzmodul – LVM und RAID (optional)

Merke: Dieses Modul bearbeiten Sie nur, wenn nach den Modulen 1 bis 7 Zeit bleibt. Die übrigen Module setzen es nicht voraus. Es braucht drei leere Platten (je 1 GB) oder drei Loop-Geräte.

Achtung: Prüfen Sie vor jedem Befehl mit `lsblk`, welche Platte gemeint ist. `/dev/sdb`, `/dev/sdc`, `/dev/sdd` sind Platzhalter. Verwenden Sie nie die Systemplatte.

RAID-Level im Vergleich (Platten zu je 1 TB)

Level	Mind. Platten	Nutzbar bei 4 Platten	Ausfälle verkraftet
RAID 0	2	4 TB	keine (Geschwindigkeit, keine Sicherheit)
RAID 1	2	1 TB	alle bis auf eine Platte (Spiegel)
RAID 5	3	3 TB	1 Platte
RAID 6	4	2 TB	2 Platten
RAID 10	4	2 TB	1 Platte je Spiegelpaar

Merke: RAID ist *kein* Backup. Versehentliches Löschen, Ransomware und Dateisystemfehler werden auf alle Platten gespiegelt.

Ohne Zusatzplatten: Loop-Geräte

```
for i in 1 2 3; do
  sudo truncate -s 1G /var/tmp/disk$i.img
  sudo losetup -f --show /var/tmp/disk$i.img    # nennt /dev/loopN
done
```

RAID 1 aufbauen, Ausfall behandeln

```
sudo apt install mdadm lvm2
sudo mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sdb /dev/sdc
cat /proc/mdstat                            # [UU] = beide Platten aktiv
sudo mdadm --detail /dev/md0
sudo mkfs.ext4 /dev/md0 && sudo mkdir /mnt/raid && sudo mount /dev/md0 /mnt/raid

sudo mdadm /dev/md0 --fail   /dev/sdb     # Ausfall simulieren: [U_] = degraded
sudo mdadm /dev/md0 --remove /dev/sdb
sudo mdadm /dev/md0 --add    /dev/sdd     # Wiederaufbau: watch cat /proc/mdstat

sudo mdadm --detail --scan | sudo tee -a /etc/mdadm/mdadm.conf
sudo update-initramfs -u
```

Die Daten bleiben im degradierten RAID 1 lesbar. Die beiden letzten Zeilen sorgen dafür, dass das RAID nach dem Neustart unter dem Namen `/dev/md0` wieder gefunden wird.

LVM auf dem RAID

Achtung: Vor `pvcreate` das Dateisystem aushängen: `sudo umount /mnt/raid`. Die Testdaten gehen verloren.

```
sudo pvcreate /dev/md0                       # Physical Volume
sudo vgcreate vg_data /dev/md0               # Volume Group
sudo lvcreate -L 500M -n lv_web vg_data      # Logical Volume
sudo mkfs.ext4 /dev/vg_data/lv_web
sudo mkdir -p /mnt/lvm
sudo mount /dev/vg_data/lv_web /mnt/lvm

sudo lvextend -r -L +300M /dev/vg_data/lv_web   # -r vergroessert das Dateisystem mit
sudo blkid /dev/vg_data/lv_web                  # UUID fuer /etc/fstab
sudo mount -a                                   # fstab testen, nicht neu starten
```

Befehl	**Zweck**
`pvs`, `vgs`, `lvs`, `lsblk`	Zustand der drei LVM-Ebenen und der Blockgeräte
`vgextend vg_data /dev/neu`	Volume Group erweitern, wenn sie voll ist (vorher `pvcreate`)
`df -h`	Größe des Dateisystems; wächst nur mit `lvextend -r`

Merke: Nach der Vergrößerung ist `lv_web` 800 MiB groß. Ohne `-r` wächst nur das Volume, das Dateisystem müsste mit `resize2fs` nachziehen.

Zusatzmodul – Firewall-Grundlagen (UFW und firewalld)

Merke: Je nach Linux-Distribution kommt ein anderes Standard-Werkzeug für die Paketfilterung zum Einsatz. Während Ubuntu und Debian meist auf die einfache **UFW** setzen, verwenden RedHat- und openSUSE-basierte Systeme standardmäßig **firewalld**.

Achtung: Änderungen an der Firewall können bei Remoteverbindungen (SSH) zum sofortigen Ausschluss führen. Aktivieren Sie den SSH-Zugriff immer vor dem Schalten des allgemeinen Status.

1. UFW (Uncomplicated Firewall) unter Ubuntu / Debian

UFW ist einsteigerfreundlich und setzt auf die darunterliegende **nftables**- bzw. **iptables**-Struktur auf.

```
sudo apt install ufw                    # UFW installieren (falls nicht vorhanden)
sudo ufw default deny incoming          # Standard: Alle eingehenden Verbindungen blockieren
sudo ufw default allow outgoing         # Standard: Alle ausgehenden Verbindungen erlauben

sudo ufw allow ssh                      # SSH-Zugriff erlauben (Port 22)
sudo ufw allow 80/tcp                   # HTTP erlauben (Webserver)
sudo ufw allow from 192.168.1.100       # Zugriff nur von einer bestimmten IP erlauben

sudo ufw enable                         # Firewall einschalten und beim Booten aktivieren
sudo ufw status verbose                 # Aktuellen Status und Regeln detailliert anzeigen
```

UFW-Befehl	Beschreibung
`sudo ufw status numbered`	Zeigt alle Regeln mit fortlaufender Nummer an
`sudo ufw delete <Nummer>`	Löscht eine Regel anhand ihrer Nummer
`sudo ufw disable`	Schaltet die UFW-Firewall komplett ab

2. firewalld (`firewall-cmd`) unter openSUSE / RHEL

firewalld arbeitet mit sogenannten Zonen (z. B. `public`, `internal`, `trusted`), die bestimmten Netzwerkschnittstellen zugeordnet werden.

```
sudo firewall-cmd --state                   # Prüfen, ob firewalld aktiv ist
sudo firewall-cmd --get-active-zones        # Aktive Zonen und zugewiesene Interfaces
    anzeigen

# Regeln permanent (dauerhaft) hinzufügen und laden:
sudo firewall-cmd --permanent --add-service=ssh
sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-port=8080/tcp

sudo firewall-cmd --reload                  # Lädt die Firewall-Regeln neu (ohne
    Verbindungsabbruch)
sudo firewall-cmd --list-all                # Alle aktiven Regeln der Standardzone anzeigen
```

firewall-cmd Aktion	**Zweck / Syntax**
Einmalig testen (ohne `--permanent`)	`sudo firewall-cmd --add-service=ftp`
Regel wieder entfernen	`sudo firewall-cmd --permanent --remove-service=ftp`

Merke: Wichtig bei `firewall-cmd`: Ohne das Flag `--permanent` gilt die Regel nur bis zum nächsten Neustart oder `--reload`. Nach dem Hinzufügen mit `--permanent` ist immer ein `sudo firewall-cmd --reload` erforderlich.

Übersicht – Die wichtigsten Linux-Kommandos

Merke: Diese Übersicht fasst die essenziellen Befehle für die tägliche Arbeit auf der Konsole zusammen. Die meisten Befehle unterstützen den Parameter `--help` oder besitzen Handbuchseiten (`man`).

```
=== Dateisystem und Navigation ===
pwd                             # Aktuelles Verzeichnis (Print Working Directory) anzeigen
ls -la                          # Alle Dateien (inkl. versteckter) detailliert auflisten
cd /pfad/zum/ziel               # In ein anderes Verzeichnis wechseln
cp -r quelle ziel               # Dateien oder Ordner rekursiv kopieren
mv alt.txt neu.txt              # Dateien verschieben oder umbenennen
rm -r ordner                    # Verzeichnis samt Inhalt löschen (Vorsicht!)

=== Systeminfo und Prozesse ===
\begin{lstlisting}
lsblk                           # Alle Festplatten und Partitionen als Baum anzeigen
df -h                           # Speicherplatz der Dateisysteme human-readable anzeigen
free -h                         # Arbeitsspeicher-Auslastung prüfen
top / htop                      # Laufende Prozesse und Ressourcennutzung anzeigen
systemctl status dienst         # Status eines Systemd-Dienstes abfragen

=== Netzwerk und Verbindung ===
ip a                            # IP-Adressen und Netzwerkschnittstellen anzeigen
ip r                            # Vorhandene Routingtable anzeigen
ss -tulpn                       # Aktive Ports und lauschende Dienste auflisten
ping -c 4 8.8.8.8               # Verbindung testen (4 Pakete senden)
curl -I https://example.com     # HTTP-Header einer Webadresse abrufen

=== Suchen, Filtern und Text ===
find /pfad -name "datei.txt"    # Gezielte Suche nach Dateien im Dateisystem
grep -rn "suchbegriff" /pfad/   # Text rekursiv in Dateien suchen
tail -n 50 /var/log/syslog      # Die letzten 50 Zeilen einer Logdatei ausgeben

=== Hilfe ===
man KommandoName                # Wenn Tool Installiert dann wird eine Info Seite angezeigt
apropos SuchString              # Sucht nach dem String
ls -h                           # Zeigt Parameter an
```

Tastenkürzel	Funktion auf der Konsole
`Tab`	Automatische Vervollständigung von Befehlen und Pfaden
`Strg + C`	Laufenden Befehl sofort abbrechen
`Strg + R`	In der Befehlshistorie rückwärts suchen
`Pfeil hoch / runter`	Vorherige Befehle durchblättern

Achtung: Vorsicht bei der Verwendung von Administratorrechten (`sudo`). Befehle wie `rm -rf /` können das gesamte System unwiderruflich zerstören.

vi und vim

Der Editor `vi` (meist als `vim` installiert) ist auf *jedem* Unix- und Linux-System vorhanden, auch im Rettungsmodus. Er wirkt zunächst ungewohnt, weil er **Modi** kennt:

Modus	Zweck
Normalmodus	Beim Start aktiv. Tasten sind *Befehle* (bewegen, löschen, kopieren). Rückkehr jederzeit mit `Esc`.
Einfügemodus	Text wird eingegeben. Aktivierung mit `i`, `a`, `o` u. a.
Kommandozeilenmodus	Beginnt mit `:` im Normalmodus. Speichern, Beenden, Suchen/Ersetzen.

Überlebens-Set

Datei öffnen: `vi datei`. Zum Schreiben `i` drücken. Mit `Esc` zurück in den Normalmodus. Dann `:wq` + `Enter` zum Speichern und Beenden oder `:q!` zum Beenden *ohne* Speichern.

Wichtige Befehle im Normalmodus

```
=== VIM BEWEGUNGEN & NAVIGATION ===
h, j, k, l          Links, unten, oben, rechts (oder Pfeiltasten)
w / b               Wortweise vor / zurück
0 / $               Zeilenanfang / Zeilenende
gg / G              Dateianfang / Dateiende (z.B. 25G = Zeile 25)

=== BEARBEITUNGS- & EINFÜGEMODUS ===
i / a               Einfügen vor / nach dem Cursor
I / A               Einfügen am Zeilenanfang / -ende
o / O               Neue Zeile unter / über aktueller Zeile öffnen

=== LÖSCHEN, KOPIEREN & ÄNDERN ===
x                   Zeichen löschen
dd / dw             Zeile / Wort löschen (zwischengespeichert)
yy / p              Zeile kopieren / einfügen (P = davor)
3dd / 5yy           Zahl vor Befehl wiederholt diesen (z.B. 3 Zeilen löschen)
u / Strg+R          Letzte Änderung rückgängig / wiederherstellen
.                   Letzte Änderung wiederholen

=== SUCHEN ===
/muster, n, N       Vorwärts suchen, nächster / vorheriger Treffer

=== KOMMANDOZEILENMODUS (ex-Befehle) ===
:w                  Speichern
:w neu.txt          Unter neuem Namen speichern
:q                  Beenden (nur ohne Änderungen)
:q!                 Beenden und Änderungen verwerfen
:wq  (oder ZZ)      Speichern und beenden
:set number         Zeilennummern anzeigen
:%s/alt/neu/g       In der ganzen Datei 'alt' durch 'neu' ersetzen
:%s/alt/neu/gc      Dto., aber mit Rückfrage vor Ersetzung
:10,20d             Zeilen 10 bis 20 löschen
:!ls                Externes Shell-Kommando ausführen
:r datei            Inhalt einer anderen Datei einlesen
```

Tipps für den Alltag

- `sudo update-alternatives --config editor`
- `vimtutor` startet eine interaktive Einführung von etwa 30 Minuten.
- `v` (zeichenweise), `V` (zeilenweise) und `Strg+V` (blockweise) starten den **visuellen Modus** zum Markieren; danach `d`, `y` oder `>` anwenden.
- `ZZ` und `:x` speichern nur bei Änderungen und beenden.
- Mit `:e!` lädt man die zuletzt gespeicherte Fassung neu.
- Dauerhafte Einstellungen stehen in `~/.vimrc`, zum Beispiel `set number`, `syntax on`, `set expandtab`.

Meine Notizen