ADMIN:DOCS

Arch Linux: Installation und Grundkonfiguration

Vollständige Schritt-für-Schritt-Anleitung zur manuellen Installation von Arch Linux auf modernen UEFI/GPT-Systemen: Partitionierung, Dateisysteme, Pacstrap, Chroot und Bootloader.

Kategorie: Arch Linux Serie **Veröffentlicht:** 17.11.2024 **Zuletzt geprüft:** 27.08.2026

Arch Linux unterscheidet sich grundlegend von klassischen Einsteiger-Distributionen wie Ubuntu, Debian oder Fedora. Während andere Systeme einen grafischen Installer bereitstellen, der Partitionierung und Software-Auswahl automatisiert, setzt Arch auf das Do-It-Yourself-Prinzip (*DIY*): Du installierst und konfigurierst jede Komponente des Betriebssystems manuell über die Befehlszeile.

Die manuelle Installation ist der beste Weg, Linux wirklich zu verstehen: Du bestimmst das Partitionsschema, das Dateisystem, den Kernel, den Bootloader und das Netzwerk-Setup selbst. Das Ergebnis ist ein extrem schlankes, maßgeschneidertes System ohne unnötigen Software-Ballast.

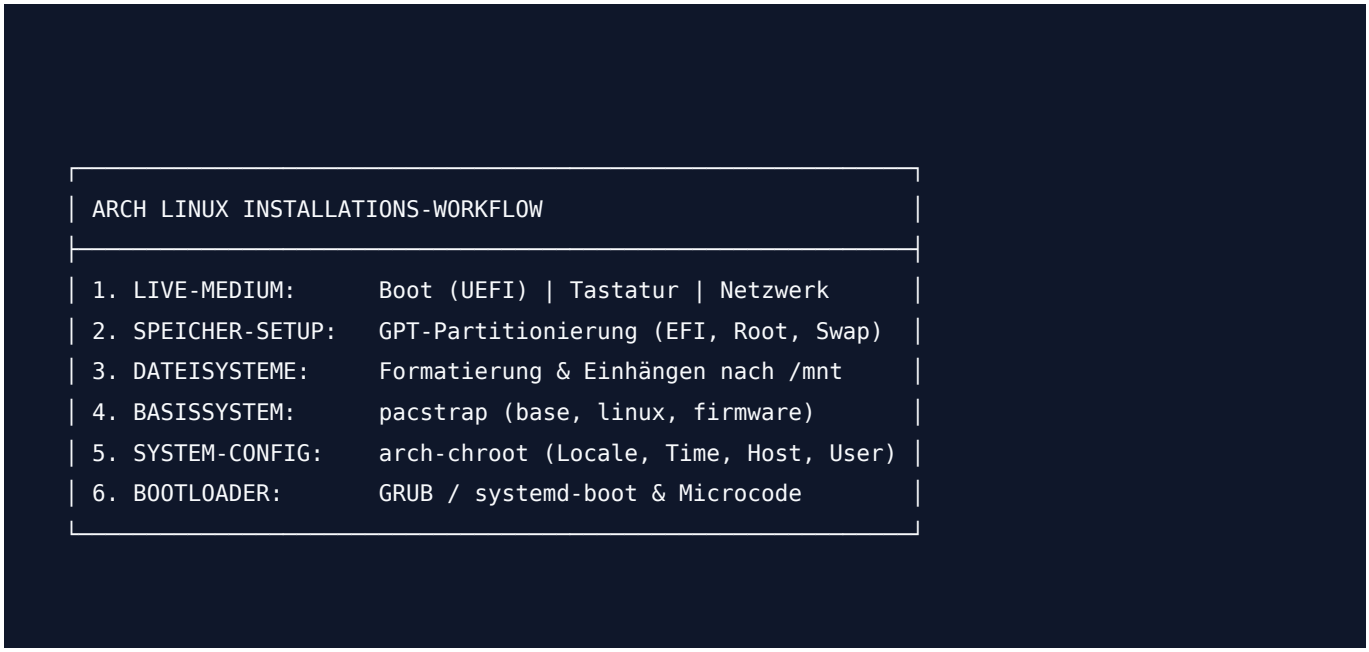
Dieser Leitfaden ist der grundlegende Einstieg in unsere Arch-Linux-Serie. Wir führen dich Schritt für Schritt und mit allen technischen Details durch den gesamten Installationsprozess für ein modernes System mit **UEFI und GPT-Partitionierung** – von der Vorbereitung des Bootmediums über `pacstrap` und `arch-chroot` bis hin zum ersten erfolgreichen Bootloader-Start.

> **Voraussetzungen:** Ein 64-Bit-fähiger Computer (x86_64) mit UEFI-Unterstützung, ein USB-Stick mit mindestens 2 GB Speicherplatz, eine stabile Internetverbindung (Ethernet oder WLAN) sowie ein grundlegendes Verständnis der Terminal-Navigation.

Quelle: https://admindocs.de/de/arch-linux-serie/archlinux-installation-und-grundkonfiguration

Der Installations-Workflow im Überblick

Der Installationsprozess von Arch Linux gliedert sich in sechs logische Phasen:



1. Vorbereitung & Bootmedium erstellen

Bevor die Installation beginnt, benötigst du das offizielle Installations-Image (*Archiso*).

ISO-Download und Signaturprüfung

Lade das aktuelle ISO-Image sowie die zugehörige Signaturdatei von der offiziellen Download-Seite herunter:

- **Download-Portal:** archlinux.org/download

Verifiziere die Integrität des Downloads auf deinem bestehenden Linux-System:

```
# SHA256-Prüfsumme verifizieren
sha256sum archlinux-*.iso
```

Vergleiche die ausgegebene Prüfsumme mit dem Hash-Wert auf der offiziellen Downloadseite.

Bootfähigen USB-Stick erstellen

Unter Linux schreibst du das Image mit dem Befehl `dd` direkt auf den USB-Stick.

Identifiziere den Gerätenamen deines USB-Sticks mit `lsblk`:

```
lsblk
```

> ⚠ **Achtung bei der Laufwerksauswahl:** Der Befehl `dd` überschreibt das angegebene Ziellaufwerk unwiderruflich! Vergewissere dich absolut sicher, dass `/dev/sdX` dein USB-Stick ist und nicht deine interne Systemfestplatte (`nvme0n1` oder `sda`).

Schreibe das Image auf den USB-Stick:

```
sudo dd bs=4M if=archlinux-*.iso of=/dev/sdX status=progress oflag=sync
```

UEFI- und BIOS-Einstellungen anpassen

Starte deinen Computer neu und öffne das UEFI/BIOS-Setup (meist über die Tasten `F2`, `F12`, `Entf` oder `Esc` beim Einschalten):

1. **Secure Boot deaktivieren:** Das Standard-Archiso unterstützt kein proprietäres Secure Boot ohne manuelle Schlüssel-Integration.

2. **SATA-Controller-Modus prüfen:** Stelle sicher, dass der Controller auf **AHCI** (nicht RAID/Optane) steht.
3. **Boot-Reihenfolge anpassen:** Setze das UEFI-USB-Laufwerk an die erste Stelle der Boot-Priorität.

2. Das Live-System starten & Initialisierung

Nach dem Start vom USB-Stick begrüßt dich das Arch-Linux-Bootmenü. Wähle den ersten Eintrag: `Arch Linux install medium (x86_64, UEFI)`.

Sobald der Bootvorgang abgeschlossen ist, landest du auf der Zsh-Kommandozeile als `root`.

Deutsches Tastaturlayout aktivieren

Das Standard-Layout im Live-System ist US-amerikanisch (`QWERTY`). Lade das deutsche Tastaturlayout (`QWERTZ`):

```
loadkeys de-latin1
```

Für eine schärfere, hochauflösende Konsolenschrift (besonders auf HiDPI-Monitoren):

```
setfont ter-128b
```

UEFI-Modus verifizieren

Stelle sicher, dass das System tatsächlich im modernen UEFI-Modus gebootet hat:

```
ls /sys/firmware/efi/efivars
```

Gibt das Verzeichnis zahlreiche EFI-Variablen aus, läuft das System im UEFI-Modus. Erscheint die Meldung `No such file or directory`, wurde das System im veralteten BIOS/CSM-Modus gestartet – prüfe in diesem Fall deine BIOS-Einstellungen.

Internetverbindung herstellen

Arch Linux benötigt während der Installation eine aktive Internetverbindung, da alle Pakete frisch aus den Online-Repositories geladen werden.

Option A – Kabelgebundene Verbindung (Ethernet): Wird meist automatisch über DHCP konfiguriert.

Option B – Drahtlose Verbindung (WLAN) mit `iwctl`: Für WLAN-Verbindungen liefert das Live-System das interaktive Werkzeug `iwd` (`iwctl`) mit:

```
# iwctl starten
iwctl
```

Innerhalb der interaktiven `[iwd]#`-Konsole:

```
# 1. WLAN-Gerätenamen ermitteln (meist wlan0)
device list
```

```
# 2. Netzwerke in Reichweite scannen
station wlan0 scan

# 3. Gefundene Netzwerke anzeigen
station wlan0 get-networks

# 4. Mit dem gewünschten WLAN verbinden (Passphrase eingeben)
station wlan0 connect "MeinWLANName"

# 5. iwctl verlassen
exit
```

Verbindungstest durchführen:

```
ping -c 3 archlinux.org
```

Systemuhr synchronisieren (NTP)

Eine exakte Systemzeit ist unerlässlich für kryptografische Zertifikatsprüfungen bei Paket-Downloads:

```
timedatectl set-ntp true
```

3. Festplattenpartitionierung & Dateisysteme (UEFI / GPT)

In diesem Schritt richten wir die Partitionstabelle auf der Ziel-Festplatte ein.

Ziel-Laufwerk identifizieren

Lasse dir alle vorhandenen Block-Devices und Partitionen anzeigen:

```
lsblk
```

Beispielausgabe einer modernen NVMe-SSD:

```
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
nvme0n1     259:0    0 500.0G  0 disk
loop0         7:0    0   1.1G  1 loop /run/archiso/bootmnt
```

In diesem Leitfaden verwenden wir `/dev/nvme0n1` als Ziellaufwerk. (Auf SATA-Systemen lautet der Name typischerweise `/dev/sda`).

Partitionierungsschema (GPT)

Wir erstellen ein standardisiertes, zukunftssicheres Partitionsschema:

Partition	Gerätepfad	Typ	Empfohlene Größe	Dateisystem	Zweck
1	`/dev/nvme0n1p1`	EFI System (`EF00`)	`1024 MB` (1 GB)	`FAT32`	Bootloader, Kernel & Initramfs
2	`/dev/nvme0n1p2`	Linux Swap (`8200`)	`8192 MB` (8 GB)	`Swap`	Auslagerungsspeicher & Zswap
3	`/dev/nvme0n1p3`	Linux Root x86-64 (`8304`)	Restlicher Speicher	`ext4`	Betriebssystem, Home & Daten

Partitionen anlegen mit `cfdisk`

`cfdisk` bietet eine übersichtliche, menügeführte Textoberfläche:

```
cfdisk /dev/nvme0n1
```

1. Wähle als Label-Typ **`gpt`** (sofern die Festplatte noch keine Partitionstabelle besitzt).
2. **EFI-Partition anlegen:**

- Wähle *New* → Größe: `1G` → Wähle *Type* → **`EFI System`**.

1. **Swap-Partition anlegen:**

- Wähle den freien Speicher (*Free space*) → *New* → Größe: `8G` → Wähle *Type* → **`Linux swap`**.

1. **Root-Partition anlegen:**

- Wähle den restlichen freien Speicher → *New* → Gesamte Restgröße bestätigen → Typ bleibt **`Linux root (x86-64)`**.

1. Wähle **`Write`**, tippe `yes` zum Bestätigen und verlasse das Programm über **`Quit`**.

Dateisysteme formatieren

Formatiere die neu erstellten Partitionen mit ihren jeweiligen Dateisystemen:

```
# 1. EFI System Partition (zwingend FAT32)
mkfs.fat -F 32 /dev/nvme0n1p1

# 2. Swap-Bereich initialisieren und aktivieren
mkswap /dev/nvme0n1p2
```

```
swapon /dev/nvme0n1p2

# 3. Root-Dateisystem formatieren (ext4 mit Label arch_root)
mkfs.ext4 -L arch_root /dev/nvme0n1p3
```

Dateisysteme unter `/mnt` einhängen

Die Partitionen müssen nun in der korrekten Hierarchie unter dem temporären Mountpoint `/mnt` eingehängt werden:

```
# 1. Root-Partition als Basis einhängen
mount /dev/nvme0n1p3 /mnt

# 2. EFI-Boot-Verzeichnis erstellen und EFI-Partition einhängen
mount --mkdir /dev/nvme0n1p1 /mnt/boot
```

Überprüfe die Mount-Struktur:

```
lsblk /dev/nvme0n1
```

Ausgabe:

```
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
nvme0n1     259:0    0 500.0G  0 disk
```

```
├─nvme0n1p1 259:1    0     1G  0 part /mnt/boot
├─nvme0n1p2 259:2    0     8G  0 part [SWAP]
└─nvme0n1p3 259:3    0 491.0G  0 part /mnt
```

4. Basissystem installieren (`pacstrap`)

Mit dem Werkzeug `pacstrap` installieren wir die grundlegenden Systempakete direkt in das gemountete Zielverzeichnis `/mnt`.

```
pacstrap -K /mnt \
  base \
  base-devel \
  linux \
  linux-firmware \
  linux-headers \
  networkmanager \
  neovim \
  nano \
  man-db \
  man-pages \
  git \
  sudo
```

Erläuterung der Kernpakete:

- `base`: Minimales Paket-Set für ein funktionsfähiges Arch-Linux-System.
- `base-devel`: Compiler-Tools (`gcc`, `make`, etc.), unverzichtbar für spätere Software-Builds und AUR-Pakete.
- `linux` & `linux-headers`: Der offizielle Linux-Kernel und Header-Dateien für Kernel-Module.
- `linux-firmware`: Hardware-Treiber-Firmware für WLAN, Grafik, Audio und Controller.
- `networkmanager`: Leistungsfähiger Daemon zur Verwaltung von Ethernet, WLAN und VPNs.

- `sudo`: Ermöglicht unprivilegierten Benutzern administrative Befehlsausführung.

Dateisystemtabelle generieren (`fstab`)

Generiere die Datei `/etc/fstab`, damit das System beim Booten weiß, welche Partitionen über welche UUIDs eingehängt werden müssen:

```
genfstab -U /mnt >> /mnt/etc/fstab
```

Prüfe den Inhalt der generierten `fstab` auf Vollständigkeit:

```
cat /mnt/etc/fstab
```

Alle drei Partitionen (`/`, `/boot`, `swap`) müssen mit eindeutigen `UUID=`-Einträgen aufgeführt sein.

5. Systemkonfiguration in der Chroot-Umgebung

Wechsle mit `arch-chroot` in das frisch installierte System. Ab diesem Moment arbeitest du direkt auf deiner Festplatte:

```
arch-chroot /mnt
```

Zeitzone und Hardware-Uhr

Setze die Zeitzone (Beispiel für Deutschland/Berlin):

```
ln -sf /usr/share/zoneinfo/Europe/Berlin /etc/localtime
```

Synchronisiere die Hardware-Uhr deines Mainboards mit der Systemzeit im UTC-Standard:

```
hwclock --systohc
```

Lokalisierung & Spracheinstellungen

Öffne `/etc/locale.gen` und aktiviere die benötigten UTF-8-Sprachdefinitionen:

```
nano /etc/locale.gen
```

Entferne das Raute-Symbol (`#`) vor den folgenden Zeilen:

```
de_DE.UTF-8 UTF-8
en_US.UTF-8 UTF-8
```

Generiere die aktivierten Locales:

```
locale-gen
```

Setze die Standardsprache des Systems in `/etc/locale.conf`:

```
echo "LANG=de_DE.UTF-8" > /etc/locale.conf
```

Setze das deutsche Tastaturlayout dauerhaft für die virtuelle Konsole in `/etc/vconsole.conf`:

```
echo "KEYMAP=de-latin1" > /etc/vconsole.conf
```

Netzwerkkonfiguration & Hostname

Vergib einen eindeutigen Netzwerknamen für deinen Computer in `/etc/hostname`:

```
echo "archbox" > /etc/hostname
```

Erstelle die lokale Namensauflösung in `/etc/hosts`:

```
127.0.0.1   localhost
::1         localhost
127.0.1.1   archbox.localdomain archbox
```

Aktiviere den NetworkManager-Dienst für den automatischen Netzwerkstart nach dem Reboot:

```
systemctl enable NetworkManager.service
```

Root-Passwort und Benutzerverwaltung

Vergib ein sicheres Passwort für das administrative Root-Konto:

```
passwd
```

Lege einen unprivilegierten Benutzer für den täglichen Betrieb an und weise ihn der administrativen Gruppe `wheel` zu:

```
# Benutzer 'adminuser' anlegen
useradd -m -G wheel -s /bin/bash adminuser

# Passwort für den neuen Benutzer vergeben
```

```
passwd adminuser
```

Sudo-Rechte für die `wheel`-Gruppe freischalten

Öffne die Sudoers-Konfiguration über `visudo`:

```
EDITOR=nano visudo
```

Suche die folgende Zeile und entferne das führende `#`:

```
%wheel ALL=(ALL:ALL) ALL
```

Speichere die Datei mit `Ctrl+O` und schließe sie mit `Ctrl+X`.

6. Prozessor-Microcode & Bootloader-Installation

Vor der Installation des Bootloaders muss der CPU-Microcode installiert werden, um Hardware-Fehler und CPU-Sicherheitslücken beim Systemstart abzufangen:

```
# Für Intel-Prozessoren:
pacman -S intel-ucode
```

```
# Für AMD-Prozessoren:
# pacman -S amd-ucode
```

Bootloader: GRUB für UEFI installieren

Installiere GRUB und die EFI-Boot-Verwaltungstools:

```
pacman -S grub efibootmgr dosfstools mtools
```

Installiere den GRUB-Bootloader in die gemountete EFI-Systempartition (`/boot`):

```
grub-install --target=x86_64-efi --efi-directory=/boot --bootloader-id=GRUB --recheck
```

Erstelle die GRUB-Konfigurationsdatei. GRUB erkennt das installierte Microcode-Paket automatisch und bindet es in den Bootvorgang ein:

```
grub-mkconfig -o /boot/grub/grub.cfg
```

Die Ausgabe bestätigt die erfolgreiche Erkennung des Linux-Kernels und des Initramfs-Images:

```
Found linux image: /boot/vmlinuz-linux
Found initrd image: /boot/intel-ucode.img /boot/initramfs-linux.img
Found fallback initrd image(s) in /boot: initramfs-linux-fallback.img
done
```

7. Abschluss der Installation & Erster Systemstart

Die Grundkonfiguration ist vollständig abgeschlossen.

Verlasse die Chroot-Umgebung:

```
exit
```

Hänge alle gemounteten Partitionen sauber aus:

```
umount -R /mnt
```

Starte das System neu und ziehe den USB-Stick ab:

```
reboot
```

Erste Anmeldung

Nach dem Neustart erscheint das GRUB-Bootmenü. Melde dich auf der virtuellen Konsole mit deinem neu erstellten Benutzerkonto (`adminuser`) an.

Überprüfe die Internetverbindung und deine Sudo-Berechtigung:

```
# Netzwerkstatus prüfen
ping -c 3 archlinux.org

# Sudo-Funktion und Paketstatus prüfen
sudo pacman -Syu
```

Herzlichen Glückwunsch! Dein minimales, sauberes und voll funktionsfähiges Arch-Linux-Basissystem ist einsatzbereit.

Installationsphasen im Überblick

Phase	Kernbefehle / Konfigurationsdateien	Zweck
Vorbereitung	`loadkeys de-latin1` , `iwctl` , `timedatectl set-ntp true`	Tastatur, Netzwerk, Zeit
Partitionierung	`cfdisk /dev/nvme0n1` , `mkfs.fat` , `mkfs.ext4` , `mkswap`	GPT-Layout & Dateisysteme
Einhängen	`mount /dev/... /mnt` , `mount --mkdir /dev/... /mnt/boot`	Mount-Hierarchie für pacstrap

Phase	Kernbefehle / Konfigurationsdateien	Zweck
Basissystem	`pacstrap -K /mnt base linux linux-firmware base-devel`	OS-Kernkomponenten
Konfiguration	`/etc/fstab`, `/etc/locale.gen`, `/etc/hostname`, `passwd`	Lokalisierung, User, Identity
Bootloader	`intel-ucode`, `grub-install`, `grub-mkconfig`	UEFI-Systemstart

Weiterführende Ressourcen

Arch Linux: Installation der grafischen Benutzeroberfläche Pacman: Der umfassende Leitfaden Arch Linux: Best Practices und Tipps zur Systempflege Arch Linux: Systemhärtung und Sicherheit – Best Practices Offizieller Arch Linux Installation Guide (Arch Wiki)

Fazit

Die manuelle Installation von Arch Linux ist der beste Weg, um die inneren Mechanismen eines modernen Linux-Betriebssystems wirklich zu verstehen. Anstatt dich auf vorgefertigte Installer-Routinen zu verlassen, hast du jeden Baustein deines Systems – vom EFI-Systempartition-Layout über das Initramfs bis zum Bootloader – eigenhändig konfiguriert.

Du verfügst nun über ein extrem schlankes, kompromisslos stabiles Basissystem, das ohne unnötigen Hintergrund-Ballast läuft. Dieses Fundament bietet maximale Kontrolle, da du genau weißt, welche Dienste aktiv sind und wie dein System strukturiert ist.

> □□ **Praxis-Tipp:** Dokumentiere deine gewählten Mount-Optionen, UUIDs und Benutzerkonfigurationen stets sorgfältig. Bei späteren Wartungsarbeiten oder der Systemwiederherstellung über ein Live-Medium spart dir eine exakte Dokumentation wertvolle Zeit.

Im nächsten Artikel unserer Serie bauen wir auf diesem soliden Fundament auf: In Arch Linux: Installation der grafischen Benutzeroberfläche richten wir hardwarebeschleunigte Grafiktreiber, den modernen PipeWire-Audiostack und einen vollwertigen KDE Plasma 6 Desktop unter

Quelle: https://admindocs.de/de/arch-linux-serie/archlinux-installation-und-grundkonfiguration

Wayland ein.