

# Grundlagen der Linux-Shell-Navigation und Systembefehle (Basic Linux Commands)

Diese Dokumentation erklärt grundlegende Linux-Shell-Befehle zur Navigation, Dateiverwaltung, Benutzerverwaltung und Systemüberwachung. Jeder Befehl wird einzeln beschrieben – inklusive Syntax, Funktion und Beispielen.

## Einleitung: Was ist Linux?

### Was ist Linux?

**Linux** ist ein freies und quelloffenes Betriebssystem, das auf dem Unix-Prinzip basiert. Es besteht aus mehreren Komponenten:

- **Kernel:** Herzstück des Systems – steuert Hardwarezugriffe und Prozesse.
- **Shell:** Schnittstelle zwischen Benutzer und System (z. B. Bash).
- **GNU-Werkzeuge:** Grundlegende Systemprogramme (z. B. `ls`, `cp`, `mkdir`).
- **Dateisystem:** Alles wird als Datei betrachtet – auch Geräte und Prozesse.

Linux ist modular aufgebaut: Paketverwaltung, Netzwerktools etc. sind einzeln austauschbar und konfigurierbar, da alle Pakete aus "Textdateien" bestehen.

### Verzeichnisstruktur (Beispiele)

Verzeichnis	Funktion
<code>/</code>	Wurzelverzeichnis
<code>/bin</code>	Systembefehle für alle Benutzer
<code>/sbin</code>	Systembefehle für Administratoren
<code>/etc</code>	Konfigurationsdateien
<code>/home</code>	Benutzerverzeichnisse
<code>/var</code>	Variable Daten (Logs, Caches, etc.)
<code>/usr</code>	Benutzerprogramme, Dokumentation
<code>/tmp</code>	Temporäre Dateien
<code>/boot</code>	Bootloader & Kernel-Dateien
<code>/dev</code>	Geräte (z. B. Festplatten, USB)
<code>/proc</code> , <code>/sys</code>	Virtuelle Kernel-Infos
<code>/opt</code>	optional installierte Zusatzsoftware

## 1. Navigation im Dateisystem

### **pwd** – *Print Working Directory*

```
pwd
```

Zeigt den Pfad des aktuellen Arbeitsverzeichnisses.

---

### **cd** – *Change Directory*

```
cd [Verzeichnis]
```

Wechselt in ein anderes Verzeichnis.

Beispiele:

- **cd** **/etc** – Absoluter Pfad
  - **cd** **..** – Ein Verzeichnis höher
  - **cd** **../..** - Zwei Verzeichnisse höher
  - **cd** – Ins Home-Verzeichnis wechseln
  - **cd** **-** – Zum vorherigen Verzeichnis zurückkehren
- 

### **ls** – *List*

```
ls [Optionen] [Pfad]
```

Zeigt den Inhalt eines Verzeichnisses.

Wichtige Optionen:

- **-l** – Langformat (Rechte, Besitzer, Größe)
- **-a** – Auch versteckte Dateien anzeigen
- **-h** – Menschlich lesbare Größen
- **-R** – Rekursiv Unterordner auflisten

Beispiel:

```
ls -lah /etc
```

---

## 2. Arbeiten mit Dateien und Verzeichnissen

## **mkdir** – Verzeichnis erstellen

```
mkdir [Ordnername]
```

Legt ein neues Verzeichnis an.

Mit **-p** auch verschachtelte Ordner:

```
mkdir -p projekt/src/data
```

---

## **rm** – Dateien oder Ordner löschen

```
rm [Datei/Verzeichnis]
```

Löscht Dateien. Mit **-r** auch Verzeichnisse.

Achtung:

```
rm -rf /wichtige_daten/
```

löscht alles rekursiv und ohne Nachfrage!

---

## **cp** – Kopieren

```
cp quelle ziel
```

Kopiert eine Datei oder ein Verzeichnis.

Wichtige Optionen:

- **-r** – Verzeichnisse rekursiv
- **-i** – Rückfrage beim Überschreiben

Beispiel:

```
cp -r projekt/ backup/
```

## **mv** – Verschieben oder umbenennen

```
mv quelle ziel
```

Verschiebt oder benennt Dateien/Verzeichnisse.

Beispiel:

```
mv urlaub.jpg bilder/  
mv alt.txt neu.txt
```

---

## **nano** – Texteditor

```
nano datei.txt
```

Ein einfacher Editor direkt im Terminal.

Wichtigste Tastenkombinationen:

- **STRG+O** – Speichern
- **STRG+X** – Beenden
- **STRG+W** – Suchen

---

## Benutzerverwaltung unter Linux – Grundlagen (aus Sicht von root)

### ◇ 1. Der root-Benutzer

- root ist der Systemadministrator mit vollen Rechten.
- Root darf alles: Systemdateien ändern, Benutzer verwalten, Dienste stoppen, etc.
- In vielen Linux-Distributionen wie Ubuntu ist root zwar vorhanden, aber:
  - Das root-Passwort ist deaktiviert
  - Die direkte Anmeldung als root ist gesperrt
  - Stattdessen wird beim ersten Systemstart ein normaler Benutzer mit sudo-Rechten eingerichtet.

☒ Hinweis: Bei Ubuntu wird während der Installation automatisch ein Benutzer angelegt, der über sudo Admin-Rechte bekommt. Das root-Konto bleibt deaktiviert und kann sich nicht direkt in der Shell einloggen.

### ◇ 2. Manuelles Anlegen eines Benutzers (wenn nur root vorhanden ist)

Wenn man sich als **root** (z. B. bei einem Server oder Minimal-Image) direkt angemeldet hat und ein neuer Benutzer erstellt werden soll:

```
useradd -m -s /bin/bash benutzername  
passwd benutzername
```

- **-m**: Erstellt automatisch ein Home-Verzeichnis (/home/benutzername)
- **-s /bin/bash**: Setzt die Login-Shell auf Bash
- **passwd**: Passwort für den Benutzer setzen

Hinweis: In Debian-Systemen wird statt **useradd** der Befehl **adduser** verwendet (Proxmox ist auf debianbasis aufgebaut). Ein Blick in die entsprechenden Dokumentationen ist hier immer hilfreich 😊

### ◇ 3. Dem Benutzer sudo-Rechte geben

Damit der neue Benutzer administrative Aufgaben übernehmen darf, muss er der Gruppe sudo hinzugefügt werden (funktioniert bei Ubuntu und Debian-basierten Systemen):

```
usermod -aG sudo benutzername
```

💡 Das bedeutet: Mitglieder der Gruppe sudo dürfen über sudo administrative Befehle ausführen.

**Diese Regel steht in der Datei:**

```
/etc/sudoers
```

Dort findet sich typischerweise die Zeile:

```
%sudo    ALL=(ALL:ALL) ALL
```

Sie erlaubt allen Benutzern der Gruppe sudo, beliebige Befehle als beliebiger Benutzer auszuführen.

### ◇ 4. Bearbeitung der sudoers-Datei – nur mit **visudo**

**Was ist visudo?**

**visudo** ist ein spezielles Programm zum Bearbeiten der Datei **/etc/sudoers**, in der festgelegt wird, welche Benutzer oder Gruppen **sudo** verwenden dürfen und mit welchen Rechten.

**Warum visudo verwenden?**

Es prüft die Syntax der Datei, bevor sie gespeichert wird.

So wird verhindert, dass durch Tippfehler oder falsche Einträge die **sudo-Funktion** unbrauchbar wird.

Wenn man **/etc/sudoers** mit einem normalen Editor (**nano**, **vim**) bearbeitet, können Fehler das System so blockieren, dass keine Admin-Befehle mehr möglich sind.

**Verwendung** Die Datei **/etc/sudoers** ist sicherheitskritisch. Schon ein kleiner Fehler kann das System unbrauchbar machen.

Man verwendet daher immer:

```
visudo
```

#### ◇ 5. Was bedeutet „sudoers“?

- ☒ „sudoers“ ist ein feststehender Begriff für: die Konfigurationsdatei **/etc/sudoers**, in der festgelegt wird, wer sudo verwenden darf die Benutzer(gruppen), die darin eingetragen sind

#### ◇ **Beispiel:** Neuer Benutzer mit sudo-Rechten unter Ubuntu Server

**Als root:**

```
useradd -m -s /bin/bash alice
passwd alice
usermod -aG sudo alice
```

Jetzt kann sich alice anmelden und mit sudo arbeiten:

```
sudo apt update
```

---

## ? 7. Hilfe und Dokumentation

**man** – *Manual-Seiten*

```
man befehl
```

Zeigt die offizielle Hilfe eines Befehls.

**--help** – *Kurzinfo*

```
ls --help
```

Zeigt Optionen und Kurzbeschreibung direkt im Terminal.

---

## 8. Netzwerk- und Systemsteuerungsbefehle

scp – Secure Copy Protocol

```
scp quelle ziel
```

Kopiert Dateien sicher über SSH zwischen zwei Rechnern.

Beispiele:

```
scp datei.txt user@remote:/home/user/  
scp -r ordner/ user@192.168.1.10:/srv/data/
```

- -r: Rekursiv (für Verzeichnisse)
- Benötigt SSH-Zugriff auf das Zielsystem

ping – Netzwerkverbindung testen

```
ping [Ziel]
```

Sendet ICMP-Pakete an ein Ziel (IP oder Domain), um die Erreichbarkeit zu prüfen.

Beispiele:

```
ping 8.8.8.8          # Ping an eine IP-Adresse (Google DNS)  
ping www.google.com  # Ping an eine Domain
```

 Warum **ping** auf **192.168.137.1** (lokaler Router / Gateway)?


Test, ob die Verbindung zum lokalen Netzwerk funktioniert.

Wenn dieser **Ping** fehlschlägt, liegt das Problem im eigenen Rechner oder LAN/WLAN.

 Warum **ping** auf **8.8.8.8** (Google DNS)?

Test, ob eine Verbindung ins Internet besteht.

Funktioniert dieser **Ping**, ist das LAN in Ordnung und die Internetverbindung aktiv – unabhängig von DNS.

 Warum **ping** auf heise.de (Domain)?

Test, ob die Namensauflösung (DNS) funktioniert.

Wenn IP-Ping klappt, aber Domain-Ping nicht, liegt das Problem beim DNS-Resolver.

☞ Kurz gesagt:

- Lokal (Gateway) → Funktioniert mein Netzkabel/WLAN?
- Extern per IP → Habe ich Internetzugang?
- Extern per Domain → Funktioniert DNS?

Beenden mit STRG+C.

shutdown – System herunterfahren oder neu starten

```
shutdown [Option] [Zeit] [Nachricht]
```

Beispiele:

```
shutdown now          # Sofort herunterfahren
shutdown -h now       # Herunterfahren und anhalten (halt)
shutdown -r now       # Neustart (reboot)
shutdown -P now       # Herunterfahren und ausschalten (poweroff)
shutdown +10 "Wartung" # In 10 Minuten mit Nachricht herunterfahren
```

Wichtige Optionen:

Option	Bedeutung
-h	Herunterfahren (halt)
-P	Ausschalten (poweroff)
-r	Neustart
+min	Verzögerung in Minuten
now	Sofort
cancel	Geplantes Herunterfahren abbrechen

Abbrechen eines geplanten Shutdowns:

```
shutdown -c
```

reboot – System neu starten

```
reboot
```



---

Startet das System sofort neu. Entspricht intern:

```
shutdown -r now
```

poweroff – System ausschalten

```
poweroff
```

Führt das System vollständig herunter und schaltet es aus (wenn die Hardware das unterstützt). Entspricht:

```
shutdown -P now
```

---

## 9. Zeit und Synchronisation unter Linux

**timedatectl** – Systemzeit und Zeitzonen verwalten


**timedatectl** ist ein Kommandozeilen-Tool, um Zeit- und Datumseinstellungen abzufragen und zu ändern.

Beispiele:

```
timedatectl status
```

 Zeigt aktuelle Uhrzeit, Zeitzone und ob NTP aktiviert ist.

```
timedatectl set-timezone Europe/Berlin
```

 Ändert die Zeitzone.

```
timedatectl set-time "2025-08-18 14:30:00"
```

 Setzt manuell Uhrzeit und Datum.

**timesyncd** – Zeit über NTP synchronisieren

**systemd-timesyncd** ist ein einfacher NTP-Client, der in modernen Linux-Systemen (z. B. Ubuntu, Debian) standardmäßig enthalten ist.

Er sorgt dafür, dass die Systemuhr automatisch mit Internet-Zeitservern (NTP-Servern) synchronisiert wird.

Aktivierung prüfen:

```
timedatectl timesync-status
```

oder:

```
systemctl status systemd-timesyncd
```

🔗 Zusammenhang:

**timedatectl** = Werkzeug, um Zeit/Datum/Zeitzone einzustellen und NTP zu steuern.

**timesyncd** = Hintergrunddienst, der die tatsächliche automatische Zeitsynchronisation übernimmt.

## △ 10. Weiterführendes

Wenn du die Grundlagen beherrschst, kannst du dich mit folgenden Themen weiterbilden:

- Bash-Skripting
- Paketverwaltung (**apt**, **dnf**, **pacman**)
- Netzwerkanalyse (**curl**, **netstat**, **ss**)
- Systemd & Dienste (**systemctl**)
- Logdateien (**journalctl**, **/var/log**)

---

## Lizenz

Dieses Werk ist lizenziert unter der **Creative Commons - Namensnennung - Weitergabe unter gleichen Bedingungen 4.0 International Lizenz**.

[Zum Lizenztext auf der Creative Commons Webseite](#)