

corewire

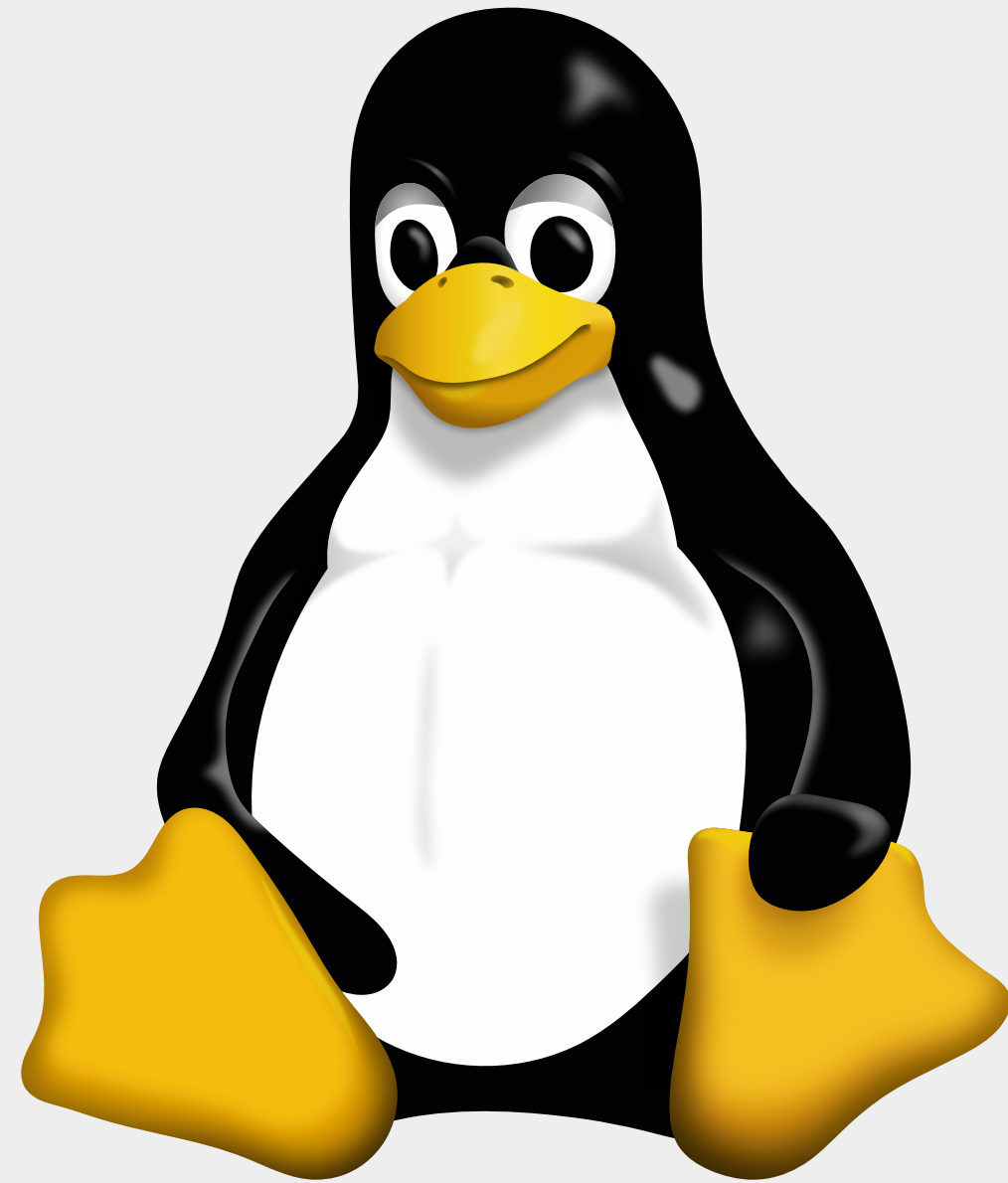
Linux

Einführung





Folien-Hinweis

- `Space`, `Page down`: Nächste Folie
- `Page up`: Vorherige Folie
- `ESC`, `o`: Übersicht

[Zur Kapitelübersicht](#)



Vorstellungsrunde

-  Name
-  Job / Aufgabenbereich
-  Vorerfahrungen
-  Erwartungen an den Kurs

Pausen

- Mittagspause: 12 - 13 Uhr
- 5-10 min Pause ca. stündlich

Agenda - Tag 1 & 2

1. Einführung in Linux
2. Navigation & Textverarbeitung
3. Shell
4. Paketmanager
5. Linux File System

Agenda - Tag 3 & 4

1. Einstieg in Docker
2. Benutzer, Gruppen und Rechte

Agenda - Tag 5 & 6

1. Arbeiten mit SSH und Remote-Zugriff
2. Logs & Fehleranalyse
3. Kubernetes-Ausblick

Linux - Motivation

- **Kostenlos** - Keine Lizenzgebühren
- **Open Source** - Frei verfügbar und anpassbar
- **Stabilität** - Sehr zuverlässig im Dauerbetrieb
- **Sicherheit** - Wenig Probleme mit Viren/Malware
- **Flexibilität** - Von Servern bis Embedded Systems
- **Community** - Große, aktive Entwicklergemeinschaft

Linux - Marktanteile

- Desktop: Ca. 6%
- Server (web): Ca. 77%
- Supercomputer: 100%
- Smartphones/Tablets: Ca. 73%

Was ist ein Betriebssystem?

- Vermittler zwischen Hardware und Software
- Verwaltung von Ressourcen
 - CPU, Speicher, I/O-Geräte
- Bereitstellung von Diensten
 - Dateisystem, Netzwerk, Prozesse
- Benutzer- und Rechteverwaltung

Was ist Linux?

- **Unix-ähnliches Betriebssystem**
- Entwickelt von Linus Torvalds (1991)
- Basiert auf dem Linux-Kernel
- Verschiedene Distributionen verfügbar

Was ist ein Kernel?

OS



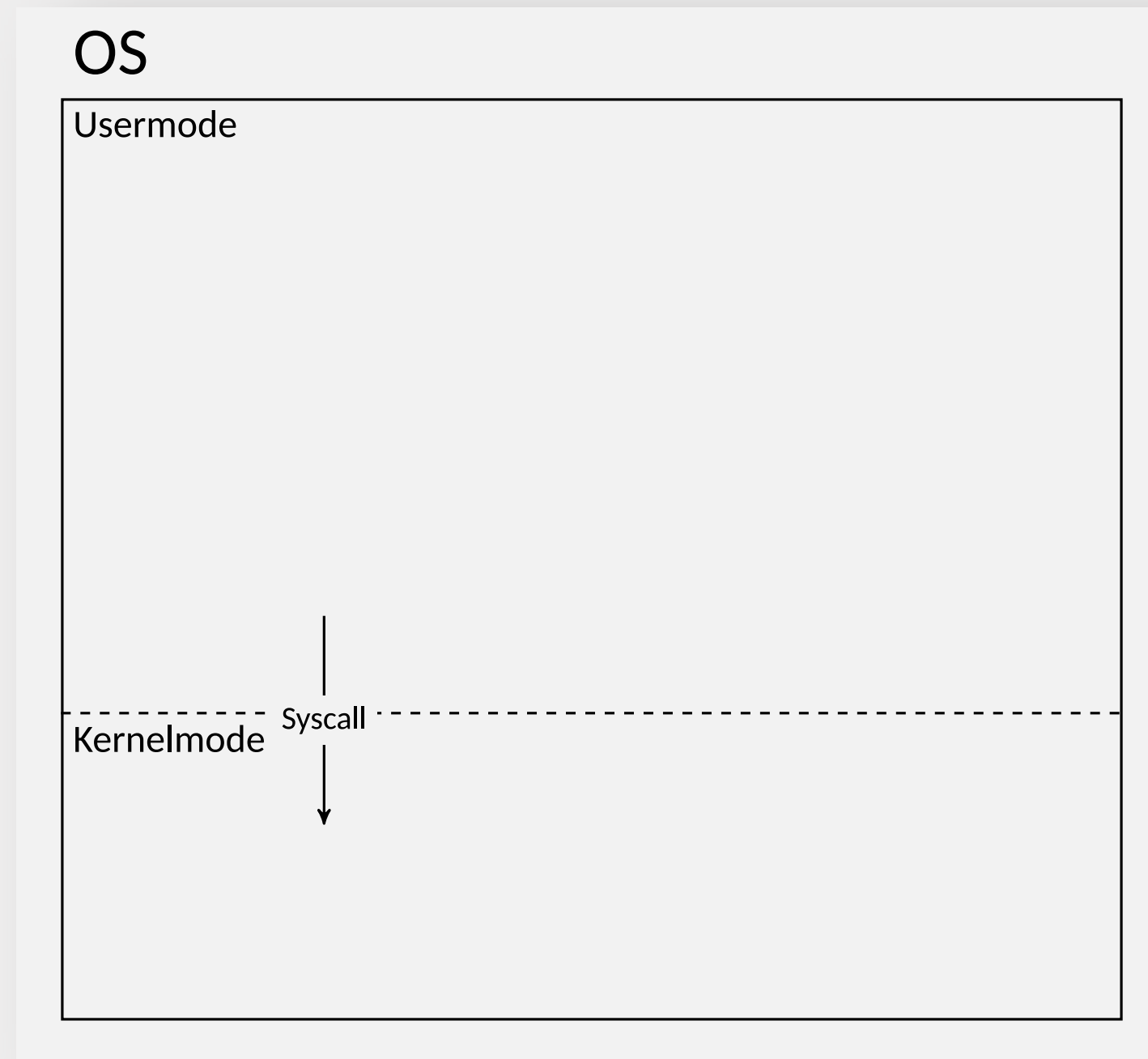
Was ist ein Kernel?

OS

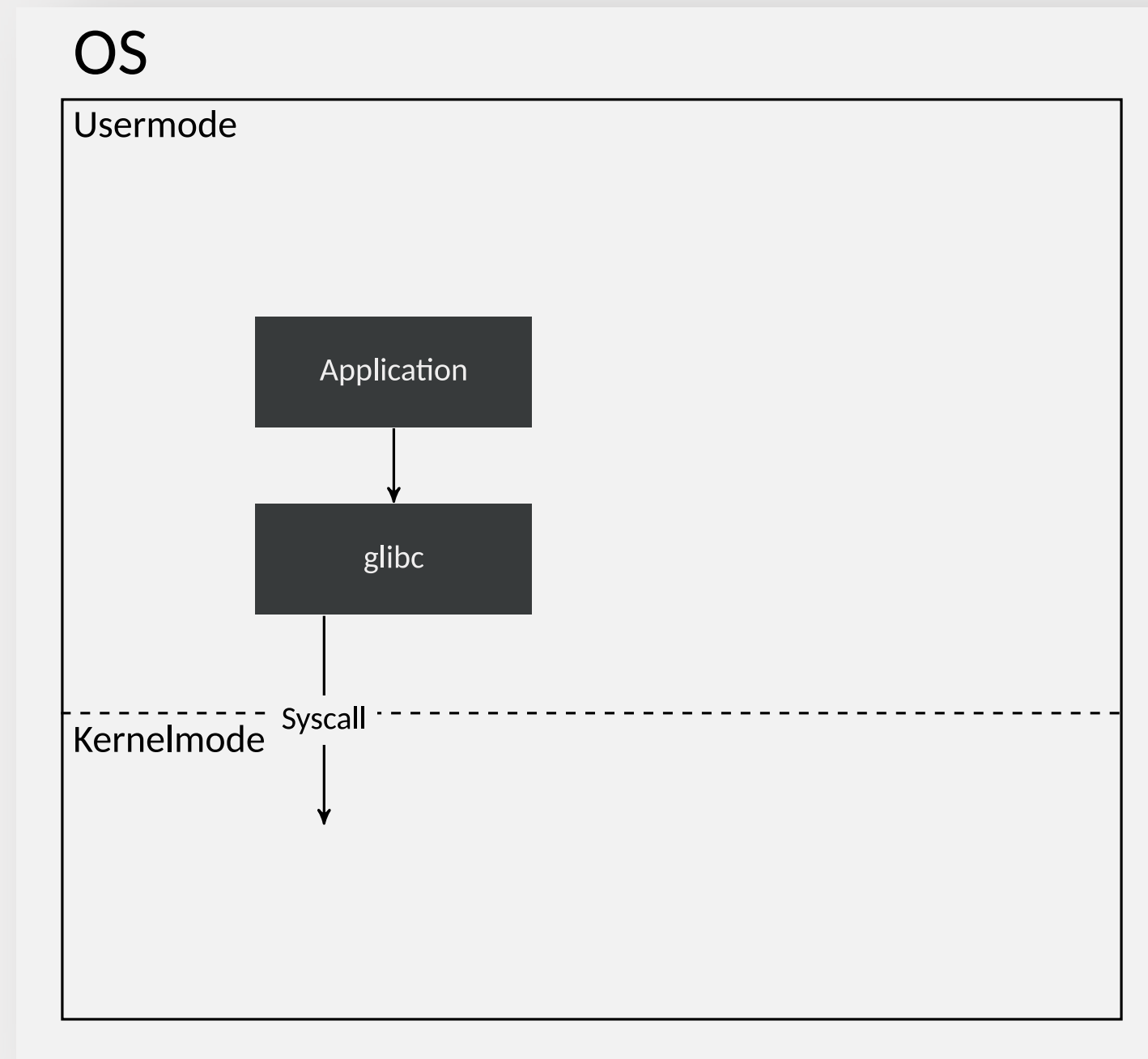
Usermode

Kernelmode

Was ist ein Kernel?



Was ist ein Kernel?



Was ist ein Kernel?

OS

Usermode

Kernelmode

Application

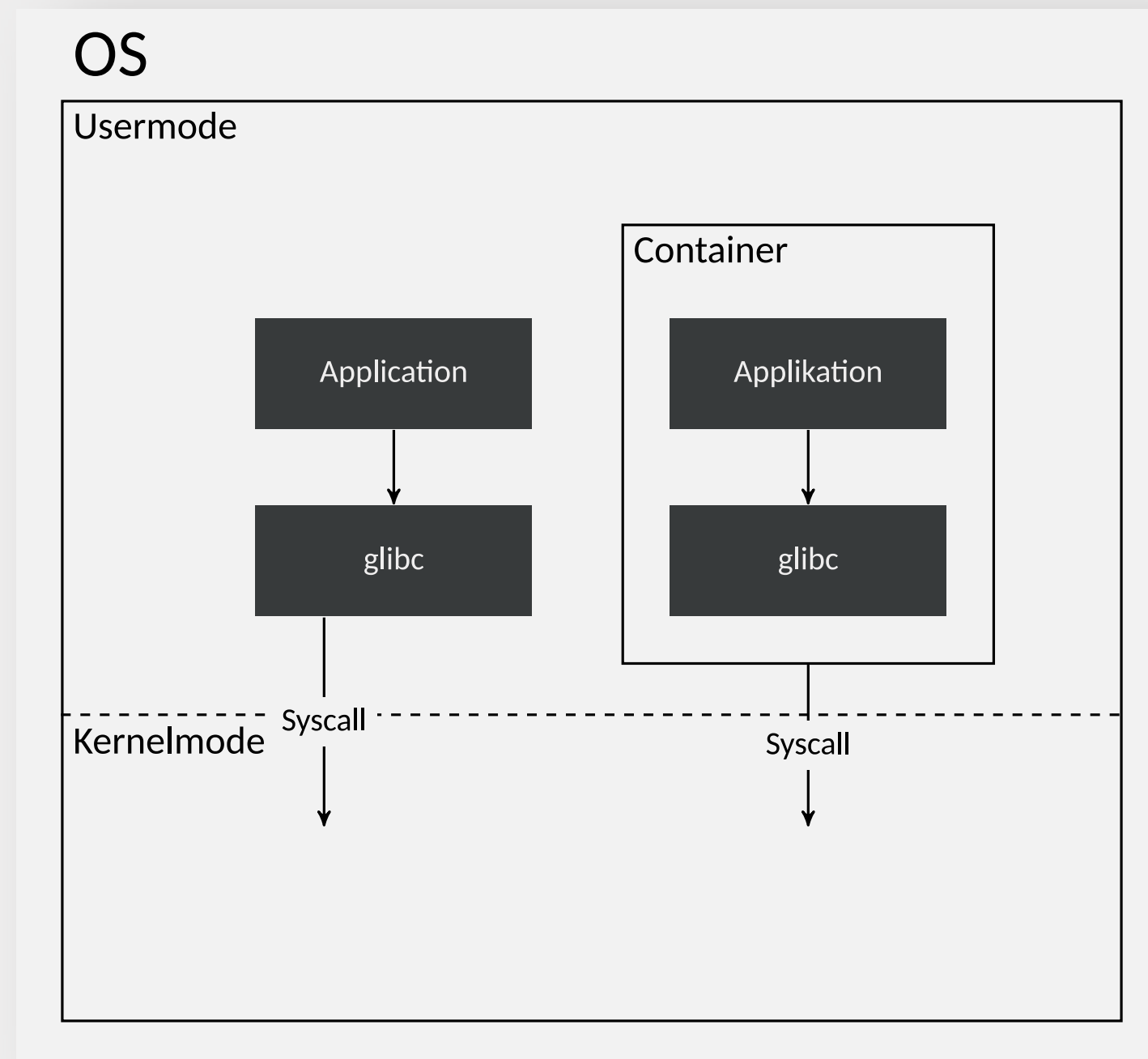
open file

glibc

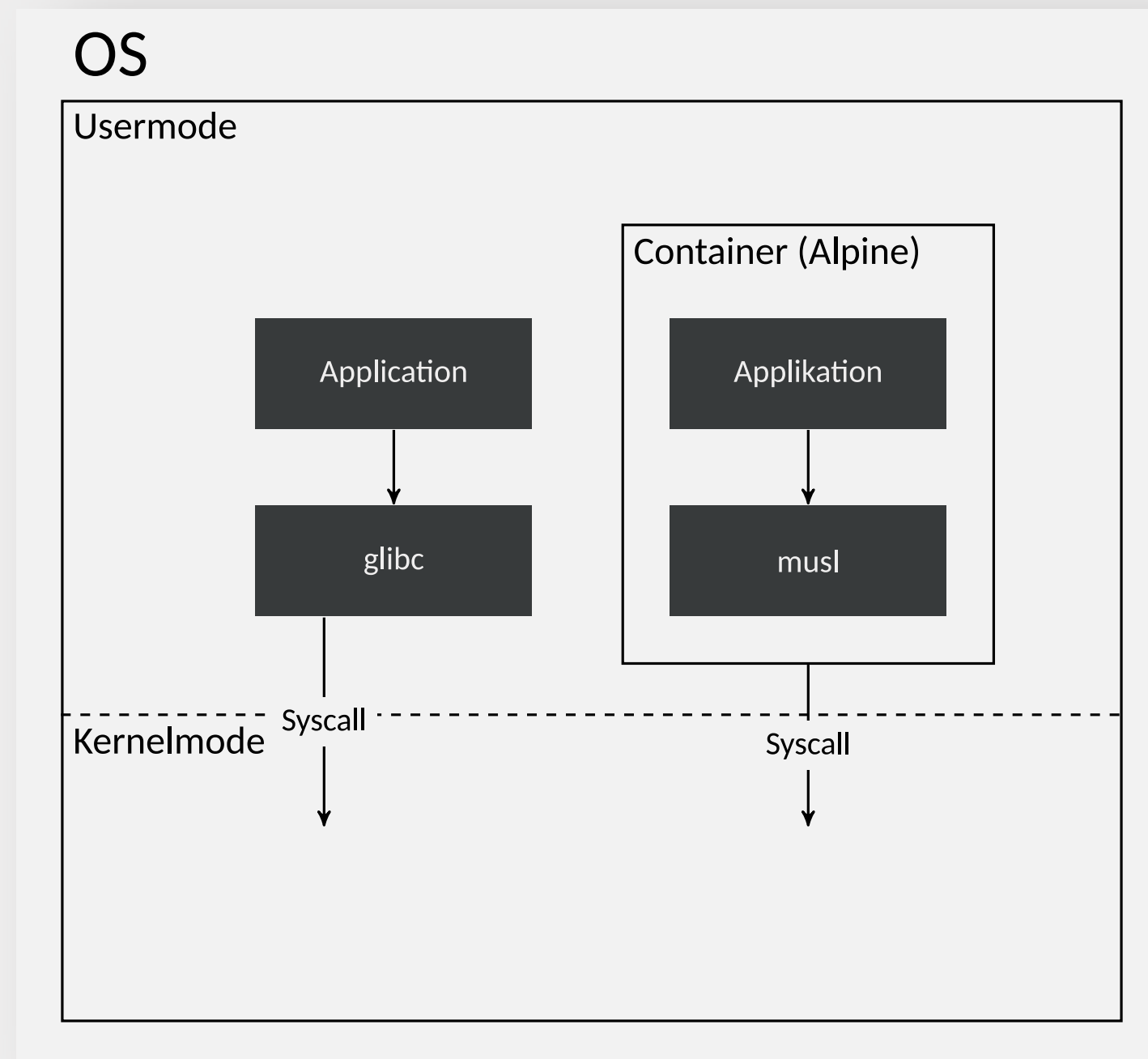
Syscall · file handle

check permission
access file system

Was ist ein Kernel?



Was ist ein Kernel?



Der Linux Kernel im Detail

Kernaufgaben

- **Prozessverwaltung:** Scheduling, Multitasking
- **Speicherverwaltung:** Virtual Memory, Paging
- **Dateisystemverwaltung:** VFS (Virtual File System)
- **Netzwerkstack:** TCP/IP, Protokolle
- **Gerätetreiber:** Hardware-Abstraktion
- **Systemaufrufe:** Interface zu User Space

Linux Distributionen

Was ist eine Linux Distribution?

- Sammlung von Software-Paketen
- Enthält Linux Kernel, System-Tools, Anwendungen
- Beispiele: Ubuntu, Fedora, Debian, Arch Linux

Linux Distributionen im Detail

Debian-Familie

- **Debian:** Stabil, konservativ, große Community
- **Ubuntu:** Benutzerfreundlich, LTS-Versionen, Corporate Support
- **Linux Mint:** Ubuntu-basiert, Windows-ähnlich

Linux Distributionen im Detail

Red Hat-Familie

- **Red Hat Enterprise Linux (RHEL):** Enterprise-fokussiert, Support
- **CentOS:** Kostenlose RHEL-Alternative (eingestellt)
- **Rocky Linux:** CentOS-Nachfolger
- **Fedora:** Cutting-edge, neue Features zuerst

Linux Distributionen im Detail

Weitere wichtige Distributionen

- **SUSE Linux Enterprise:** Enterprise-fokussiert, Europa
- **Arch Linux:** Rolling Release, DIY-Philosophie
- **Alpine Linux:** Sicherheitsfokussiert, minimal
- **Gentoo:** Source-basiert, maximale Optimierung

Distributionswahl - Kriterien

- **Zweck:** Desktop vs. Server vs. Embedded
- **Support:** Community vs. kommerzieller Support
- **Release-Modell:** Fixed vs. Rolling Release
- **Hardware-Kompatibilität:** Treiber-Unterstützung
- **Package-Manager:** apt, yum, pacman, zypper
- **Sicherheit:** Update-Häufigkeit, Security-Team

Release-Zyklen

Release-Zyklen verstehen

Rolling Release

- Kontinuierliche Updates
- Immer aktuellste Software
- Kein großer Versionssprung
- Beispiele: Arch Linux, openSUSE Tumbleweed

Release-Zyklen verstehen

Fixed Release (Point Release)

- Feste Versionszyklen (z.B. alle 6 Monate)
- Mehr Stabilität durch getestete Updates
- Ältere Software-Versionen
- Beispiele: Ubuntu, Fedora, RHEL

Release-Zyklen verstehen

Long Term Support (LTS)

- Verlängerte Unterstützung (2-10 Jahre)
- Mehr Stabilität durch getestete Updates
- Ältere Software-Versionen
- Beispiele: Ubuntu LTS, RHEL, SLES

Linux Anwendungsbereiche

Linux Anwendungsbereiche

Server & Cloud

- Webserver (Apache, Nginx)
- Datenbanken (MySQL, PostgreSQL)
- Cloud-Plattformen (AWS, Azure, GCP)

Linux Anwendungsbereiche

Entwicklung

- Programmierung und Softwareentwicklung
- DevOps und CI/CD Pipelines
- Container-Technologien (Docker, Kubernetes)

Linux Anwendungsbereiche

Embedded Systems

- IoT-Geräte
- Router und Netzwerkgeräte
- Smart TVs und Set-Top-Boxen

Linux Anwendungsbereiche

Desktop & Workstation

- Alternative zu Windows/macOS
- Wissenschaftliche Berechnungen
- Multimedia-Produktion

Open Source und Linux

Open-Source-Prinzipien

Die vier Freiheiten (nach Free Software Foundation)

- **Freiheit 0:** Das Programm zu jedem Zweck ausführen
- **Freiheit 1:** Funktionsweise studieren und anpassen
- **Freiheit 2:** Kopien weitergeben und anderen helfen
- **Freiheit 3:** Verbesserungen veröffentlichen

Vorteile von Open Source

- **Transparenz:** Quellcode einsehbar
- **Innovation:** Schnelle Entwicklung durch Community
- **Kosteneffizienz:** Keine Lizenzgebühren
- **Unabhängigkeit:** Keine Vendor-Lock-ins
- **Anpassbarkeit:** Code kann modifiziert werden

Linux Grundkonzepte

Alles ist eine Datei

- Dateien, Verzeichnisse, Geräte
- Einheitliche Schnittstelle
- `/dev`, `/proc`, `/sys`

Multi-User System

- Mehrere Benutzer gleichzeitig
- Rechteverwaltung
- Benutzer und Gruppen

Installation von Software

- Meist über Paketmanager
- Abhängigkeiten werden automatisch verwaltet
- Beispiele: apt, yum, pacman, zypper

Shell und Terminal

- Kommandozeileninterface
- Mächtige Textverarbeitung
- Automatisierung durch Skripte
- Verschiedene Shell-Varianten verfügbar

System Services (Daemons)

- Hintergrundprozesse ohne Terminal
- Starten beim Boot oder bei Bedarf
- Bieten Systemfunktionalitäten
- Verwaltet durch Init-System

systemd - Komponenten

- **systemctl**: Service-Verwaltung
- **journalctl**: Log-Verwaltung
- **networkd**: Netzwerk-Management
- **resolved**: DNS-Resolver
- **timedatectl**: Zeit/Datum-Verwaltung

```
$ systemctl status nginx    # Service-Status prüfen  
$ systemctl start nginx     # Service starten  
$ systemctl enable nginx    # Autostart aktivieren
```

Wichtige System Services

- **sshd**: SSH-Server für Remote-Zugriff
- **cron**: Zeitgesteuerte Aufgaben
- **rsyslog**: System-Logging
- **NetworkManager**: Netzwerk-Verwaltung
- **dbus**: Inter-Process Communication
- **udev**: Device Management

Zur Kapitelübersicht

- Nächstes Kapitel: [Navigation & Textverarbeitung](#)