ADMIN:DOCS

# Ubuntu-Upgrade: Von Version 24.04 LTS auf 26.04 LTS

Schritt-für-Schritt-Anleitung für das Upgrade von Ubuntu 24.04 LTS (Noble Numbat) auf 26.04 LTS (Resolute Raccoon) – inklusive cgroup v2, sudo-rs und APT 3.2 Rollback.

**Kategorie:** Server environments **Veröffentlicht:** 25.04.2026

Das Distributions-Upgrade von **Ubuntu 24.04 LTS (Noble Numbat)** auf **Ubuntu 26.04 LTS (Resolute Raccoon)** modernisiert den Linux-Server-Unterbau grundlegend: `Linux Kernel 7.0`, der vollständige Übergang auf `cgroup v2-only`, die Rust-basierte Sicherheitsarchitektur mit `sudo-rs` sowie native Rollback-Funktionen in `APT 3.2`.

Dieser Leitfaden führt dich schrittweise durch die Vorbereitung, den eigentlichen Upgrade-Vorgang, die Behebung möglicher Konflikte und die Nachkonfiguration moderner Serverfeatures.

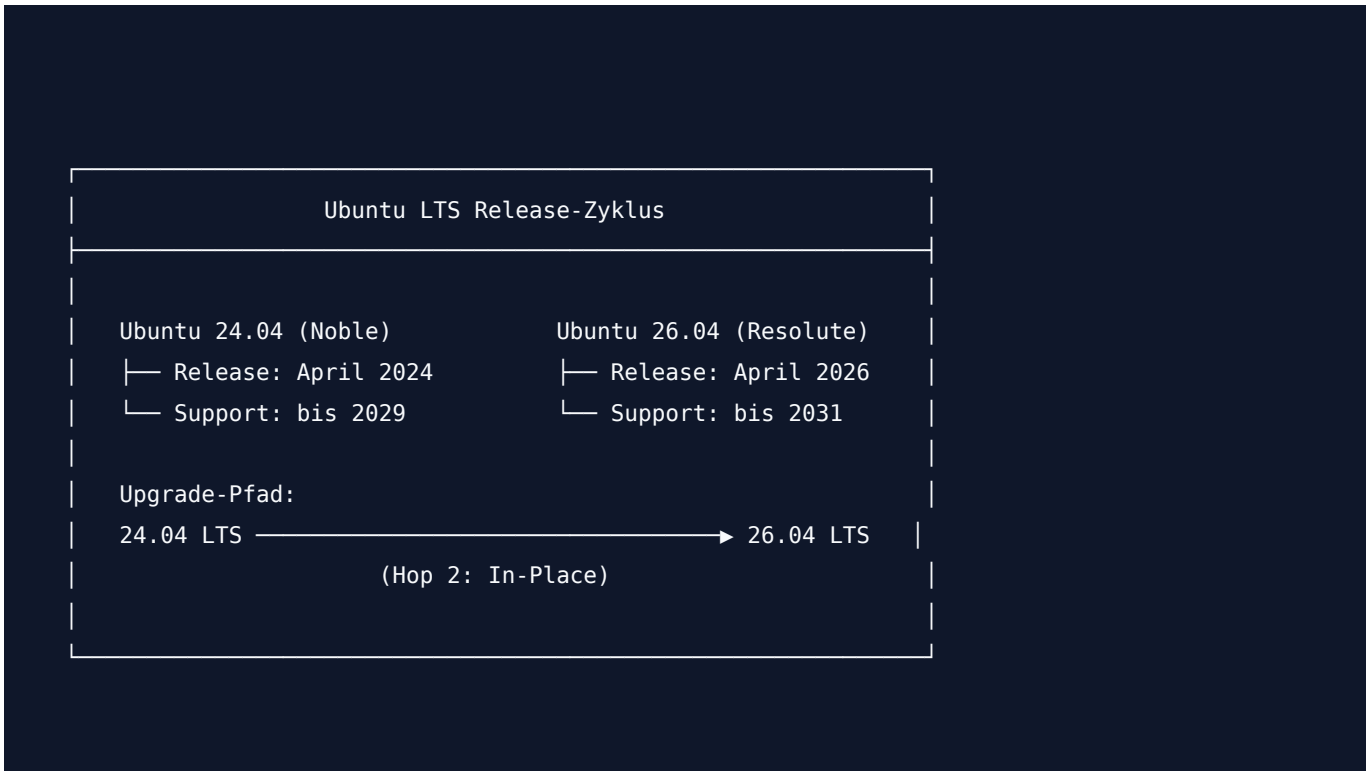
> □□ **Voraussetzung:** Ubuntu unterstützt In-Place-Upgrades immer nur zwischen aufeinanderfolgenden LTS-Releases. Falls dein System noch auf **Ubuntu 22.04 LTS** läuft, führe bitte zuerst den ersten Schritt durch: Ubuntu-Upgrade: Von Version 22.04 LTS auf 24.04 LTS.

> ⚠ **Verfügbarkeit des Upgrades:** Der automatische Upgrade-Pfad für LTS-Systeme wird von Canonical typischerweise mit dem ersten Point-Release (Ubuntu 26.04.1 LTS im Spätsommer 2026) freigegeben. Teste das Upgrade vorab in einer Staging- oder VM-Umgebung.

Quelle: https://admindocs.de/de/serverumgebungen/ubuntu-upgrade-von-version-24-04-lts-auf-26-04-lts

# Grundlagen und Neuerungen in Ubuntu 26.04 LTS

## Die wichtigsten Architekturänderungen




### Vergleich der Kernkomponenten:

| Komponente | Ubuntu 24.04 LTS (Noble) | Ubuntu 26.04 LTS (Resolute) | Bedeutung & Breaking Changes |
|---|---|---|---|
| **Linux Kernel** | 6.8 | 7.0 | Verbesserte CPU-Scheduler, neuere I/O-Treiber |
| **Control Groups** | cgroup v1 + v2 | **cgroup v2 only** | cgroup v1 komplett entfernt (Docker/Container-Check nötig) |
| **Sudo-Stack** | sudo (C) | **sudo-rs (Rust)** | Memory-safe sudo-Implementierung mit nativer sudoers-Kompatibilität |
| **Paketmanager** | APT 2.9 | **APT 3.2** | Eingebaute Transaktions-Historie & Rollback-Support |

| Komponente | Ubuntu 24.04 LTS (Noble) | Ubuntu 26.04 LTS (Resolute) | Bedeutung & Breaking Changes |
|---|---|---|---|
| **systemd** | 255 | 259 | Erweiterte Log-Retention und Service-Isolation |
| **Python** | 3.12 | 3.13 | JIT-Compiler-Optionen, strikte venv-Trennung |
| **Desktop / GUI** | GNOME 46 (X11/Wayland) | GNOME 50 (Wayland-only) | X11-Server als Session-Standard entfernt |

# Upgrade-Vorbereitung auf Ubuntu 24.04

## 1. Ausgangszustand und Hardware-Voraussetzungen

Prüfe vor dem Start den Ausgangszustand deines 24.04-Systems:

```
# Distribution und Version bestätigen
lsb_release -a

# RAM & Festplattenplatz (mindestens 25 GB freier Speicher auf /)
free -h
df -h /
```

### Vorbereitung auf cgroup v2 (Breaking Change Check):

```
# Aktuelles cgroup-Dateisystem prüfen
stat -fc %T /sys/fs/cgroup/
# Erwartete Ausgabe: cgroup2fs
```

> ⚠ **Achtung bei älteren Containern / Docker-Daemons:** Da Ubuntu 26.04 cgroup v1 nicht mehr unterstützt, müssen Container-Engines (Docker, Podman, containerd) auf Versionen laufen, die cgroup v2 vollständig beherrschen (Docker 20.10+ bzw. Docker CE 24+).

## 2. Vollständiges System-Backup erstellen

```
#!/bin/bash
# backup_vor_hop2.sh
set -euo pipefail

BACKUP_DIR="/backup/ubuntu24_to_26_$(date +%Y%m%d_%H%M%S)"
echo "=== Erstelle Backup vor Upgrade auf 26.04 in: $BACKUP_DIR ==="

sudo mkdir -p "$BACKUP_DIR"
sudo cp -a /etc "$BACKUP_DIR/etc"
sudo cp -a /home "$BACKUP_DIR/home"
dpkg --get-selections > "$BACKUP_DIR/installed_packages_2404.txt"

echo "Backup abgeschlossen:"
sudo du -sh "$BACKUP_DIR"
```

## 3. Bestehendes 24.04-System aktualisieren

```
# Alle Paketlisten synchronisieren und vollständiges Upgrade durchführen
sudo apt update
sudo apt full-upgrade -y
```

```
# System bereinigen
sudo apt autoremove --purge -y
sudo apt autoclean

# Snap-Pakete auffrischen
sudo snap refresh
```

## Der Upgrade-Prozess (24.04 → 26.04)

### Upgrade in einer geschützten Session starten

Verwende bei Remote-Verbindungen zwingend `screen` oder `tmux`, damit das Upgrade bei Netzwerkunterbrechungen nicht abbricht.

```
# Screen-Sitzung initiieren
screen -S resolute-upgrade

# Upgrade einleiten
sudo do-release-upgrade
```

```
┌──────────────────────────────────────────────────────────┐
│                Ablauf des Release-Upgrades               │
├──────────────────────────────────────────────────────────┤
│                                                          │
│  1. Analyse & Paketlisten       [███▒▒▒▒▒▒] 10%          │
│     ├── Deaktivierung inkompatibler Drittanbieter-PPAs   │
│     └── Umstellung auf 26.04 Repositories (resolute)     │
│                                                          │
│  2. Download & Dependency Tree  [█████▒▒▒▒] 50%          │
```

```
│     └── Bezug der Pakete für Kernel 7.0, systemd 259        │
│                                                             │
│  3. Installation & Migration    [████████▒▒] 80%            │
│     ├── Ersetzung von klassischem sudo durch sudo-rs        │
│     └── cgroup v2-only Konfigurationsanpassungen            │
│                                                             │
│  4. Abschluss & Reboot          [██████████] 100%           │
│     └── Neustart in Ubuntu 26.04 LTS                        │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

Sobald alle Pakete installiert und obsolete Abhängigkeiten entfernt wurden, bestätigst du den Systemneustart:

```
sudo reboot
```

## Validierung & Neue Kern-Features

Nach dem Reboot verifizierst du die erfolgreiche Installation:

```
# OS-Version und Kernel 7.0 prüfen
lsb_release -a
uname -r

# Fehlgeschlagene Systemdienste kontrollieren
systemctl --failed

# Netzwerk- und Netplan-Status prüfen
sudo netplan status
```

```
ip addr show
```

## 1. sudo-rs in der Praxis

Ubuntu 26.04 liefert standardmäßig die in Rust geschriebene Implementierung `sudo-rs` aus. Sie erhöht die Speichersicherheit bei unveränderter Syntax.

```
# Version und Status prüfen
sudo -V

# Berechtigungen prüfen (liest bestehende /etc/sudoers unverändert ein)
sudo -l
```

> □□ **Kompatibilitätshinweis:** Bestehende `/etc/sudoers` und `/etc/sudoers.d/*`-Konfigurationen bleiben 1:1 gültig. Es sind keine Änderungen an deinen Aliasen oder Rechteregeln erforderlich.

## 2. APT 3.2 Rollback-Funktion nutzen

Ein herausragendes Feature in Ubuntu 26.04 ist die native Transaktionshistorie in APT 3.2.

```
# Verlauf der letzten Paketaktionen anzeigen
sudo apt history

# Letzte fehlerhafte Installation rückgängig machen (Undo)
sudo apt history undo last
```

```
# Rollback zu einem bestimmten Zeitpunkt
sudo apt history undo 2026-08-20
```

> **Vorteil im Serverbetrieb:** Bei fehlerhaften Paketupdates entfällt das manuelle Heraussuchen und Downgraden von Einzelbibliotheken.

## 3. cgroup v2 & Docker-Optimierung

Da cgroup v1 entfallen ist, verwaltest du Ressourcen-Limits direkt über die v2-Schnittstelle von systemd:

```
# Ressourcenverbrauch live überwachen
systemd-cgtop

# Ressourcen-Limit für einen Service konfigurieren (z.B. Nginx/SSH)
sudo systemctl set-property ssh.service MemoryMax=512M CPUQuota=50%
```

Für Docker stellst du sicher, dass der cgroup2-Treiber aktiv ist ( `/etc/docker/daemon.json` ):

```
{
  "exec-opts": ["native.cgroupdriver=cgroup2"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "10m",
    "max-file": "3"
  }
}
```

```
sudo systemctl restart docker
```

# Nachkonfiguration & System-Härtung

## 1. SSH- und Firewall-Standards anpassen

```
# SSH-Konfiguration öffnen
sudo nano /etc/ssh/sshd_config

# Empfohlene Sicherheitseinstellungen:
# PermitRootLogin no
# PasswordAuthentication no
# PubkeyAuthentication yes
```

```
# UFW Firewall-Status und aktive Ports kontrollieren
sudo ufw status verbose
```

### 2. Systembereinigung & Log-Rotation

```
# Alte Kernel-Versionen und verwaiste Pakete bereinigen
sudo apt autoremove --purge -y
sudo apt autoclean

# Systemd-Journal auf maximale Größe beschränken (/etc/systemd/journald.conf)
sudo journalctl --vacuum-size=200M
```

## Problembehandlung & Notfall-Szenarien

Auch bei sorgfältiger Vorbereitung können während eines Major-Upgrades unerwartete Fehler auftreten:

### 1. Abgebrochener Upgrade-Prozess (*Inconsistent State*)

Falls das Upgrade vorzeitig stoppt:

```
# Paketverwaltung entsperren und konfigurieren
sudo dpkg --configure -a

# Fehlende Abhängigkeiten automatisch nachinstallieren
sudo apt install -f -y

# Paketstatus auditieren
sudo dpkg --audit
```

### 2. Boot-Probleme & Notfall-Shell

Startet das System nach dem Upgrade nicht in den neuen Kernel:

```
# 1. Im GRUB-Menü "Advanced Options" -> "Recovery Mode" wählen
# 2. "Drop to root shell prompt" auswählen

# Dateisystem im Schreibmodus mounten
mount -o remount,rw /

# Journal auf fatale Boot-Fehler prüfen
journalctl -xb -p err

# GRUB-Bootloader neu installieren
update-grub
```

### 3. Python 3.13 und PEP 668 (*externally-managed-environment*)

Wenn systemweite Python-Skripte nach dem Upgrade fehlschlagen:

```
# Empfohlener Weg: Eigene virtuelle Umgebung nutzen
python3 -m venv ~/my-environment
source ~/my-environment/bin/activate
pip install -r requirements.txt

# Für CLI-Tools: pipx verwenden
sudo apt install pipx
pipx install <paketname>
```

## Langfristige Wartung: Automatisierte Updates & Monitoring

Um dein Ubuntu 26.04 LTS-System über die kommenden fünf Jahre stabil und sicher zu betreiben, empfiehlt sich die Aktivierung automatischer Sicherheitsaktualisierungen:

```
# Unattended-Upgrades einrichten
sudo apt install unattended-upgrades apt-listchanges -y
sudo dpkg-reconfigure -plow unattended-upgrades
```

### Automatischer Health-Check via Cron-Job:

Erstelle ein Monitoring-Script unter `/usr/local/bin/system_health.sh`:

```
#!/bin/bash
# /usr/local/bin/system_health.sh
set -euo pipefail

LOG_FILE="/var/log/system_health.log"
DATE=$(date '+%Y-%m-%d %H:%M:%S')

echo "=== System Health Check - $DATE ===" >> "$LOG_FILE"

# Festplattenauslastung prüfen
DISK_USAGE=$(df / | awk 'NR==2 {print $5}' | tr -d '%')
if [ "$DISK_USAGE" -gt 85 ]; then
    echo "WARNUNG: Festplatte zu ${DISK_USAGE}% belegt" >> "$LOG_FILE"
fi

# Fehlgeschlagene Dienste prüfen
FAILED=$(systemctl --failed --no-legend | wc -l)
if [ "$FAILED" -gt 0 ]; then
    echo "WARNUNG: $FAILED fehlgeschlagene Systemd-Services!" >> "$LOG_FILE"
    systemctl --failed --no-legend >> "$LOG_FILE"
fi

echo "=== Status: OK ===" >> "$LOG_FILE"
```

### Script ausführbar machen:

```
sudo chmod +x /usr/local/bin/system_health.sh
```

## Fazit

Das Upgrade von Ubuntu 24.04 LTS auf 26.04 LTS bringt signifikante Sicherheits- und Architekturverbesserungen: Der moderne Linux Kernel 7.0, die speichersichere `sudo-rs`-Implementierung und native Rollback-Mechanismen in APT 3.2 heben die Plattform auf den aktuellen Enterprise-Stand.

Entscheidend für einen reibungslosen Übergang sind die sorgfältige Vorab-Prüfung von cgroup-v2-Abhängigkeiten (insbesondere bei Container-Workloads), das Durchführen des Upgrades in einer stabilen `screen`-Sitzung und die anschließende Bereinigung verwaister Kernel und Pakete.

## Weiterführende Ressourcen

- Linux Server Härtung: SSH mit FIDO2 & CrowdSec
- Teil 1: Ubuntu-Upgrade von 22.04 auf 24.04 LTS
- LPIC-1 Serie: Linux-Grundlagen
- Docker & Docker Compose auf Linux
- Ubuntu Serverumgebungen